# ELECTRA Workshop on Methodologies and Evaluation of Lexical Cohesion Techniques in Real-world Applications (Beyond Bag of Words)

(in association with the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval)

Pestana Bahia, Salvador, Brazil

August 19, 2005

http://research.yahoo.com/workshops/electra2005/

# WORKSHOP DESCRIPTION

Lexical cohesion can be subdivided into two distinct areas: (1) lexical associations, that embody a wide spectrum of language phenomena such as named entities, multiword units, collocations and word co-occurrences and (2) lexical relations that provide evidence of the semantic and discourse structure of text through relations between terms over large distances. The central goal of this workshop is to bring together researchers in NLP and IR to discuss the use of lexical cohesion in text applications, such as document and passage retrieval, question answering, topic segmentation and text summarization. Indeed, despite the fact that both communities are working with the same material (human language), collaboration between them has so far been limited. In this workshop we are interested in pointing at successes and failures of the integration of lexical cohesion in real-world IR applications. On the one hand, lexical cohesion has received much attention in Information Retrieval research during its more than 30-year old history, but so far with mixed results. On the other hand, a considerable amount of research has been devoted to this subject, both in terms of theory and practice, by the Natural Language Processing community, but with limited evaluation in real-world applications. It is clear that we are at a point where both communities should meet in order to discuss related issues. This is the objective of this workshop. In particular, we will address two questions that are of great importance for real-world IR applications.

(1) Efficient methodologies for Lexical Cohesion identification

Lexical cohesion has received attention in IR research since its outset. We can point to (a) the identification and the use of multiword units for indexing and search, and (b) the extraction of long-distance lexical relations for tasks such as passage retrieval, topic segmentation or text summarization.

On the one hand, the interest in multiword units (or phrases) can be partially attributed to the fact that phrases typically have a higher information content and specificity than single words, and therefore represent the concepts expressed in text more accurately than single terms.

On the other hand, interest in long-distance lexical relations in text has been motivated in IR research by the realization of the limitations of most IR models that assume term independence in text. As a consequence, a number of techniques have been developed to improve term independence models, such as passage retrieval and query expansion techniques.

The choice of the methodologies and techniques for these tasks has always been restricted by the problem of efficiency that is critical for real-world IR applications. Indeed, real-world IR applications are constrained by variables such as processing time and memory space. Identifying and extracting lexical associations and lexical relations is a computationally intensive process. In recent years new algorithms and new technologies have been proposed to introduce lexical cohesion techniques in large scale applications, thus avoiding previous intractable implementations. Previous workshops on lexical cohesion have mainly focused on the unconstrained extraction process. In this workshop, we would like to focus on the comparison of different factors that can influence the scalability of the treatment of lexical cohesion in real-world applications, namely data structures, algorithms, parallel and distributed computing or grid computing. We would also be interested in new methodologies for lexical cohesion that may easily scale to real-world applications based on complexity measurements.

(2) Evaluation of the benefits of Lexical Cohesion in IR applications

Contiguous lexical associations have often been used in experimental IR systems. Different techniques have been studied for this purpose: (a) statistical methods based on co-occurrence statistics or ngram language modeling techniques (b) hybrid techniques based on simple statistics and shallow linguistic techniques such as part-of-speech tagging and noun-phrase chunking and (c) knowledge-based techniques. However, the importance of the contribution of phrase matching has not been systematically quantified. Moreover, the evaluation of such techniques is difficult in IR applications, as the number of environment variables is very large and each system combines a variety of indexing and matching techniques. Therefore, a more focused and systematic approach towards analyzing the uses of lexical associations in IR and their evaluation is needed. This workshop will provide a framework for such analysis, and will present for discussion a number of challenging questions regarding the use of lexical associations in text. In particular we will ask questions such as: How should multiword units be incorporated into IR models designed for single terms? What weighting models can be used for them? How should they be matched against their lexical-syntactic variants in text? How should we handle non-contiguous lexical associations? How can we avoid over-weighting a phrase occurrence in a document matching more than one phrase in the query? These are only few questions of a huge field of research full of unsolved problems.
In contrast with contiguous lexical units, relations between non-contiguous lexical units are important building blocks of the text, forming its lexical cohesion. Indeed, the complete meaning of a word in text can only be realized when it is interpreted in combination with the surrounding words, forming lexical cohesive ties with them. These lexical relations have been used for a number of IR tasks, for example query expansion, passage retrieval, topic segmentation and text summarization. However, most of the techniques do not use deep semantic or discourse structure information in identifying such relations, instead relying on their statistical evidence i.e. their co-occurrence patterns. In fact, very little work has explored the use of NLP techniques such as lexical chaining or discourse analysis that make use of semantic and discourse structure within text to improve the performance of IR applications. One of the main objections to the use of such techniques has been the claim that they are more computationally demanding than statistical co-occurrence techniques. However, with the development of more efficient algorithms by the NLP community it will be interesting to further explore the use of such techniques in IR applications.
As a consequence, we would like to gather people who use lexical relations in different subfields of IR. Non-trivial questions are addressed here. What types of lexical relations prove useful for different IR tasks? What statistical models are most effective for the identification of lexical relations for different IR tasks? Can linguistic techniques for identifying lexical relations in text, such as lexical chaining or discourse analysis techniques be useful for any IR tasks? How can contiguous or non-contiguous lexical cohesive relations be identified in text? How can we reliably evaluate and compare these techniques?

