Saarland University

Faculty of Natural Sciences and Technology I

Department of Computer and Communication Technology

Master thesis

Exploring techniques for semantic linking of semi-structured online resources

submitted by

Somnath Meignana Murthy

submitted

24-03-2015

Supervisor

Prof. Dr. Jörg Siekmann

Advisor

Dr. Sergey Sosnovsky

Reviewers

Prof. Dr. Jörg Siekmann

Dr. Sergey Sosnovsky

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig 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 convey my sincere gratitude to my advisor, Dr. Sergey Sosnovsky, Head of the Intelligent e-Learning Technology Lab in CeLTech at DFKI for his continued guidance throughout the thesis.

I have to mention his invaluable advices, comments and suggestions since the start of the thesis that helped me a lot for the successful completion of my studies. It all started with the 'Technology Enhanced Learning' seminar which I participated in the winter semester 2012 that created an interest in working in this group.

I would also like to thank my brother, Mr. Abdur Raafiu Mohamed Farook for the extended support and guidance which he has showed in this last 6 months to support me in the thesis work.

I am very thankful to University of Saarland and my friends here in Saarland especially the Gucks Team. I like to mention Pedro and Eva who helped me a lot in the initial days to get accustomed to Europe. He's the one who introduced lot of good friends who made my life in Germany much easier after first few months.

I also like to thank my very good friend Jamal and Deepak who helped my family when they were missing me.

Finally, I would like to thank my lovely parents who sent me to Germany with such a great heart to achieve my dream of doing Masters and my dear sister who always has encouraging words for me whenever I am down and make me smile.

Abstract

The usage of books has started vanishing in modern days since more and more books started migrating to digital devices. Online books are available from ancient historical subjects till currently trending modern advanced techniques. If there's a possibility that few details or nuances are not understood by a person in a book, s/he can refer to another book covering the same subject. So the link between the books which talk about the same subject helps the user a lot to get an in-depth knowledge on the subject and also clarify her/his doubts. The modern e-learning environment does this job by linking the textbooks which have similar themes or concepts. This project suggests the concept of intelligent linking of textbooks based on different techniques in different domains: Information Retrieval and Algebra. This project is done based on TF-IDF and basic Vector Space Model (VSM) with the glossary and part-of-speech technique and the final results are compared to the traditional term-based approaches and the probabilistic topic modeling known as Latent Dirichlet Allocation (LDA). NDCG values are computed for all the eight techniques with 3 different rankings (Best similarity, Top 3 and Top 10), which are then compared with the existing NDCG values for traditional term-based approaches and the probabilistic topic modeling developed along with expert-based mapping.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Many people who have access to Internet in the recent days have started using more of online books. The main reason behind the extended use of online books are: user's accessibility to wide range of books from different authors, portability, ease in searching the content and so on.

When online books have so many advantages, the user has one major setback which is "finding the right content". Since there are immense number of content present in the Web for a given subject, user faces the problem of getting the right content from hundreds and thousands of learning content, textbooks and tutorials. Even though the resources are properly organized, still it remains as a concern.

The large number of online books gives lot of opportunities and ideas to modern students. The complexity to understand the text differs in each book since users read books from different authors. So when they are not contented with the explanation of a particular topic in a primary book, they tend to refer other books to get some in-depth knowledge and detailed clarification that provides information on the same topic.

Some authors use very simple text to explain the content and some of them use advanced vocabulary. So in case of complex texts and contents in one book, user tends to refer to other books of same subjects for better understanding. But the problem in this reference is finding the right match for the content what the user is searching for.

The modern advanced e-learning environment solves this problem by pro-

viding a link between the contents in different textbooks which share the same subject. The linking of textbooks can be done based on different criteria such as matching word by word in multiple documents, by highlighting the important words or content in books which adds more relevance to the subject and then comparing them with the other book, or by just comparing them as an whole sentence as the meaning and the usage of words differs in each sentence even though the words remain the same.

For example, let us imagine that a user is interested to learn about the topic "Multifactor ANOVA" from Statistics book. The user has a primary textbook which explains the same topic. When the user cannot understand the topic clearly, he/she decides to look for detailed explanation in the other books from his e-learning system. Now the system searches for the related content in the same domain and returns the list of related topics from various books. The list of ranked links to the relevance content in the other books: a chapter titled "Multi-factor Analysis of Variance" in one book, a subsection in another book titled "Multi-Factor Between-Subjects Designs" under a chapter "Analysis of Variance", a chapter titled "ANOVA and MANOVA" in a third book.

The plan of linking documents to make the learner's job easy has already been performed by many different researchers. But many traditional approaches such as simple keyword matching and so on has failed to be perfect, since many a times it has matched to some irrelevant documents. There are also some cases where the topic on the whole is related to the users query but the content not exactly related to the doubt in the topic which the user likes to read about.

These problems in turn increases the problem of the learner as he/she is not sure in many cases whether the document linked is absolutely correct and they very rarely identify these mistakes caused by system and recover from it as they are bound to scepticism. This makes them lose their inspiration and interest in learning which in turn causes disbelief in e-learning systems.

Many systems were built for dynamic linking of texts and contents to improve the online content (e.g. COHSE [5] and Magpie [6]). These above systems are good in named-entity recognition, which locates or classifies elements such as persons, locations, organizations, dates, times, currency, etc., or just identifying the concept matching keywords in a source thesaurus or ontology and totally ignoring the rest of the content. These kinds of systems had been so successful in other domains even though they are simple, but it is not the same in case of e-learning in educational domain. One cannot

expect keywords in two random books match always as it doesn't contain specially formatted entries.

1.2 Task

In this project we propose the concept of linking the textbooks based on the basic VSM model and TF-IDF along with various techniques. The other methods which were experimented are Glossary and Part of Speech (PoS) Tagging, in different combinations with CS and TF-IDF. This research was conducted on textbooks from 2 different domains: Algebra and Information Retrieval. The eight main combinations that were tested are: 1. Basic VSM 2. VSM + Glossary 3. PoS 4. PoS + Glossary 5. TF-IDF 6. TF-IDF + Glossary 7. TF-IDF + PoS 8. TF-IDF + PoS + Glossary.

Normalized Discounted Cumulative Gain (NDCG) is calculated with all the techniques that are explained in the experimental section for all the eight different combinations mentioned above with 3 different rankings: Best similarity (NDCG@1), Top 3 (NDCG@3) and Top 10 (NDCG@10) matches. The NDCG values of our experiment are then compared with the results of NDCG values which are already computed for the traditional term-based approaches (Lucene's query-based model) and the probabilistic topic modeling obtained from [1].

1.3 Background

There are 2 different models: Lucene's query-based model and Guerra et al.'s Latent Dirichlet Allocation (LDA) model which are already performed for the dynamic linking of textbooks in [1]. This project is an extension of the work done by Guerra in [1]. The above models maps each and every chapter and section from one book to the chapters and sections in the other book. The cosine similarity and TF-IDF score calculated from the models are then matched with the expert-based mapping and the final NDCG score is computed for all the eight models.

Chapter 2

Technology

The techniques that are used in this project are explained with an example in this section. The whole process for all the eight methods implemented in this project is clearly shown in the Figure 2.1.

2.1 HTML/Webpage Parsing

The textbooks that are used for testing are saved in HTML format from the web. Initially the content from all the files present in both the books are extracted and saved in a document. This extraction is done using the Jsoup library which helps to get all the content from the HTML files or even directly from the webpage. But the extracted content is only the 7 headings and body tags from the webpage and it doesn't contain images, citation and so on.

2.2 Filtering out stopwords

Stopwords refers to words that are filtered from the natural language data prior to, or after the processing or computing. But not every natural language processing tools filters stopwords, only few. So there is no such one global list of stop words that every natural language processing tools follow. Each processing tools uses different stop words according to the domain and needs of the natural language. The most common stopwords that are used are 'a', 'the', 'is', 'in', 'at', etc.

In this approach, we have used stopwords list from Lucene. Lucene is an information retrieval library developed by Apache Software which is free open source software. The main feature of Lucene is full text indexing and searching capabilities and it is also used for the execution of Internet search engines. The four main libraries used in our approach are Lucene core, Lucene analyzers, Lucene queries and Lucene query parser and the version 3.6.2 is used for this implementation.