## ACKNOWLEDGEMENTS

We wish to thank all the reviewers for their remarkable work. The quality of the Workshop mainly depends on their efforts for evaluation and the final program is the result of hard work from the Program Committee.

We wish to thank all the members of the Local Organizing Committee for their constant availability and competence.

We also want to thank all the authors for the quality of the submitted papers.

Finally, we want to thank our partners for their support.

## PARTNERS

Yahoo! Search Marketing (USA)

Yahoo! Research (USA)

University of Waterloo (Canada)

University of Beira Interior (Portugal)

# WORKSHOP PROGRAM

9:00 – 9:15 *Workshop opening and introduction*

**INVITED SPEAKER**

9:15 – 10:15 *Keynote speech: Phrases and Other Structure in Queries*, Bruce Croft (Center for Intelligent Information Retrieval, University of Massachusetts)

10:15-10:30 Question Session

10:30 – 11:00 Coffee break

**MORNING FULL PAPER SESSION**

11:00 – 11:20 *Comparing Query Formulation and Lexical Affinity Replacements in Passage Retrieval*, Egidio Terra (Pontifícia Universidade Católica do Rio Grande do Sul, Brazil) and Charles C. Clarke (University of Waterloo, Canada)

11:20 – 11:40 *A Study of Document Relevance and Lexical Cohesion between Query Terms*, Olga Vechtomova (University of Waterloo, Canada), Murat Karamuftuoglu (Bilkent University, Turkey) and Stephen Robertson (Microsoft Research Cambridge, England)

11:40 – 12:00 *Human Annotation of Lexical Chains*, Bill Hollingsworth (University of Cambridge Computer Laboratory, England) and Simone Teufel (University of Cambridge Computer Laboratory, England)

12:00 – 12:30 Panel Discussion 1

12:30 – 13:30 Lunch break

**AFTERNOON FULL PAPER SESSION**

13:30 – 13:50 *A Method to Calculate Probability and Expected Document Frequency of Discontinued Word Sequences*, Antoine Doucet (University of Helsinki, Finland) and Helena Ahonen-Myka (University of Helsinki, Finland)

13:50 – 14:10 *Unsupervised Topic Segmentation Based on Word Cooccurrence and Multi-Word Units for Text Summarization*, Gaël Dias (University of Beira Interior, Portugal) and Elsa Alves (New University of Lisbon, Portugal)

14:10 – 14:30 *Hypernyms Ontologies for Semantic Indexing*, Florian Seydoux (Swiss Federal Institute of Technology of Lausanne, Switzerland) and Jean-Cédric Chappelier (Swiss Federal Institute of Technology of Lausanne, Switzerland)

14:30 – 15:00 Panel Discussion 2

**POSTER/DEMO SESSION**

15:00 – 15:30 *Poster/Demo session*

*A Framework for Detecting Contextual Concepts in Texts*, Ágnes Sándor (Xerox Research Centre Europe, France)

*Automatic Hierarchical Clustering of Web Pages*, Ricardo Campos (University of Beira Interior, Portugal) and Gaël Dias (University of Beira Interior, Portugal)

*Evaluating Latent Semantic Vector Models with Synonym Tests and Document Retrieval*, Leif Grönqvist (Växjö University, Sweden)

15:30 – 15:45 – Coffee break

**SHORT PAPERS SESSION**

15:45 – 15:55 *An Evaluation of some Latent Semantic Vector Models Using a New Swedish Evaluation Set*, Leif Grönqvist (Växjö University, Sweden)

15:55 – 16:05 *Feature Representation for Effective Action-Item Detection*, Paul N. Bennett (Carnegie Mellon University, USA) and Jaime Carbonell (Carnegie Mellon University, USA)

16:05 – 16:15 *Predicting Extraction Performance Using Context Language Model*, Eugene Agichtein (Microsoft Research) and Silviu Cucerzan (Microsoft Research)

16:15 – 16:45 Panel Discussion 3

*Workshop closing*

## WORKSHOP CHAIRS

Rosie Jones (Yahoo! Inc, USA)

Olga Vechtomova (University of Waterloo, Canada)

Gaël Harry Dias (University of Beira Interior, Portugal)

## WORKSHOP PROGRAM COMMITTEE

Brigitte Grau - (LIMSI, France)

Bruce Croft - (University of Massachusetts, USA)

Charlie Clarke - (University of Waterloo, Canada)

Diana Inkpen - (University of Ottawa, Canada)

Dunja Mladenic - (Josef Stephan Institute, Slovenia)

Patrick Pantel - (University of Southern California, USA)

Egidio Terra - (Pontifícia Universidade Católica do Rio Grande do Sul, Brazil)

Gabriel Lopes - (New University of Lisbon, Portugal)

Graeme Hirst - (University of Toronto, Canada)

Gregory Grefenstette - (CEA, France)

Hal Daume - (University of Southern California, USA)

Helena Ahonen-Myka (University of Helsinki, Finland)

Murat Karamuftuoglu - (Bilkent University, Turkey)

Nicola Stokes - (University College Dublin, Ireland)

Peter Turney - (National Research Council Canada, Canada)

Rafael Muñoz - (University of Alicante, Spain)