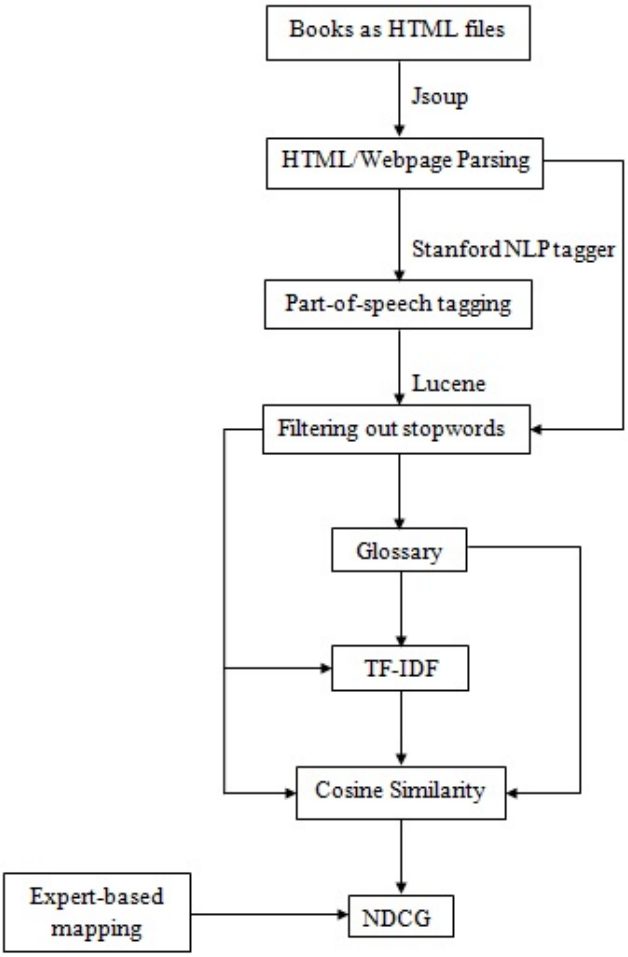

Figure 2.1: Process for all the methods

So the stopwords (Lucene) are now filtered from the content that is saved in the document earlier in HTML/Webpage parsing. Then the filtered content is now put in two different folders respectively from the 2 online books.

2.3 Cosine Similarity

There exist a number of approaches to find the dynamic linking between the documents as discussed before. Here, the Cosine similarity approach is used. Cosine similarity actually refers to calculating the cosine of the angle between two vectors of an inner product space. If the cosine angle between the vectors is 0°, then the value is 1. For all the other angles, it always remains less than 1.

In this approach, only the orientation of the vectors is taken into consideration and not the magnitude. If the orientations of both the vectors are same, then the value of Cosine similarity is 1 and if the vectors are at 90°, then the value is 0 and if the vectors are in opposite direction, then the value should be -1. But the Cosine similarity is mostly used only in positive space, so the results are bounded in [0, 1].

The efficiency of Cosine similarity can be seen especially in case of sparse vectors, as they consider just the non-zero dimensions which is one of the major reasons for the popularity of Cosine similarity approach.

The cosine of two vectors can be attained from Euclidean dot product formula:

$$a \cdot b = \parallel a \parallel \parallel b \parallel cos\theta \tag{2.1}$$

Cosine similarity, cos () can be represented with the help of a dot product and magnitude for vectors A and vector B, as

$$similarity = cos\,(\theta) = \frac{A \cdot B}{\parallel A \parallel \parallel B \parallel} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{2.2}$$

Now, the stopwords filtered files from both the books are compared with each other. The final results are compared with the manually mapped pages from both the books to check the accuracy of the cosine similarity values.

2.4 Using Glossary to detect domain keywords

Glossary or Vocabulary refers to a set of terms in a specific domain knowledge which is mentioned at the end of a book. These terms can be some specialized words or rare and newly introduced words from a book. This helps the reader to know the exact page where the words are mentioned in the book and also the definitions of those words.

Cosine Similarity with glossary words is computed for the comparison of both the books to estimate whether the accuracy improves. So for the project, glossary words are collected from various books and online references under the respective domains. The collected words are saved in a text file. The terms which have more than one word are combined as one word in both the books.

The specialized glossary words are given extra importance in the computation. So each word from the glossary file is given extra weightage during the computation. This is done so that all the normal words in the files are not given the same weightage as the specialized words present in the document.

2.5 TF-IDF

TF-IDF refers to Term Frequency - Inverse Document Frequency that is used as a weighting factor to calculate the importance of a word to the particular document in a corpus. The TF-IDF value basically increase or decrease based on the combination of two main calculations: 1. number of times a word exists in a document; 2. how frequent the particular word occurs in the corpus.

The frequency of the word in the corpus is estimated because there are few words which occur more frequently in general. These words doesn't need to be stopwords all the time and in this way TF-IDF helps to filter out the stopwords in various topics including text summarization and classification.

One of the main applications of TF-IDF is to score and rank the relevance of a document to a given query in search engines. The ranking function is computed after the scoring is individually done for each term in the document and finally summing up the TF-IDF for each term in the query.

Now, TF-IDF is explained clearly along with Cosine similarity calculation with an example below:

Example:

Three documents have been created to explain the working example and each document contains few lines of text.

Document 1: Online games are popular than field games in present days

Document 2: Google is now the most visited online website

Document 3: The Christmas days are special.

The below query is searched on the above documents: 'Days online'.

Now each step is explained in detail:

Step 1: Term frequency

Term frequency measures the number of times a term (word) occurs in a document. Now the frequency for each of the term in all the 3 documents is presented below:

TF for Documents

Doc 1	Online	games	are	popular	than	field	in	present	days
TF	1	2	1	1	1	1	1	1	1

Doc 2	Google	is	now	the	most	visited	online	website
TF	1	2	1	1	1	1	1	1

Doc 3	The	christmas	days	are	special
TF	1	1	1	1	1

The size of each document varies; henceforth the frequency of the terms in a large document is much higher than the smaller document. So the technique of normalization is introduced, which normalizes the document by dividing the frequency of the term by total number of terms that are present in the document.

For example the term 'games' occurs twice in the first document and the total number of terms in the document is 10. So the normalized TF for the term 'games' in this document is 2/10= 0.2. In the same way normalized TF is calculated below for each term in all the documents:

Normalized TF for Documents

Doc 1	Online	games	are	popular	than	field	in	present	days
Normalized TF	0.1	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Doc 2	Google	is	now	the	most	visited	online	website
Normalized TF	0.125	0.125	0.125	0.125	0.125	0.125	0.125	1

Doc 3	The	christmas	days	are	special
Normalized TF	0.2	0.2	0.2	0.2	0.1

Step 2: Inverse Document Frequency

Inverse document frequency is a measure of the number of documents in which the term is present in the corpus. It is calculated by dividing the total number of documents present in the corpus by number of documents where the particular term exists and taking the logarithm of that quotient.

$$idf\,(t, D) = log \frac{\mid D \mid}{\mid \{d \in D \,:\, t \in d\} \mid} \tag{2.3}$$

where $\mid D \mid$ is the cardinality, or the total number of document in the corpus

$\mid \{d \in D \,:\, t \in d\} \mid$ is the number of documents where the term occurs.

The need of searching a document is to find the most relevant document to the query. So if the comparison is done by giving equal importance to every term in the document, it may be wrong. There are few terms in the document which occur a lot in every document and it doesn't mean those 2 documents are relevant if it contains this term.

There are also terms that occurs very less and this terms are exactly the ones with which the comparison should be done. So the weightage is given more for the terms that occurs less in the document and less for the terms that occurs more in the document respectively. For example the IDF is calculated for the term 'games' as

$$IDF(games) = log_e \left(\frac{3}{1}\right) = 1.0986122887 \tag{2.4}$$

The total number of documents is 3 and number of documents in which the term 'games' appears is 1.

Terms	IDF
Online	0.40546510811
games	1.0986122887
are	0.40546510811
popular	1.0986122887
than	1.0986122887
field	1.0986122887
in	1.0986122887
present	1.0986122887
days	0.40546510811
Google	1.0986122887
is	1.0986122887
now	1.0986122887
the	0.40546510811
most	1.0986122887
visited	1.0986122887
website	1.0986122887
christmas	1.0986122887
special	1.0986122887

Table 2.1: IDF Calculation

Now the IDF is calculated in the same way for all the other terms from 3 documents. The terms 'online', 'are', 'the' and 'days' are having less weightage since it occurs in 2 documents.

Step 3: TF-IDF

The query searched here to find the most relevant documents are: 'Days online'. Now the TF-IDF is calculated by multiplying the TF with IDF. For example the TF-IDF for the term 'Days' in Document 1 is computed by multiplying the normalized TF for the term 0.1 with the IDF 0.40546510811 and the result is 0.040546510811. So the TF-IDF is computed for the terms 'online' and 'days' in all the documents.