# TABLE OF CONTENTS

# AUTHOR INDEX

| NAME | AFFILIATION | EMAIL |
|---|---|---|
| Ágnes Sándor | Xerox Research Centre Europe | agnes.sandor@xrce.xerox.com |
| Antoine Doucet | University of Helsinki | doucet@cs.helsinki.fi |
| Bill Hollingsworth | University of Cambridge | william.hollingsworth@cl.cam.ac.uk |
| Charles Clarke | University of Waterloo | claclark@plg.uwaterloo.ca |
| Egidio Terra | PUC/RS | egidio@inf.pucrs.br |
| Elsa Alves | New University of Lisbon | elsalves@zmail.pt |
| Eugene Agichtein | Microsoft Research | eugeneag@microsoft.com |
| Florian Seydoux | EPFL | forian.seydoux@epfl.ch |
| Gaël Dias | University of Beira Interior | ddg@di.ubi.pt |
| Helena Ahonen-Myka | University of Helsinki | hahonen@cs.helsinki.fi |
| Jaime Carbonell | Carnegie Mellon University | jgc+@cs.cmu.edu |
| J.C. Chappelier | EPFL | jean-cedric.chappelier@epfl.ch |
| Leif Grönqvist | Växjö University | leif.gronqvist@msi.vxu.se |
| Murat Karamuftuoglu | Bilkent University | hmk@bilkent.edu.tr |
| Olga Vechtomova | University of Waterloo | ovechtom@uwaterloo.ca |
| Paul N. Bennett | Carnegie Mellon University | pbennett+@cs.cmu.edu |
| Ricardo Campos | University of Beira Interior | ricardo.campos@ipt.pt |
| Silviu Cucerzan | Microsoft Research | silviu@microsoft.com |
| Simone Teufel | University of Cambridge | simone.teufel@cl.cam.ac.uk |
| Stephen Robertson | Microsoft Research | ser@microsoft.com |

## FULL PAPERS

# Comparing Query Formulation and Lexical Affinity Replacements in Passage Retrieval

Egidio Terra
Faculty of Informatics
PUC/RS
Porto Alegre, Brazil
egidio@inf.pucrs.br

Charles L.A. Clarke
School of Computer Science
University of Waterloo
Waterloo, Canada
claclark@plg.uwaterloo.ca

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

## General Terms

Performance, Experimentation

## Keywords

Question Answering, Passage Retrieval, Query Expansion

## ABSTRACT

We compare different query formulation strategies and expansion based on lexical affinities in the context of passage retrieval. Our method to expand the queries using lexical affinities replaces only the missing terms from the original query in candidate passages while scoring them. The replacement term's affinity with the missing term is used to weight the substitution, and the degree of affinity is computed using statistics generated from a terabyte corpus. The passages extracted using this replacement method and a set of passages extracted using different formulation strategies are evaluated using TREC's QA test set.

## 1. INTRODUCTION

In open domain question answering (QA), the process of finding answers for the questions normally takes one of these two approaches: 1) a subset of the collection is selected for further processing by the answer selection component; or 2) the whole collection is directly used by the answer selection component to find the answer. Only a certain, often small, number of documents will have one or more answers for a question, thus the first approach is often used in practice since the amount of data to be processed is reduced considerably, as it is the noise passed for posterior processing in the QA system. An important aspect in limiting the search on a sub-collection is that any imprecision in the process will prevent the system from finding the answer. The goal of sub-collection limitation is then reduce the amount of data to be further processed with the smallest error possible.

The task of sub-collection creation is accomplished by finding passages or documents that potentially contain the answer for a given question. In particular, in this work we focus on passage retrieval. In the context of QA, given a fixed retrieval model and collection, one must formulate queries that closely resemble the passages containing the answers to the questions. However, not always query terms occur in the relevant passages, either because in conjunction with other query terms it provides no or little extra information or due to the presence of an alternative term that shares a reasonably close meaning in relevant passages. This problem is normally addressed by the use of explicit query expansion or pseudo-relevance feedback. We approach this problem from a different perspective, by providing replacements for all the missing query terms using lexical affinity. The replacements can have semantic relationship with the missing terms or may be one of their morphological variants. We assume that pairs of lexical items, individual words or multiword sequences, that co-occur frequently, more often than expected by chance, have higher affinity to each other [17].

The method to use replacements by taking into account lexical affinity was introduced in Terra and Clarke [17]. We modified existing retrieval methods, one passage retrieval method and one document retrieval method, to use lexical affinity replacement. In our previous evaluation, document collection and queries were fixed in order to compare the original and the modified methods and the results showed significant improvements from the modified method to original one using the same queries. The current work extends our previous one by comparing the original passage retrieval method with expanded queries and the modified method using original queries.

We perform our experiments using the passage retrieval component of the University of Waterloo's MultiText QA system used in TREC 1999 to 2003. In particular, for the TREC 2003 QA test set [19], we generate different types of queries exploring lexical and syntactical aspects from the question, comparing results obtained using different strategies. Our results show that the replacements method based

*SIGIR Workshop on Methodologies and Evaluation of Lexical Cohesion Techniques in Real-world Applications — Beyond Bag of Words* '05 Salvador, Brazil
Copyright 200X ACM 1595930345/05/0008 ...$5.00.

on lexical affinities outperforms in precision common explicit query expansion and formulation strategies.

The remaining of this work is organized as follows: Section 2 describes some related work on query formulation and explicit expansion. The scoring function used in our passage retrieval is presented in Section 3. The lexical affinity replacement method is presented in Section 4 and the approach to measure affinity is presented in Section 5. Some common query formulation strategies used in our comparison are presented in Section 6. The following Section 7, presents the results and discussion.