Query Terms	Doc 1	Doc 2	Doc 3
Days	0.40546510811	0	0.081093021622
online	0.40546510811	0.0506831851375	0

Step 4: Cosine Similarity

First the TF-IDF for the query is calculated and then the Cosine Similarity is computed.

Query Terms	TF	IDF	TF*IDF
Days	0.5	0.40546510811	0.202732554055
online	0.5	0.40546510811	0.202732554055

For example

$$CS\left(query, doc1\right) = \frac{Dot\left(query, doc1\right)}{\parallel query \parallel \parallel doc1 \parallel} \tag{2.5}$$

$\parallel query \parallel = \sqrt{\left((0.202732554055)^2 + (0.202732554055)^2\right)} = 0.286707127479$

$\parallel doc1 \parallel = \sqrt{\left((0.040546510811)^2 + (0.040546510811)^2\right)} = 0.0573414254958$

$CS\left(query, doc1\right) = \frac{0.016440195}{(0.286707127479 \times 0.0573414254958)} = 1$

Finally, the cosine similarity is computed for all the 3 documents and the terms in the query. The results shows that the document 1 has the perfect score of 1 because both the terms 'days' and 'online' are present only in the document 1.

Cosine Similarity	Doc1	Doc2	Doc3
	1	0.707106781	0.707106781

2.6 Part-of-Speech Tagging

The research in NLP has focused on tasks that provide meaning to the structure inherent in the language without the need of complete understanding and one such task is called as PoS Tagging. Part-of-Speech tagging refers to the task of labeling each word in a sentence with its appropriate part of speech and it is also called as grammatical tagging or word-sense disambiguation.

The part-of-speech is based on the definition of the word and also the context in which it is used by relating it with the other words in a phrase, sentence or a paragraph. So each word is identified whether it is a noun, verb, adjective, adverbs, etc. A simple example of a tagged sentence:

$My/PRPcat/NNalso/RBlikes/VBZdrinking/VBGjuice/NN./.$

One of the most important applications of part-of-speech tagging is information extraction (also refers to message understanding, data extraction, or text data mining). It is also used as a pre-processing step of parsing where unique tagging to each word reduces the number of parses.

In schools there are 9 most commonly used parts of speech tags in English: adjective, adverb, article, conjunction, interjection, noun, preposition, pronoun and verb. The most commonly used tagset is an UPenn Treebank II which is a parsed text corpus that has 36 tags. The other punctuation marks that are also tagged along with the 36 different tags in the standard tagset are:

#	Pound sign
$	Dollar sign
.	Sentence-final punctuation
,	Comma
:	Colon, semi-colon
(	Left bracket character
)	Right bracket character
‘	Left open single quote
“	Left open double quote
’	Right close single quote
”	Right close double quote

2.7 Normalized Discounted Cumulative Gain

Discounted cumulative gain (DCG) is a measure to find the effectiveness of the results from web search engines or related applications and rank them according to their quality. It computes how well the document is actually related to the graded relevance scale from the results of the web search engine based on its position in the ranking list.

The effectiveness, or usefulness, or gain is decremented from top to bottom in the ranking list for each result at lower ranks. The main concept in DCG is that the highly relevant documents will be more useful when it appears in the top of a web search engine ranking list.

Discounted cumulative gain works on the principle that the highly relevant documents which appear in the lower position on the ranking list should

Number	Tag	Description
1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential there
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NNP	Proper noun, singular
15.	NNPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PRP	Personal pronoun
19.	PRP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	to
26.	UH	Interjection
27.	VB	Verb, base form
28.	VBD	Verb, past tense
29.	VBG	Verb, gerund or present participle
30.	VBN	Verb, past participle
31.	VBP	Verb, non-3rd person singular present
32.	VBZ	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb

Figure 2.2: Part-of-speech tagset in English

be given less weightage since the value of the graded relevance reduces proportionally to the position of the document on the ranking list logarithmically.

Discounted cumulative gain at a particular position p on the ranking list is defined as:

$$DCG_P = rel_1 + \sum_{i=2}^{P} \frac{rel_i}{log_2(i)} \tag{2.6}$$

where rel_i is the graded relevance of the ranking list at position i.

The size of ranking list always vary in size according to the query and the performance of the web search engine or other application cannot be uniform or even just with the DCG so the cumulative gain has to be normalized at each position for a particular value of p for all the queries.

Normalized Discounted cumulative gain (NDCG) for a query can be calculated as:

$$nDCG_P = \frac{DCG_P}{IDCG_P} \tag{2.7}$$

where IDCG refers to Ideal Discounted cumulative gain, which is calculated by ordering the documents in the ranking list from maximum possible DCG till position p according to relevance. In a perfect ranking list scenario, the NDCG will be 1.0 as the DCG and IDCG for that particular position p will be the same.

Example: A query that is searched on internet yielded some list of documents. The documents in the ranking list were present in the following order:

D1, D2, D3, D4, D5, D6, D7, D8

A sample experiment was conducted where the participants were asked to judge the relevance of the documents in the ranking list to the query searched. The participants can rate from 0-3 where 0 means totally irrelevant, 1 meaning a bit relevant, 2 meaning somewhere in between and 3 meaning completely relevant to the query searched.

The participants finally give the following relevance scores to the query:

3, 2, 2, 0, 1, 3, 1, 2

So now the DCG for each result are given in order:

$$DCG_8 = rel_1 + \sum_{i=2}^{8} \frac{rel_i}{log_2(i)} \tag{2.8}$$

i	rel_i	$log_2 i$	$\frac{rel_i}{log_2 i}$
1	3	0	N/A
2	2	1	2
3	2	1.585	1.262
4	0	2.0	0
5	1	2.322	0.431
6	3	2.584	1.161
7	1	2.807	0.356
8	2	3.0	0.667

$$DCG_8 = 3 + (2 + 1.262 + 0 + 0.431 + 1.161 + 0.356 + 0.667) = 8.877$$

Now the next step is to normalize the DCG values so first we order the documents based on the relevance of the ranking list which is as follows:

3, 3, 2, 2, 2, 1, 1, 0

So the Ideal Discounted cumulative gain or IDCG is computed as:

$IDCG_8$ = 3+ (3+1.262+1+0.861+0.387+0.356+0) = 9.866

So the final Normalized DCG for the query is:

$$nDCG_8 = \frac{DCG_8}{IDCG_8} = \frac{8.877}{9.866} = 0.8997 \tag{2.9}$$

2.8 Expert-based Mapping

Expert-based mapping refers to the mapping provided by the experts in the domain based on the exact relevance of the documents. The documents are compared and when s/he finds the degree of relevance of one document to the other, they are judged on a scale ranging from 0-3 with 0 meaning irrelevant, 1 meaning a bit relevant, 2 meaning almost relevant and 3 meaning completely relevant to each other.

The other main criterion for computing expert-based mapping is confidence. The confidence level for each mapping of relevance is judged in the scale from 1-3. If the expert is sure that the relevance of 2 documents is 100% true, then s/he gives the confidence as 3 or else the expert gives 1 or 2 based on how sure s/he feels about the mapping.

Then the confidence and relevance are multiplied with each other to get a score. The score is then used to compute the DCG by having both the

values of expert-based mapping and the values obtained from our experimental methods. The DCG calculated from the relevance of expert mapping is called as Ideal DCG.

Chapter 3

Experimental Settings

3.1 Implementation

The details regarding the types of domain, formation of glossary words from different books, different APIs and libraries used to implement the code for various techniques and the mapping done for the same books by different experts are explained in this section:

3.1.1 Domains, Corpora and Glossaries

These approaches were tested on two different domains: Elementary Algebra and Information Retrieval. Four textbooks from Elementary Algebra and five textbooks from Information Retrieval were used in the project.

Elementary Algebra Textbooks

1. Elementary Algebra, by W. Ellis & D. Burzynski.
2. Elementary Algebra - v1, by J. Redden.
3. Understanding Algebra, by J. Brennan.
4. Fundamentals of Mathematics, edited by D. Burzynski & W. Ellis.

Information Retrieval Textbooks

1. Introduction to Information Retrieval, by C. Manning, P. Raghavan & H.Schtze.

2. Modern Information Retrieval, by R. Baeza-Yates and B. Ribeiro-Neto.

3. Finding Out About, by R. Belew.

4. Information Storage and Retrieval Systems, by G. Kowalski.

5. Information Retrieval, by C. van Rijsbergen.

Once the initial mapping values are computed using each of the approaches, Normalized Discounted Cumulative Gain (NDCG) value for each approach is calculated by comparing these values with the expert-based mapping values. The experts mapped the documents by assigning scores to a level of relevance and level of confidence, both ranging from 1 to 3 (low, medium, high).