## 2. RELATED WORK

Radev et al. [12] examine the query formulation process for natural language questions. In order to generate the query they selectively choose which words from the original question are included by iteratively examining these words. They also explicitly expand the original question terms using WordNet and also using words with distributional similarity computed *prior* to the query processing.

Tellex et al. [15] investigate many different passage retrieval techniques using TREC2002 QA test set. They compare a probabilistic and a boolean model for initial document retrieval and find that the boolean model delivers a good performance when compared to a specific and well known probabilistic retrieval model. A boolean model is also used to retrieve an initial set of documents in other QA systems [20], Yang *et al.* acknowledge that the the number of questions covered by documents retrieved using the boolean queries is not very high, however the number of passages retrieved is also smaller. To compensate the number of questions not covered they iteratively issue boolean queries, with a "successive constraint relaxation" approach. Saggion *et al.* [14] also investigates iterative relaxation approaches for conjunctive boolean queries, including expanding individual terms, the use of some structural components in the query, such as quotes, and also deleting terms from the conjunction.

Monz [11] investigates the document retrieval component of the FlexIR system in the context of QA. He proposes the use of stemming as a way to increase effectiveness, while pseudo-relevance feedback as applied to *ad hoc* tasks yields poor performance.

Bilotti *et al.* [1] study the effect of stemming and explicit expansion using inflectional variants and found that stemming produced a lower recall while explicit expansion resulted in higher recall. Their study focused on document retrieval in the context of QA.

Roberts and Gaizauskas study some strategies to retrieve passages, either by pre or post-processing a set of retrieved documents [13]. They also formalize the concept of "coverage", used earlier by Tellex et. [15] and also used by Collins-Thompson et al. [7] and in this work.

Clarke and Terra [4] compare document and passage retrieval. Their results show that document retrieval has a slightly better coverage than passage retrieval. However, the passage retrieval provides a smaller sub-collection, thus reducing the amount of noise for down-stream components of QA systems.

## 3. PASSAGE RETRIEVAL

In order to investigate different expansion strategies, we use the passage retrieval component from the MultiText system. It has been used successfully in question answering [5, 3, 9] and to extract terms for pseudo-relevance feedback [21]. It can be used to retrieve passages directly from the corpus with no need for documents pre-fetching. This passage retrieval method is used for the Query formulation strategies used in this paper.

From a query $Q = \{t_1, t_2, .., t_k\}$ let $T \subseteq Q$. Given an extent of text comprising all words in the interval $(u, v)$. The extent length is $l = v - u + 1$ and the probability of $P(t, l)$ that the extent contains one or more occurrences of $t$ is

$$\begin{aligned} P(t,l) &= 1 - (1 - p_t)^l \\ &= 1 - (1 - lp_t + O(p_t^2)) \\ &\approx lp_t. \end{aligned}$$

The probability that an extent $(u, v)$ contains all the terms from $T$ is then

$$\begin{aligned} P(T,l) &= \prod_{t_i in T} P(t,l) \\ &= \prod_{t \in T} lp_t \\ &= l^{|T|} \prod_{t \in T} p_t. \end{aligned}$$

The estimation of $p_t$ is given by the Maximum Likelihood Estimator (MLE) for $t$ in the collection

$$p_t = f_t/N$$

where $f_t$ is the collection frequency of $t$ and $N$ is the collection size in words. The score for an extent of length $l$ containing the terms in $T$ is the self-information of $P(T, l)$

$$\sum_{t_i \in T} \log(N/f_t) - |T| \log(l) \qquad (1)$$

The score is higher for short passages containing all terms in $T$ and there is a trade-off on the number of terms and size of the passage.

The original passage retrieval method was presented by Clarke *et al.* [5, 6] and it provides an efficient algorithm to retrieve all passages comprising 1 to $|Q|$ query terms. The running time to extract all extents contain the terms in $T$ is $O(|Q|\mathcal{J}_l log(N))$ where $|Q|$ is the total number of query terms, $\mathcal{J}_l$ is the number of extents containing $|T|$ query terms and $N$ is the corpus size in words. The algorithm is based on the positions of query terms, checking for close occurrence of other query terms and skipping repetitions of the same term. This algorithm benefits from the sorted position entries in the inverted list used to index the underlying collection and quickly locate terms.

| Query type | Coverage C@100 | Questions Covered | Documents Correct | # Documents | Precision P@100 | Precision P@20 |
|---|---|---|---|---|---|---|
| Okapi BM25 + AQUAINT | 0.903 | 327 | 5,368 | 36,200 | 0.1483 | 0.2381 |
| Okapi BM25 + Terabyte | 0.887 | 319 | 9,146 | 34,738 | 0.2633 | 0.3229 |

**Table 1: Effectiveness of the document retrieval in the initial set**

## 4. LEXICAL AFFINITIES REPLACEMENT METHOD

In explicit query expansion, new terms are added to the original query in order to prevent the loss of the related concept to missing original query term. For instance, in stemming, an occurrence of inflected form of a query term is to be accepted as its own. The replacement method presented here was introduced by Terra and Clarke [17]. It does not use any additional term if the original query terms are in the passage. If an original query term is missing then a new term is used. All the terms in the passage have their lexical affinity with the missing term computed and the term with the highest affinity is chosen to replace the missing term.