The final score is computed by multiplying the relevance and confidence levels and finally aggregating both the expert values. Since the expert values are provided only for two books in each Elementary Algebra and Information Retrieval, NDCG values in each of the approach is also calculated only for the first two books mentioned in the list in each domain.

One more main objective here is to calculate the NDCG for the most similar file (NDCG@1), top three similar files (NDCG@3) and top ten similar files (NDCG@10). Finally, these NDCG values computed are compared with the baseline values for NDCG@1, NDCG@3 and NDCG@10 in both the Elementary Algebra and Information Retrieval domain.

Glossary terms are initially collected from different sources for both the Algebra and Information Retrieval domain. The words are aggregated from different thesauri so that the glossary contains wide range of terms. For Algebra, the words are collected from the database such as mathwords, Mr.X mentor of mathematics, Department of Scientific Computing (Florida State University), quia and so on. For Information Retrieval, the terms are accumulated from the websites such as School of Computer Science (UMass), NLP group of Stanford University, Willpower information, School of Berkeley and so on.

Algebra thesauri contain 443 terms from all the above databases and Information Retrieval contains 385 in total. All these words are saved in a text file and later accessed by the experimental method to find those words in the books. Then those words are tested in the experiment with 25 different weightages in the scale ranging from 1.01 to 10 (1.01, 1.03, 1.05, 1.08, 1.1, 1.2, 1.3, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10).

3.1.2 APIs and Libraries

The 2 main APIs that are used in the experiment are Lucene and Stanford part-of-speech tagger. Lucene is a high performance open source library and is free information retrieval software engine for full featured text searches written in Java programming language by Doug Cutting.

The TFIDFSimilarity class from Apache Lucene is used in the Lucene Query-based model to calculate the similarity. The main libraries used in our approach are Lucene core, Lucene analyzers, Lucene queries and Lucene query parser and the version 3.6.2 is used for this implementation.

```
"a", "an", "and", "are", "as", "at", "be", "but", "by",
"for", "if", "in", "into", "is", "it",
"no", "not", "of", "on", "or", "such",
"that", "the", "their", "then", "there", "these",
"they", "this", "to", "was", "will", "with"
```

Figure 3.1: List of stop words in Lucene

Stanford NLP Tagger

The Stanford Natural Language Processing Group from Stanford has developed a tagger called as Stanford NLP Tagger. This tagger was originally written by Kristina Toutanova from Stanford and later its speed, performance, usability and support was improved for other languages by Anna Rafferty, Christopher Manning, Dan Klein, John Bauer, Michel Galley, and William Morgan.

This full tagger model consists of three trained models for English, and also Arabic tagger models, Chinese tagger models, French tagger models, German tagger models, and Spanish tagger models. This tagger can also be trained for any other language by providing proper training to the PoS-annotated text for the particular language.

Stanford PoS tagger is an open source software, so the Stanford-postagger jar files and the 6 different English tagger files from the software are used in this project. The different models that are trained in English are bidirectional tagger, caseless tagger and left3words tagger.

Left3words tagger is the one which we used for this project which means the words are tagged based on its definition and also by considering the relation of the word with the 3 words that are on the left of the particular word.

The props of all these English taggers are saved in a folder in the project and english-left3words-distsim.tagger is mentioned in the JAVA project. The stanford-postagger jar files are also uploaded in the library and the tagging is done by the Maxent Tagger from the jar file.

MaxentTagger tagger = new MaxentTagger("Taggers/english-left3words-distsim.tagger")

3.1.3 Expert-based Mapping

Expert-based mapping for this experiment was performed in 1. The mapping was done by 10 experts: one professor, six PhD students from the school of Information sciences (University of Pittsburgh) and three researchers from the CeLTech in DFKI. To make the judgements more specific and ambitious, only two experts mapping were taken for each chapter.

The mapping is done for 2 books from Algebra domain: Elementary Algebra, by W. Ellis & D. Burzynski and Elementary Algebra - v1, by J. Redden and 2 books from Information Retrieval domain: Introduction to Information Retrieval, by C. Manning, P. Raghavan & H.Schtze and Modern Information Retrieval, by R. Baeza-Yates and B. Ribeiro-Neto.

There are also some rules for the mapping to be done: i) every chapter, section or subsection from the books should be mapped to zero or more parts of other book; ii) mapping should be very precise; iii) each chapter, section and subsection should be mapped to every other chapter, section and subsection of other book. The experts assign the mapping a confidence level and relevance level with the scale ranging from 1-3.

Once the mapping is done, the score is calculated by multiplying the confidence and relevance levels. Then the final score is aggregated by taking the score of both the experts and the final scores are merged into a single list. The interface created by Guerra from [1] is shown in the Figure 3.2.

3.2 Ten experimental models

This project computes the similarity between the content in the textbooks using different approaches. The initial two approaches used in the study are already calculated in 1 and the remaining eight methods are computed as part of this project and the detailed explanation of all the ten models are listed and explained below:

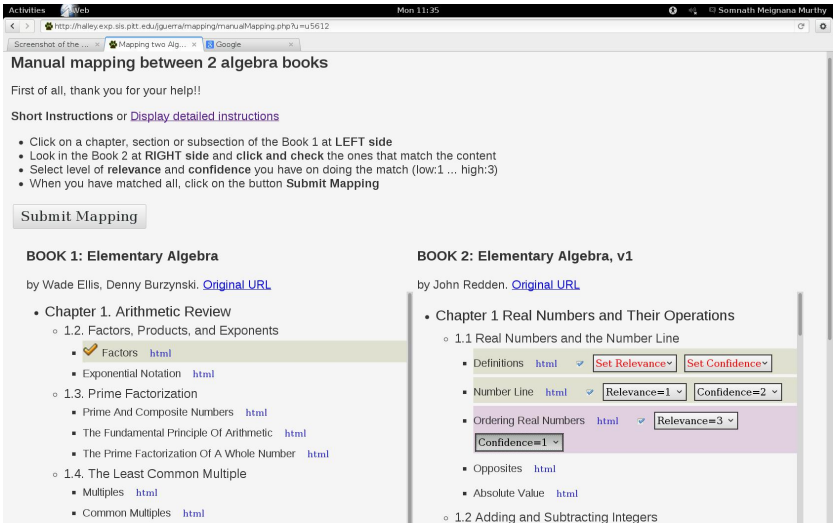

Figure 3.2: Manual mapping interface used by experts

1. Lucene's Query-based
2. Guerra et al.'s LDA based
3. Basic VSM
4. VSM + Glossary
5. PoS
6. PoS + Glossary
7. TF-IDF
8. TF-IDF + Glossary
9. TF-IDF + PoS
10. TF-IDF + PoS + Glossary

3.2.1 Lucene's Query-based

Lucene's Query-based model also known as baseline model refers to the calculation of the similarity between two files based on the TFIDFSimilarity

class from Apache Lucene. This Lucene scoring function is different when compared to simple TFIDF and it is calculated with the formula:

$$score\,(q,d) = coord\,(q,d) \cdot queryNorm\,(q) \cdot \sum_{t\,in\,q} ((tf\,(t\,in\,d) \cdot idf(t) \cdot t.getBoost\,() \cdot norm\,(t,d)) \tag{3.1}$$

where

tf(t in d) refers to term frequency and the default calculation is given as:

$$tf\,(t\,in\,d) = frequency^{\frac{1}{2}} \tag{3.2}$$

idf(t) correlates to inverse document frequency, defined as the number of documents in which the term t appears which gives high weightage to the less frequent words to the total score. The default idf(t) calculation is:

$$idf\,(t) = 1 + log\left(\frac{numDocs}{docFreq+1}\right) \tag{3.3}$$

coord(q,d) measures the number of times the query terms appears in the specified document. More the number of query terms in the document, higher the score compared to other documents.

queryNorm(q) normalizes the queries to make the scores comparable and it is computed at the search time. The default computation is given as:

$$\begin{aligned} queryNorm\,(q) &= queryNorm\,(sumofSquaredWeights) \\ &= \frac{1}{(sumofSquaredWeights)^{\frac{1}{2}}} \end{aligned} \tag{3.4}$$

The Boolean query computes the sum of squared weights as:

$$sumofSquaredWeights = q.getBoost\,()^2 \cdot \sum_{t\,in\,q} (idf\,(t) \cdot t.getBoost\,())^2 \tag{3.5}$$

t.getBoost() is a boost in the search time to find the term t in the query.

norm(t,d) contains Field boost and lengthNorm. These factors are multiplied when the document is added to index and the boosts are multiplied together, if the documents have multiple fields with the same name.

$$norm\,(t,d) = lengthNorm \cdot \prod_{field\,f\,in\,d\,named\,as\,t} f.boost\,() \tag{3.6}$$

The final norm(t,d) is encoded and stored as a single byte.

3.2.2 Guerra et al.'s LDA based

Latent Dirichlet Allocation (LDA) is one of the most popular methods to link the online textbooks dynamically based on probabilistic topic modeling. This approach produces a better result of section-level textbook linking of the content compared to the existing traditionally used term-based approaches. Each document is a combination of multiple topics and each topic is a combination of multiple words in LDA model. So the algorithm assigns each word in a document to a topic, which in turn reduces the number of highly probable topics in a document.

Once the input is given with number of topics, every document in the model is represented as a probability vector of all the topics and every topic as probability vector of all the words in the vocabulary. Finally all the chapters, sections and subsections are represented in a vector form which can be then used to compute the similarity between the documents. This model uses 2 different approaches: Topic Aggregation (TA) and Re-Indexing (RI) and they have also tested the system with single book and multiple books.

Algebra

	NDCG@1			NDCG@3			NDCG@10		
Baseline	*.3662*			*.5807*			*.6582*		
	Mean	Std. Dev.	Sig. (p)	Mean	Std. Dev.	Sig. (p)	Mean	Std. Dev.	Sig. (p)
SB-TA	.547	.025	<.001	.647	.018	<.001	.691	.015	<.001
MB-TA	.532	.036	<.001	.620	.021	<.001	.663	.017	.165
SB-RI	.456	.027	<.001	.601	.027	<.001	.675	.019	<.001
MB-RI	.414	.040	<.001	.572	.032	.132	.647	.026	.022

Information Retrieval

	NDCG@1			NDCG@3			NDCG@10		
Baseline	*.057*			*.186*			*.258*		
	Mean	Std. Dev.	Sig. (p)	Mean	Std. Dev.	Sig. (p)	Mean	Std. Dev.	Sig. (p)
SB-TA	.345	.051	<.001	.461	.042	<.001	.536	.033	<.001
MB-TA	.309	.063	<.001	.418	.045	<.001	.520	.039	<.001
SB-RI	.360	.066	<.001	.484	.053	<.001	.556	.045	<.001
MB-RI	.336	.050	<.001	.456	.054	<.001	.534	.041	<.001

Figure 3.3: Latent Dirichlet Allocation and baseline results

Topic Aggregation refers to aggregating the topic distribution of all the subsections to find the topic distribution of a section. Re-Indexing means once the topic aggregation is done for a section, the aggregated content of all

the documents is re-indexed once again using the topic model. This is done because now the aggregation is done with all the subsections in a section and finally a new LDA model is built.

When the LDA model is compared with the Lucene's Query-based model it gives the result as in Figure 3.3.

3.2.3 Basic VSM

Basic VSM also known as simple Cosine Similarity is the measure of the cosine of the angle between two vectors of an inner product space. The two Algebra books Elementary Algebra, by W. Ellis & D. Burzynski and Elementary Algebra - v1, by J. Redden are saved in the form of HTML files initially. These files needed to be parsed and the content from the HTML files has to be retrieved before comparison.

The Jsoup parser is used for HTML parsing and the data extraction. Jsoup is a Java library which can be used to parse HTML from a URL or a file, find or extract data and manipulate it. The headers (h1, h2, h3, h4, h5, h6 and h7) and the body are the content extracted from the HTML files as done in the Statistics domain.

Technical procedures were executed in succession on the contents of the textbooks: converting the whole text to lowercase, discard the stop-words from Lucene in all the text files, also additionally remove all the frequently occurring words in the domain (for e.g.: "examples", "exercises", "solutions" in Algebra).

Stopwords removal from the content is one of the main techniques that should be performed because the presence of these words impact the similarity values a lot. As these words exist many a times in every file, the similarity score will rise up which actually should not happen as the files in this case doesn't have to be similar all the time.

So the content from two stopwords removed books are now saved in two different folders Book1 and Book2 respectively. Each file from Book1 is compared with every other file from Book2 using the first approach, basic VSM. So this step is repeated for every file in the Book1.

Once the similarity is calculated, the NDCG values are aggregated for the whole book in 3 different combinations as discussed above. So the NDCG@1, NDCG@3 and NDCG@10 are computed for this Cosine Similarity approach.

Then the two Information Retrieval books Introduction to Information Retrieval, by C. Manning, P. Raghavan & H.Schtze and Modern Information

Retrieval, by R. Baeza-Yates and B. Ribeiro-Neto are also compared using the Cosine similarity approach in the same way as Algebra books.

NDCG@1, NDCG@3 and NDCG@10 are computed from the DCG values for these above books. The NDCG values for all the 8 approaches are presented and discussed in the results chapter.

3.2.4 VSM + Glossary

VSM + Glossary is an approach very similar to simple vector space model but the only difference is the addition of Glossary terms. At first Glossary terms are collected from the four books from Algebra domain and five books from Information Retrieval domain respectively. In addition few important words are also accumulated from the Internet in both the domains.

Glossary terms collected are then saved in a text file. Cosine Similarity is calculated by comparing terms one by one in both the files. So since each term/word is compared and computed individually, glossary terms also should be a single word. But all the glossary terms in the file are not single words; some exist as 2 words or more than that (for e.g. Arithmetic Series, Average Rate of Change, and so on).

In such cases, we combine it as one whole term (ArithmeticSeries or AverageRateofChange) by removing the spaces in all the files. In this way the glossary words from all the files can be easily compared with the collected Algebra and Information Retrieval thesauri text files respectively.

Once the above procedures for Glossary are done, the Cosine Similarity is again calculated similar to the previous experiment. But this time the value for a term which exists in both the files are increased more than the general value of 1.

The weightage are implemented and tested for 25 different values from 1.01 to 10 (1.01, 1.03, 1.05, 1.08, 1.1, 1.2, 1.3, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10) to check how the accuracy of NDCG changes with each value.

These 25 different values are implemented for each NDCG@1, NDCG@3 and NDCG@10 and they are plotted as graph in the results section. The NDCG output values are not in a linear or exponential rise for all these 25 values. Hence all these values are needed to be executed for each NDCG in order to find the best NDCG value possible.

3.2.5 PoS

The files from both the books are parsed and saved in two different folders but this time the stopwords are not applied in this step. So the whole book is extracted since part of speech tagging cannot be performed on the sentences in which the stopwords are removed.

Part-of-speech taggers are installed and all the files from both the books are tagged. Then all the stopwords and other unwanted words are manually collected and removed from the files as it is not done in the initial stages.

Finally both the books are compared by simple cosine similarity technique once the tagging is completed. This same method is applied to both the domains: Information retrieval and Algebra.

3.2.6 VSM + POS + Glossary

The only change in this method compared to the previous method is the inclusion of Glossary terms and adding extra weightage to them. The files extraction without the removal of stopwords is performed in the same way as previous method.

Once the part-of-speech tagging is executed, the stopwords are removed in the same way. In addition to the stopwords the glossary terms are made as one term in all the files and simple cosine similarity is applied to estimate the results.

These glossary terms are tested by giving different weightages from 1.01 to 10 (25 different weightages). These different weightages are calculated for NDCG@1, NDCG@3 and NDCG@10 and are then plotted as graphs in the result section to evaluate the variation in NDCG values at each level.

3.2.7 TF-IDF

The method is the same as basic VSM along with TF-IDF technique. The files are extracted and stopwords are removed and then TF-IDF is performed. A linked list with the name allTerms is created which contains arrays of arrays. All the files from both the folders are taken as individual array so there will be so many numbers of arrays as many numbers of files. Now these arrays contain individual arrays in which each of the words from the respective files are saved.

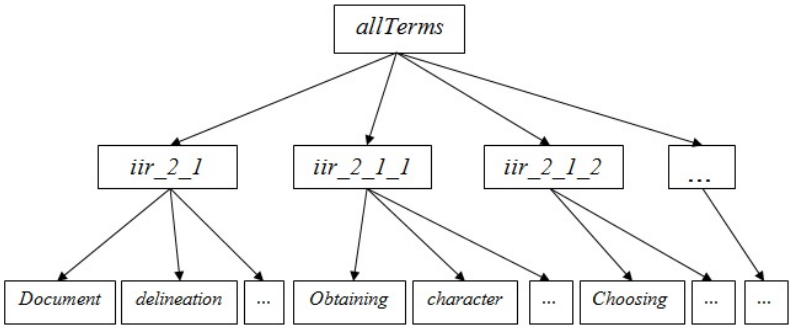

Figure 3.4: Linked list implementation of TF-IDF doc vector in the project

Once the files are saved in the linked list, the term frequency and inverse document frequency values are calculated for each of the word in every single file. Then the cosine similarity is calculated by taking the value for each term from the calculated TF-IDF values.