This modified passage retrieval only considers the whole query $Q$ since every extent has a representative for missing query terms and uses the degree of affinity between the missing query term and its representative to adjust the scoring function of the original method. We make a simplifying assumption that a sequence of terms containing the term $t$ also have a co-occurrence of $t$ and itself, i.e. $p_{t,t} = p_t$. If the term $t$ is not in the document a replacement term $r$ will be used. The weight of the replacement is the conditional probability $p_{t|r}$, which is calculated by estimating the maximum likelihood for $p_r$ from the corpus and estimating the joint probability by

$$p_{t,r} = f_{r,t}/N'$$

where $f_{r,t}$ is the joint frequency and $N'$ is the total number of pairs considered for the joint frequency in the corpus.

We take a winner-takes-it-all approach and choose the best $r$ in the extent,

$$\arg\max_{r \in (u,v)} \; p_{t|r}$$

Finally, the modified version of equation 1 using replacements is given by

$$\sum_{t_i \in Q} \log(N/f_{ti}) \cdot p_{ti|r} - |Q| \log(l) \tag{2}$$

We should note that since every extent has a representative for a query term, we can make arbitrary decisions on the extent size. This creates a trade-off between extent size and replacement quality. On the other hand, the fact that any extent can have a representative does not allow us to use the efficient algorithm existing for the original method. Instead of selecting the extent in sub-linear time complexity (log of the corpus size) as the original method, our approximation extracts the passages in linear time.

This method can be considered a query expansion technique but not a traditional explicit expansion where new terms are added prior to the query execution. It is neither a pseudo-relevance feedback since there is not an initial retrieval stage from which new terms are added to the query. The replacement of missing terms is done while executing the query, by finding replacements when scoring each passage.

## 5. COMPUTING LEXICAL AFFINITIES

To compute lexical affinity, we use the approach used by Terra and Clarke [17]. For lexical affinity the pointwise mutual information (PMI) is used to score relatedness between pairs of terms.

$$PMI(w_1, w_2) = log \frac{P_{w1,w2}}{P_{w1}P_{w2}} \tag{3}$$

The reason for choosing PMI is twofold [17]. First, it was demonstrated to be effective for language phenomena [18]. Second, it has a relationship with the inverse collection frequency —*icf* (or *idf* if document frequencies are used) . This relationship comes from the assumption that $P_{w,w} = P_w$, thus

$$\begin{aligned} PMI(w,w) &= log \ \tfrac{P_{w,w}}{P_w \cdot P_w} \\ &= log \tfrac{P_w}{P_w \cdot P_w} \\ &= -log \ P_w \\ &= icf_w \end{aligned} \tag{4}$$

In the case of the pair of words $w_1$ and $w_2$, the maximum value for the pointwise mutual information is bounded by $PMI(w_1, w_2) \leq icf_{w1}$ and $PMI(w_1, w_2) \leq icf_{w2}$. This can be easily verified since the PMI formula has maximum value when the joint probability is equal to the smallest marginal (if marginals are different). Therefore, we can use *icf* to normalize the PMI for a given word we want to replace

$$CondPMI(w_1, w_2) \;=\; \frac{log \ (P_{w1,w2})/(P_{w1} \cdot P_{w2})}{log \ (1)/(P_{w1})} \tag{5}$$

which is monotonic to

$$\frac{(P_{w1,w2})/(P_{w1} \cdot P_{w2})}{1/P_{w1}} \;=\; P_{w1|w2}$$

Thus, if we fix one word, in this case the missing query term, we can rank the affinity of remaining words of the passage. Since the goal is to find a replacement for one query term at each time, the denominator of the equation 5 is fixed for every replacement. We should note that there is a problem with the normalization in the conditional PMI. The problem occurs when PMI is negative, in which case we just set it to zero. Setting the negative value to zero could be avoided if we offset both *icf* and PMI by the minimal PMI value. We ignore negative PMI and set its value to zero, thus we use a self-regulated cut-off for the minimal value for a conditional PMI. We assume that any word in the document with a negative PMI with respect to the missing query term is not a good candidate for replacement.

For estimation of $P_{w1,w2}$ use the maximum likelihood :

$$P_{w1,w2} = f_{w1,w2}/N' \tag{6}$$

where the joint-frequency $f(w_1, w_2)$ is the number of the co-occurrences of $w_1$ and $w_2$ at distances ranging from four to 40 words apart. The lower cut-off prevents phrasal relationships (e.g. if the term "New" is a query term but "York" is not, then the latter is probably not a good replacement for the first). As most of the co-occurrences of "New" and "York" happen at distance one, then this cut-off will avoid this bias for pairs in the same phrase. Terra and Clarke [18] showed that windows of 32 words are a good setting for an upper bound on the distance. Our upper cut-off was arbitrarily set close to it (40). The value of $N'$ is the size of the window times the corpus size ($36 \cdot N$).

## 6. QUERY FORMULATION STRATEGIES

To compare the lexical affinities replacement method with query formulation we created a series of queries using some common strategies. For all of them we perform stop word exclusion.

- Bag-of-Words

  This is one of the most common forms to specify a query. In particular the vector space, probabilistic and many language models employ this strategy to generate queries. In the scope of QA, the query is comprised of the question terms and the order in which terms are specified is not important. The query terms are considered to be independent from each other.