3.2.8 TF-IDF + Glossary

The method is the combination of basic VSM + Glossary method along with TF-IDF method. The glossary terms are made as one term in all the files and the glossary thesauri file and then the cosine similarity with TF-IDF values are computed.

The final NDCG value is computed in the scale ranging from 1.01 to 10 for 25 different weightages and the graphs are provided and discussed in the results and discussion section.

3.2.9 TF-IDF + PoS

The method is the combination of Part-of-Speech Tagging with basic VSM and TF-IDF. The files are parsed without the removal of stopwords and part-of-speech tagging is applied to all the files. Then the stopwords are manually removed from the files.

This procedure is followed by the steps in TF-IDF method. The term frequency and inverse document frequency are calculated and the final TF-IDF values are again used to compute the NDCG values with the cosine similarity.

3.2.10 TF-IDF + PoS + Glossary

The method is the combination of Part-of-Speech Tagging with Glossary evaluation and TF-IDF. The whole parsed files undergo the tagging from Stanford Part-of-Speech tagger which is followed by the stopwords removal.

The importance of this method of collecting the glossary thesauri and combining the terms are executed. Then the glossary terms are integrated as single terms in both the folders of the books and the glossary thesauri text file.

Finally the TF-IDF values are calculated for each term for each book in both the folders and the ultimate NDCG value is computed. In similar to Part-of-Speech Tagging with CS and Glossary, the three NDCG values (NDCG@1, NDCG@ and NDCG@10) are calculated for different weightages in the scale ranging from 1.01 to 10 and these values are plotted as graph.

The results of all these above discussed eight methods along with graphs are discussed in the next section.

Chapter 4

Results and Discussion

4.1 Introduction

The results of the experiment to find the better ways of comparing the files in a domain and identifying the best match for a particular file are provided and discussed in this section. The experiment performed has attempted a list of 8 methods as mentioned above in the previous sections which are: 1. Basic VSM; 2. VSM + Glossary; 3. PoS; 4. PoS + Glossary; 5. TF-IDF; 6. TF-IDF + Glossary; 7. TF-IDF + PoS; 8. TF-IDF + PoS + Glossary.

The experiment results are then compared with the baseline results which are performed by calculating the similarity using the modified TF-IDF version used in Lucene JAVA library. The NDCG (NDCG@1, NDCG@3 and NDCG@10) values of the baseline results are obtained from the background section for 2 domains: Algebra and Information Retrieval and the outputs are explained below.

4.2 Algebra

The outcome of the experimental results for the domain Algebra is explained and discussed in this section in detail with the Table 4.1.

Algebra				
		NDCG@1	NDCG@3	NDCG@10
Lucene's Query-based		.366	.581	.658
Guerra et al.'s LDA based	SB-TA	.547*	.647*	.691*
	MB-TA	.532	.620	.663
	SB-RI	.456	.601	.675
	MB-RI	.414	.572	.647
Basic VSM		.307	.515	.566
VSM + Glossary		.376	.514	.577
PoS		.252	.335	.397
PoS + Glossary		.235	.330	.392
TF-IDF		.429	.614	.690
TF-IDF + Glossary		.446	.621	.684
TF-IDF + PoS		.402	.602	.665
TF-IDF + PoS + Glossary		.390	.591	.643

Table 4.1: Best NDCG values for all the methods in Algebra

The results from Table 4.1 are also tested with the Latent Dirichlet Allocation (LDA) results for the same domain from the experiments performed by Sergey Sosnovsky, Julio Guerra and Peter Brusilovsky in [1]. Lucene's Query-based provides very good results for the Algebra domain. The outcome of the first four methods in this experiment which is tested just with the basic VSM have less NDCG values compared to the other methods such as Lucene's Query-based and Guerra et al.'s LDA results.

In the first 4 methods, PoS + glossary has the worst values followed by PoS + basic VSM. The values for NDCG@1, NDCG@3, and NDCG@10 in basic VSM are 0.3069, 0.5152 and 0.5659 respectively.

Generally, the glossary terms and part-of-speech tagging in a method is usually expected to increase the performance and the NDCG values in the experiment. Instead if both the part-of-speech tagging and glossary are applied together, the values at NDCG@1, NDCG@3 and NDCG@10 are least compared to the other methods (especially the basic VSM method).

The glossary with VSM has been experimented with 25 different weightages as specified above and the results for each of NDCG@1, NDCG@3 and NDCG@10 are plotted in the Figure 4.1, 4.2 and 4.3. The highest value is also marked and labelled with the weightage and NDCG value.

The best value for NDCG@1 (0.3762) is at the weightage point 4.5,

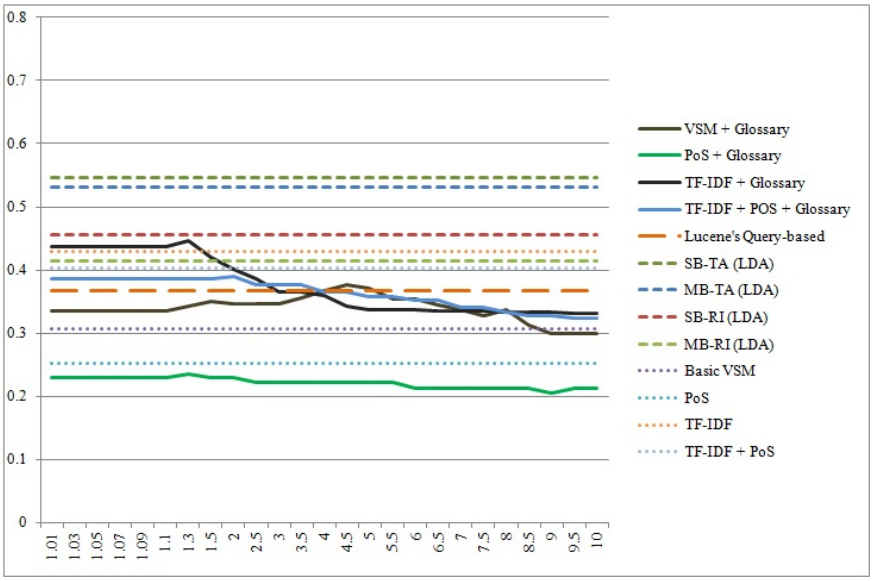

Figure 4.1: Results of all the methods for NDCG@1 in Algebra

NDCG@3 (0.5138) is at the weightage point 1.5 and NDCG@10 (0.5771) is at the weightage point 2.5. The glossary terms added to the VSM has increased the NDCG values as expected but still it is way less than the other methods.

The weightage points are not the same for all the highest values in each NDCG but the value becomes steady in the range from 0-5 and after 5 till 10 the values are poor. The part-of-speech tagging performed on the files has surprisingly reduced the NDCG values. The values for NDCG@1, NDCG@3, and NDCG@10 in part-of-speech with VSM model are 0.3069, 0.5152 and 0.5659 respectively.

The NDCG values has decreased further when the glossary terms are added to the part-of-speech tagging and the results are clearly explained with all the 25 different weightage values. The best value for NDCG@1 (0.2345) is at the weightage point 1.3, NDCG@3 (0.3303) is at the weightage point 2.5 and NDCG@10 (0.3917) is at the weightage point 2.5. The values are the highest for the PoS and glossary in the scale 1-2.

In the last 4 methods, the values have improved a lot due to the addition of TF-IDF concept. The TF-IDF with VSM method gives a much better results of 0.4291, 0.6139 and 0.6896 for NDCG@1, NDCG@3, and NDCG@10

respectively compared to the Lucene's Query-based and Guerra et al.'s LDA method described in the background section.