- Stemming (Bag+Stem)

  A common strategy in information retrieval is to apply a stemmer to the query terms. The intuition is that using the word stem, and not the original form from the question, will help overcome mismatching vocabulary problems. These queries are comprised of the question terms with stemming. The collection index contains both the stem and original forms.

- Boolean conjunction (Bool)

  To ensure that all questions terms are present in the query, some QA systems use the boolean expressions [1, 14, 15, 20]. Our boolean queries are a conjunction of the question terms after stop words exclusion.

- Quotes

  In these queries we keep the original quotes when supplied in the question, e.g., WHAT COUNTRY IS KNOW AS THE "LAND OF THE RISING SUN?". For the purpose of retrieval, these quotations are treated as phrases and their constituent words are not used in query other than in the phrasal component. The remaining of the question words (not stop words) are used as in the bag-of-words approach.

- Quotes plus Noun Phrases (Phrases)

  To further investigate phrases in our passage retrieval method we explore noun phrases in the questions that are not part of quotes. The words in the questions were tagged using a standard POS tagger and adjacent pairs were concatenated if the sequence matches one of the following : 1) adjective followed by noun; 2) a non-proper noun followed by any noun; 3) foreign word followed by any noun; 4) any noun followed by a foreign word; 5) proper-noun followed by proper noun; and 5) numeral followed by any noun. Quotations were kept from the question. We must note that the POS tagger sometimes fails but that does not happen often, e.g. HOW/WRB DID/VBD JERRY/NNP GARCIA/NNP DIE/NNP ? where the main verb is tagged as a proper noun.

- Verb expansion (VE)

  The Waterloo's MultiText QA system of TREC expanded verbs as a way to improve effectiveness [6]. We use a probabilistic version of Earley's parser and a grammar created to handle QA questions. Each regular verb is stemmed and all irregular verbs are expanded. Bilotti *et al.* [1] expanded verbs, along with other expansions, in a "conjunction of disjunction" boolean queries (query terms are ANDed and expansion for individual terms are ORed).

- Verb expansion plus Quotes (VE+Quotes)

  These queries have both expanded verbs and quotes from the original questions. These components, along with some heuristics expansions such as expanding "U.S" to ("U.S" or "United States"), form queries used in our last TREC-QA participations. The heuristics expansions were removed in the experiments reported here, leaving only verb expansion and quotes as phrases.

## 7. RESULTS AND DISCUSSION

We evaluate the performance of the different query formulation strategies in passage retrieval using TREC 2003 QA test set. We focus on the 413 factoid questions from which we use the 362 that have available regular expression patterns, created from all submissions after human judgments were done. These patterns can be used in a script to perform automatic assessments, called lenient in TREC, as opposed to human judgments, called strict in TREC. Two target corpora were used. The official TREC corpus — AQUAINT — and a terabyte collection [4, 17, 3, 18].

The replacement weights were all extracted from the terabyte corpus, using MLE, as discussed in Section 5. However, it is interesting to note that using a window of larger size increases the chance of observing co-occurrence and, along with proper normalization, this can be viewed as a sort of smoothing [16].

To measure effectiveness of the passage retrieval with the different strategies we calculated the *coverage*, the percentage of the 362 questions where at least one retrieved passage containing the answer at 20 documents (C@20), the same metric used by Tellex et al. [15] and Roberts and

| Query type | Coverage C@20 | Questions Covered | Passages Correct | # Passages | Precision P@20 |
|---|---|---|---|---|---|
| Bag of words | 0.738 | 267 | 1269 | 7240 | 0.1753 |
| Bag+stem | 0.710 | 257 | 1251 | 7240 | 0.1728 |
| Boolean (bool) | 0.483 | 175 | 669 | 3787 | 0.1767 |
| Quotes | 0.735 | 266 | 1261 | 7240 | 0.1742 |
| Quotes+phrases | 0.669 | 242 | 1076 | 7032 | 0.1530 |
| Verb expansion (VE) | 0.746 | 270 | 1223 | 7240 | 0.1689 |
| VE +quotes | 0.749 | 271 | 1226 | 7240 | 0.1693 |
| Replacement | 0.749 | 271 | 1412 | 7240 | 0.1950 |

**Table 2: Passage Retrieval from top 100 okapi documents in the** AQUAINT **Corpus**

| Query type | Coverage C@20 | Questions Covered | Passages Correct | # Passages | Precision P@20 |
|---|---|---|---|---|---|
| Bag of words | 0.751 | 272 | 1894 | 7240 | 0.2616 |
| Bag+stem | 0.735 | 266 | 1835 | 7240 | 0.2535 |
| Boolean (bool) | 0.702 | 254 | 1474 | 5640 | 0.2613 |
| Quotes | 0.754 | 273 | 1891 | 7240 | 0.2612 |
| Quotes+phrases | 0.718 | 260 | 1681 | 7090 | 0.2371 |
| Verb expansion (VE) | 0.785 | 284 | 1877 | 7240 | 0.2593 |
| VE +quotes | 0.785 | 284 | 1899 | 7240 | 0.2623 |
| Replacements | 0.757 | 274 | 2033 | 7240 | 0.2808 |

**Table 3: Passage Retrieval from top 100 okapi documents in the Terabyte Corpus**

Gaizauskas [13]. We also used *precision* at 20 documents (P@20).

To restrict the passage selection we first retrieve a set of documents, using Okapi BM25, to which we apply all the query formulations and the replacement method. As such, the effectiveness of the passage selection is bounded by the original effectiveness of the document retrieval on the two collections, as presented in Table 1. The number of questions covered using the different collections is similar, a little bit higher in the AQUAINT collection but the precision is higher in the terabyte collection, as reported at 100 documents used in the initial retrieval. The same pattern occur at 20 documents.

The different strategies for explicit expansion and the lexical affinity replacement method were then applied to the 100 documents in each question to select the best 20 passages. For each query formulation a single passage is extracted from each document using equation 1. The same procedure is executed for the replacement method: one passage per document, passages scored by equation 2. While the matching span of a query is variable in the passage retrieval method used here, we extend all the returned passages to be 170 words long, roughly 1000 bytes-long and one quarter of the document average size.

The results of the passage selection in the AQUAINT corpus are shown in Table 2. Both verb expansion and the replacement methods cover the highest number of questions. In precision at 20 passages the replacement method is better: the difference with any other query formulation is statistically significant at 99% significance level using Wilcoxon signed rank test, as shown in Table 4. For the Terabyte corpus the results are shown in Table 3. Once again, the verb expansion strategies yield better coverage. The replacement method is worse than verb expansion in coverage but it is again the best in precision, with the differences between the replacement and other methods, with exception of the boolean queries, being statistically significant at 99% using Wilcoxon signed rank test. It is interesting to note that the trends in coverage are maintained across the two collections: poor performance of boolean queries and phrases. Also, stemming seems to harm more than help in precision, contradicting Monz [11] and in accordance with Bilotti *et al.* [1], although the metrics used here are different.

From all the strategies, the use of phrases has the worst outcome. Phrases can be decomposable or undecomposable [8]. The decomposable ones can be rewritten in different forms and, as consequence, be absent from some relevant passages, which we can call "mismatching decomposable phrases problem". This outcome can also be explained by fact that the scoring function used is designed to handle individual terms in order to address the bag-of-word approach and by assuming independence among query terms. The same is not observed when using quotes, since quotes are important as specified and tend not be rewritten, i.e. they are undecomposable phrases. Verb expansion consistently improves coverage but results in precision at 20 are mixed, mostly not statistically significant when compared to other query formulation strategies but the lexical affinity replacement method.

Boolean queries in conjunctive form are more restrictive: fewer passages are retrieved when these queries are used. This reduction helps final precision since every correct passage will have a greater impact. The coverage of boolean queries is smaller, a result of the reduced number of passage (i.e. less chance to cover questions). These findings suggest an explanation for the successful adoption of boolean

| p-values | Bag+stem | Boolean | Quotes | Phrases | VE | VE+Quotes | Replacement |
|---|---|---|---|---|---|---|---|
| Bag of words | 0.2096 | 0.1287 | 0.1003 | 2.76E-005 | 0.1632 | 0.1634 | 0.0003 |
| Bag+stem | - | 0.3234 | 0.3864 | 0.0148 | 0.9305 | 0.9258 | 1.29E-005 |
| Boolean (bool) | | - | 0.1568 | 0.8451 | 0.4246 | 0.4067 | 0.0014 |
| Quotes | | | - | 8.90E-005 | 0.3033 | 0.3109 | 7.51E-005 |
| Quotes+phrases | | | | - | 0.0086 | 0.0104 | 2.78E-009 |
| Verb expansion (VE) | | | | | - | 1.0000 | 1.69E-005 |
| VE +quotes | | | | | | - | 1.91E-005 |

**Table 4: p-values for p@20 pairwise Wilcoxon signed rank test -** AQUAINT

| p-values | Bag+stem | Boolean | Quotes | Phrases | VE | VE+Quotes | Replacement |
|---|---|---|---|---|---|---|---|
| Bag of words | 0.0283 | 0.6338 | 0.5062 | 0.0001 | 0.8277 | 0.9937 | 0.0005 |
| Bag+stem | - | 0.1358 | 0.0354 | 0.1455 | 0.1474 | 0.1078 | 1.43E-006 |
| Boolean (bool) | | - | 0.6657 | 0.0060 | 0.4872 | 0.5710 | 0.0540 |
| Quotes | | | - | 2.96E-005 | 0.8786 | 0.9336 | 0.0010 |
| Quotes+phrases | | | | - | 0.0037 | 0.0007 | 1.80E-008 |
| Verb expansion(VE) | | | | | - | 0.2839 | 0.0013 |
| VE +quotes | | | | | | - | 0.0031 |

**Table 5: p-values for p@20 pairwise Wilcoxon signed rank test - Terabyte**

queries, used in multiple iterations, in some QA systems [1, 10, 14, 15, 20]. Nonetheless, it is arguable that a QA system that can take advantage of the redundancy [5, 2] to find answers to questions would benefit from a large number of passages, in particular if the precision is at the same level or higher, as delivered by the lexical affinities replacement method. In fact, since the replacement method guarantees that exactly one representative for each query term is always present, we can view this method as performing a boolean conjunctive query of the original terms, either by themselves or through a proxy.