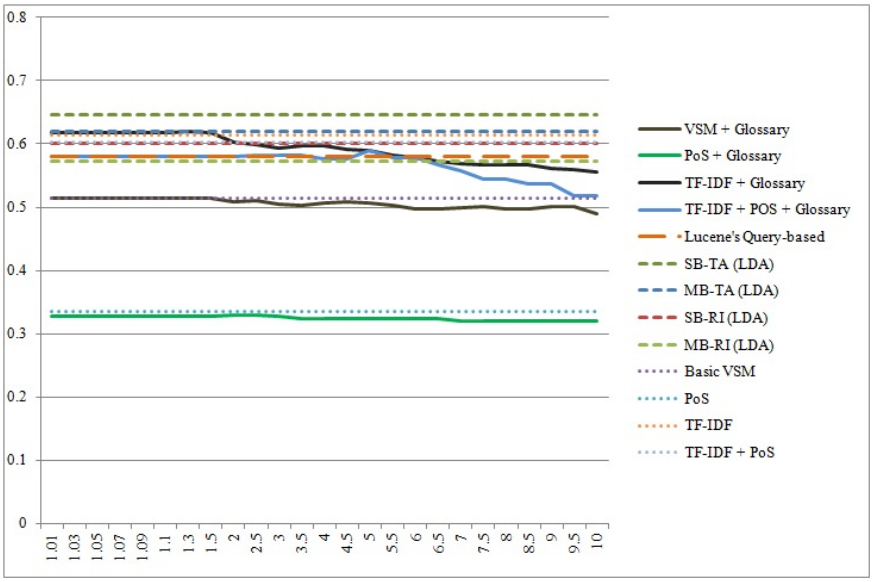

Figure 4.2: Results of all the methods for NDCG@3 in Algebra

The next method is the introduction of glossary terms to this TF-IDF concept which is expected to give a better result than simple TF-IDF and again the weightage for the glossary terms are tested with 25 different weightage values to find the best one and the results are plotted again.

The glossary with TF-IDF results have the best NDCG values among all the methods for Algebra domain of 0.445 at the weightage point 1.3, 0.621 at the weightage point 1.3, and 0.684 at the weightage point 1.05 for NDGC@1, NDCG@3 and NDCG@10 respectively.

The part-of-speech tagging concept as experienced with the basic VSM model has lower NDCG values for PoS tagging with TF-IDF of 0.4022, 0.6010 and 0.6653 for NDCG@1, NDCG@3, and NDCG@10 respectively compared to the simple TF-IDF but the advantage is that these values are still better.

The next method is the combination of all the methods: TF-IDF with CS, PoS tagging and glossary which is experimented with 25 different weightage values and the results are plotted. The results shows that the values of this method provides the poorest results among all TF-IDF methods of 0.3899

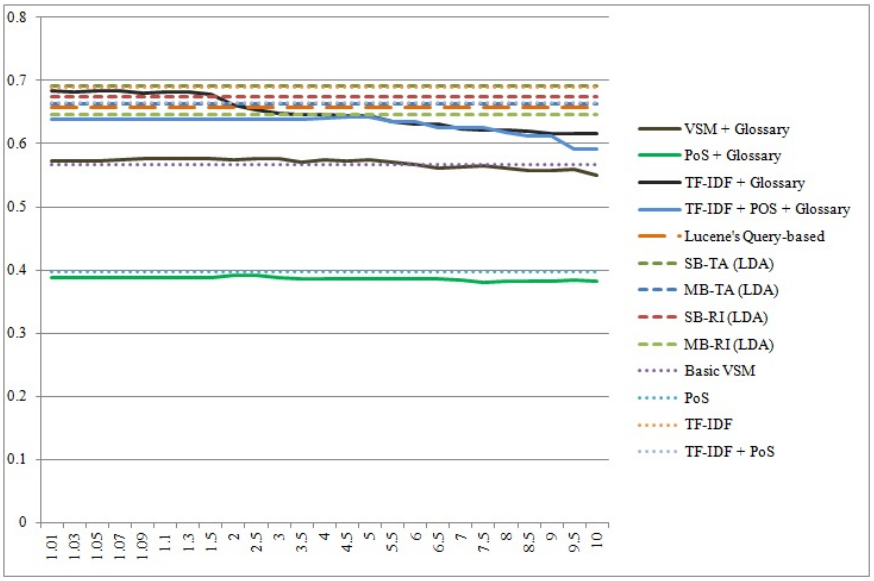

Figure 4.3: Results of all the methods for NDCG@10 in Algebra

at the weightage point 2, 0.5905 at the weightage point 5, and 0.6425 at the weightage point 4.5 for NDGC@1, NDCG@3 and NDCG@10 respectively.

Figure 4.1, 4.2 and 4.3 explains the results of all the methods in detail for 25 different weightages in Algebra for NDCG@1, NDCG@3 and NDCG@10 respectively.

4.3 Information Retrieval

The baseline value for Information Retrieval is very less and the result of all the 8 methods provides much better NDCG values for this domain. The first four methods with simple cosine similarity are better than baseline but the NDCG values for TF-IDF methods are much better than Lucene's Query-based and Guerra et al.'s LDA method methods.

The outcome of the experimental results for the Information Retrieval domain is explained and discussed in this section in detail with the Table 4.2.

Information Retrieval				
		NDCG@1	NDCG@3	NDCG@10
Lucene's Query-based		.057	.186	.258
Guerra et al.'s LDA based	SB-TA	.345	.461	.536
	MB-TA	.309	.418	.520
	SB-RI	.360	.484	.556
	MB-RI	.336	.456	.534
Basic VSM		.193	.359	.445
VSM + Glossary		.218	.372	.462
POS		.229	.356	.438
POS + Glossary		.244	.375	.458
TF-IDF		.231	**.500**	**.572**
TF-IDF + Glossary*		.289	**.524***	**.592***
TF-IDF + POS		.233	**.501**	**.565**
TF-IDF + POS + Glossary		.285	**.509**	**.571**

Table 4.2: Best NDCG values for all the methods in IR

The NDCG values for basic VSM of 0.1929, 0.3594 and 0.4450 for NDCG@1, NDCG@3 and NDCG@10 respectively are the lowest among all the methods but still the values are almost double of Lucene's Query-based values. When the glossary is added to the VSM, the values are slightly better and these are experimented with 25 standard glossary weightage values as explained in the Figure 4.4, 4.5 and 4.6.

The best value for NDCG@1 (0.2179) is at the weightage point 2, NDCG@3 (0.3719) is at the weightage point 2 and NDCG@10 (0.4618) at the weightage point 1.5. The NDCG values are very less for higher weightage values and it is steady and higher only in the range 1-2.

The part-of-speech tagging with VSM unlike the Algebra domain gives a good higher NDCG value than the basic VSM of 0.2285, 0.3555 and 0.4377 for NDCG@1, NDCG@3 and NDCG@10 respectively. The concept of PoS tagging with glossary is then tested with 25 standard weightages.

The results are the best out of the first four methods in this domain which are completely opposite to Algebra domain. The values are steady and higher in the range from 1-3 and the best values are at weightage points 2.5 for NDCG@1 (0.2436), 2 for NDCG@3 (0.3751) and 3 for NDCG@10 (0.4575).

The NDCG values with the TF-IDF concept are completely the best when

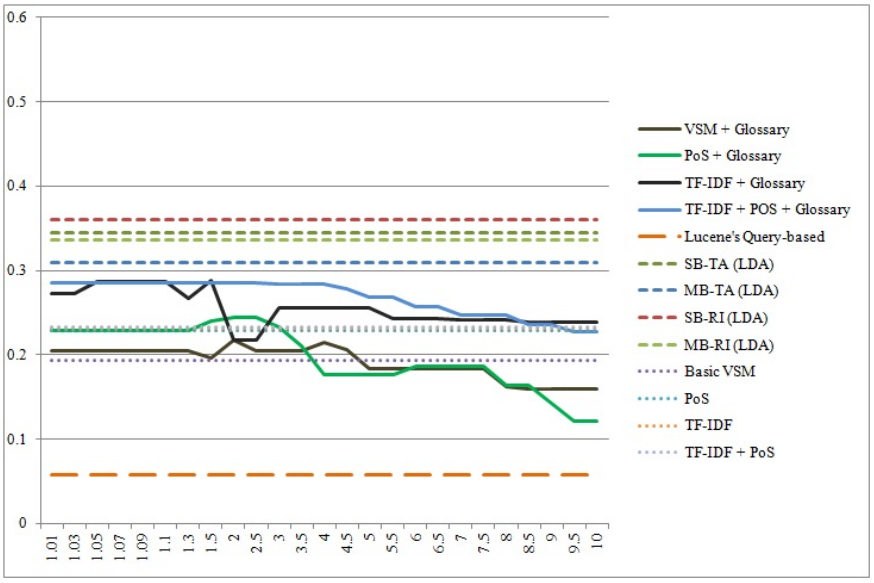

Figure 4.4: Results of all the methods for NDCG@1 in IR

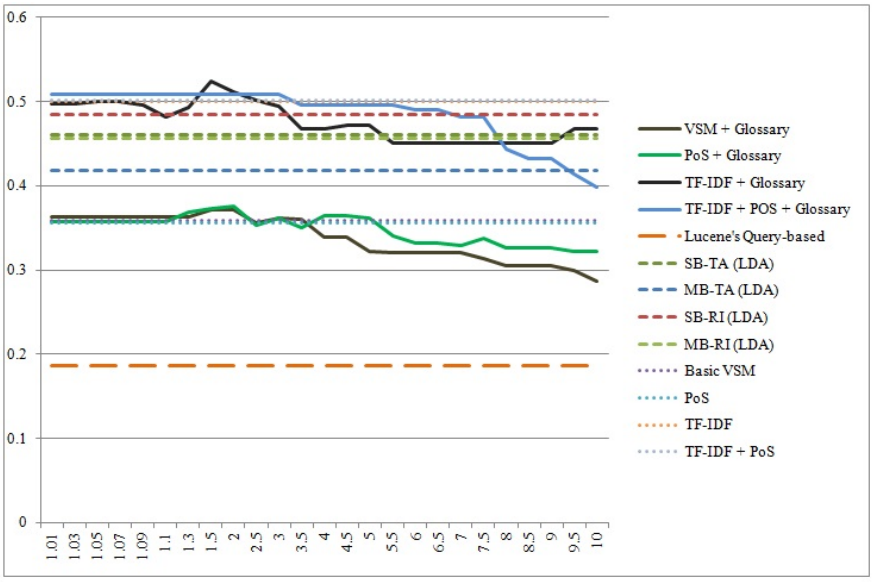

Figure 4.5: Results of all the methods for NDCG@3 in IR

compared to both the Lucene's Query-based and Guerra et al.'s LDA method values. The values for NDCG@1, NDCG@3, and NDCG@10 in the simple TF-IDF model are 0.2305, 0.4998 and 0.5716 respectively. Then the glossary terms and weightage values are added to TF-IDF and tested are explained clearly in Figure 4.4, 4.5 and 4.6.

The glossary added to TF-IDF has increased the NDCG value significantly to 0.2885 (NDCG@1) at the weightage values 1.5, 0.5239 (NDCG@3) at the weightage values 1.5, and 0.5916 (NDCG@10) at the weightage values 1.5. The scale for TF-IDF with glossary is consistent at the point 1.5.

The part-of-speech with TF-IDF model has once again showed better results in the domain with the results of 0.2325, 0.5010 and 0.5650 for NDCG@1, NDCG@3 and NDCG@10 respectively. Finally the method with the combinations of all TF-IDF, PoS and glossary are experimented with the standard 25 weightage values.

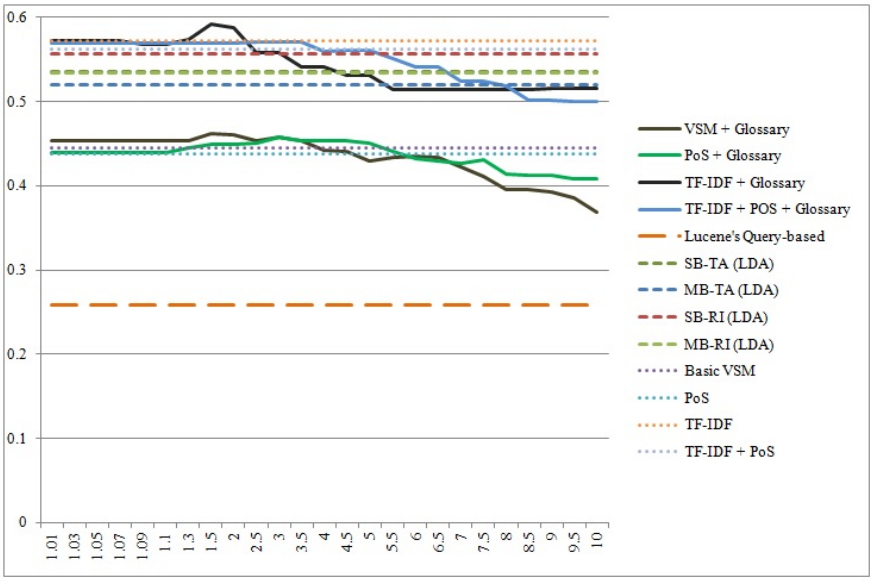

Figure 4.6: Results of all the methods for NDCG@10 in IR

The best value for NDCG@1 (0.2847) at the weightage point 2.5, NDCG@3 (0.5090) at the weightage point 3 and NDCG@10 (0.5711) at the weightage point 3 is the second best method in this Information Retrieval domain providing such successful results better than the Lucene's Query-based and

Guerra et al.'s LDA based values. The scale for TF-IDF with PoS and Glossary is the best from 2-3.

Chapter 5

Conclusion

In this project we have examined eight different methods for the dynamic linking of textbooks. These methods have been tested on two different domains: Algebra and Information Retrieval. The approach is mainly based on TF-IDF which gives better results for the linking of chapters or sections in textbooks based on the similarity. There are also other methods such as Glossary and Part-of-Speech tagging which were executed and tested along with TF-IDF.

The results in the Algebra domain suggest that the 4 methods with TF-IDF give a better output as the Lucene's Query-based and also as good as the Guerra et al.'s LDA results. In case of Information Retrieval domain, the output of 4 methods with basic VSM is better than the Lucene's Query-based method and the other 4 methods with TF-IDF produces a much better output than Latent Dirichlet Allocation method.

The output of the methods with glossary is tested with 25 different weightages in the scale ranging from 1.01 to 10. The results in all the different graphs clearly indicate that the best results are obtained in the scale from 1.01 to 3. This explains that when the weightage values are increased, the NDCG values keeps decreasing after the range of 3.

So the results of all our methods also suggest that the performance of the methods depends also on the domain and not just on the techniques such as part-of-speech tagging and glossary even if these techniques produce better results along with TFIDF.

Bibliography

[1] Guerra, J., Sosnovsky, S., and Brusilovsky, P., When One Textbook is not Enough: Linking Multiple Textbooks Using Probabilistic Topic Models, *Proceedings of EC-TEL 2013: Eight European Conference on Technology Enhanced Learning, Paphos (Cyprus)*, 8095,(2013)125-138.

[2] Yang, Y., and Chute, C.G., An Example-Based Mapping Method for Text Categorization and Retrieval, *ACM Transactions on Information Systems (TOIS)*, 12,(1994)252-277.

[3] Aizawa, A., An information-theoretic perspective of TF-IDF measures, *Information Processing and Management*, 39,(2003)45-65.

[4] Fountain, A., Hall, W., Heath, I., and Davis, H., MICROCOSM: an open model for hypermedia with dynamic linking, *Hypertext: concepts, systems and applications*,(1992)298-311.

[5] Bechhofer, S., Goble, C., Carr, L., Kampa, S., Hall, W. and De Roure, D., COHSE: Conceptual Open Hypermedia Service, *Annotation for the Semantic Web* S. Handschuh & S. Staab (Eds.),(1992)193-211.

[6] Domingue, J., Dzbor, M., and Motta, E., Magpie: supporting browsing and navigation on the semantic web, *Proceedings of 9th International Conference on Intelligent User Interfaces (IUI 04), Funchal, Madeira, Portugal*,(2004)191-197.

[7] Yang, Y., and Chute, C.G., An example based mapping method for text categorization and retrieval, *ACM Transaction on Information Systems (TOIS)*, 12,(1994)252-277.

[8] Jednoralski, D., Melis, E., Sosnovsky, S., and Ullrich, C., Gap Detection in Web-based Adaptive Educational Systems, *Proceedings of the 9th*

International Conference on Web-based Learning (ICWL2010), Shanghai, China, Dec. 8-10, 2010, 6483,(2010)111-120.

[9] Mayes, J. T., Kibby, M. R., and Watson, H., StrathTutor: The development and evaluation of a learning-by-browsing on the Macintosh, *Computers & Education*, 12,(1988)221-229.

[10] Li, H., Learning to Rank for Information Retrieval and Natural Language Processing, *Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers*, 2,(2011)123.

[11] Yilmaz, E., Kanoulas, E., and Aslam, J. A., A simple and efficient sampling method for estimating AP and NDCG, *SIGIR 08 Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*,(2008)603-610.

[12] Valizadegan, H., Jin, R., Zhang, R., and Mao, J., Learning to Rank by Optimizing NDCG Measure, *Advances in Neural Information Processing Systems*, 22,(2009)1883-1891.

[13] Brill, E., A simple rule-based part of speech tagger, *ANLC '92 Proceedings of the third conference on Applied natural language processing, ACL, Trento, Italy*,(1992)152-155.

[14] Brill, E., Automatic grammar induction and parsing free text: A transformation-based approach, *Proceedings of the 31th Annual Meeting of the ACL*,(1993)259-265.