## 8. CONCLUSIONS

We compare different query formulation strategies and a lexical affinity replacement method in passage retrieval in the context of QA. We used lexical affinities to identify replacements for missing query terms while scoring passages. The replacement term weight is adjusted by its affinity with the missing one. This term replacement method produced consistent and significant improvements in precision in comparison with other query strategies. In terms of coverage, the replacement method has not outperformed query formulations, in particular verb expansion, which may suggest that a combination of verb expansion and the replacement method may produce both better coverage and precision. Our findings on phrases mirror the results of other works [8]. In particular, quotations are undecomposable phrases and must be used as they appear in the questions. Further investigation on decomposable phrases as suggested by Spark-Jones [8], with different scoring for phrases and individual terms may also improve effectiveness. In particular, these phrases can have their degree of lexical affinity taken into account in a modified scoring function.

## 9. REFERENCES

[1] M. W. Bilotti, B. Katz, and J. Lin. What works better for question answering: Stemming or morphological query expansion. In *SIGIR 2004 IR4QA: Information Retrieval for Question Answering Workshop*, 2004.

[2] E. Brill, J. Lin, M. Banko, S. Dumais, and A. Ng. Data-intensive Question Answering. In *Proceedings of 2001 Text REtrieval Conference*, Gaithersburg, MD, 2001.

[3] C. Clarke, G. Cormack, M. Laszlo, T. Lynam, and E. Terra. The impact of corpus size on question answering performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 369–370, Tampere, Finland, 2002.

[4] C. Clarke and E. L. Terra. Passage retrieval vs. document retrieval for factoid question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 427–428, Toronto, Canada, 2003.

[5] C. L. A. Clarke, G. V. Cormack, and T. R. Lynam. Exploiting redundancy in question answering. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 358–365. ACM Press, 2001.

[6] C. L. A. Clarke, G. V. Cormack, T. R. Lynam, and E. Terra. *Advances in Open Domain Question Answering*, chapter Question answering by passage selection. Kluwer Academic Publishers. To appear, 2004.

[7] K. Collins-Thompson, E. Terra, J. Callan, and C. L. A. Clarke. The effect of document retrieval quality on factoid question answering. In *ACM SIGIR Conference on Research and development in Information Retrieval*, 2004.

[8] K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 2. *Information Processing and Management*, 36(6):809–840, 2000.

[9] J. Lin, A. Fernandes, B. Katz, G. Marton, and S. Tellex. Extracting answers from the web using data annotation and knowledge mining techniques. In *The Eleventh Text REtrieval Conference (TREC 2002)*, Gaithersburg, MD, 2002.

[10] D. Moldovan, M. Paca, S. Harabagiu, and M. Surdeanu. Performance issues and error analysis in an open-domain question answering system. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 33–40, 2002.

[11] C. Monz. Document retrieval in the context of question answering. In F. Sebastiani, editor, *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR-03), Lecture Notes in Computer Science 2633*, pages 571–579. Springer-Verlag Heidelberg, 2003.

[12] D. R. Radev, H. Qi, Z. Zheng, S. Blair-Goldensohn, Z. Zhang, W. Fan, and J. Prager. Mining the web for answers to natural language questions. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 143–150. ACM Press, 2001.

[13] I. Roberts and R. Gaizauskas. Evaluating passage retrieval approaches for question answering. In *Proceedins of the 26th European Conference on Information Retrieval Research (ECIR 2004), Lecture Notes in Computer Science 2997*, pages 72–84. Springer-Verlag Heidelberg, 2004.

[14] H. Saggion, R. Gaizauskas, M. Hepple, I. Robrts, and M. Greenwood. Exploring the performance of boolean retrieval strategies for open domain question answering. In *SIGIR 2004 IR4QA: Information Retrieval for Question Answering Workshop*, 2004.

[15] S. Tellex, B. Katz, J. Lin, A. Fernandes, and G. Marton. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 41–47, 2003.

[16] E. Terra. *Lexical Affinities and Natural Language Applications*. Ph.D. thesis, School of Computer Science, University of Waterloo, October 2004.

[17] E. Terra and C. L. Clarke. Scoring missing terms in information retrieval tasks. In *13th ACM Conference on Information and Knowledge Management(CIKM)*, 2004.

[18] E. Terra and C. L. A. Clarke. Frequency estimates for statistical word similarity measures. In *Proceedings of Human Language Technology conference / North American chapter of the Association for Computational Linguistics annual meeting*, pages 244–251, Edmonton, Alberta, 2003.

[19] E. M. Voorhees. Overview of the TREC 2003 Question Answering track. In *In proceedings of 2003 Text REtrieval Conference*, pages 14–27, Gaithersburg, MD, 2003.

[20] H. Yang, T.-S. Chua, S. Wang, and C.-K. Koh. Structured use of external knowledge for event-based open domain question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 33–40, 2003.

[21] D. L. Yeung, C. L. A. Clarke, G. V. Cormack, T. R. Lynam, , and E. Terra. Task-specific query expansion (multitext experiments for trec 2003). In *2002 Text REtrieval Conference*, Gaithersburg, MD, 2003.

# A Study of Document Relevance and Lexical Cohesion between Query Terms

Olga Vechtomova
University of Waterloo
Waterloo, Canada
ovechtom@uwaterloo.ca

Murat Karamuftuoglu
Bilkent University
Ankara, Turkey
hmk@bilkent.edu.tr

Stephen Robertson
Microsoft Research Cambridge
Cambridge, UK
ser@microsoft.com

## ABSTRACT

Lexical cohesion is a property of text, achieved through lexical-semantic relations between words in text. Most information retrieval systems make use of lexical relations in text only to a limited extent. In this paper we empirically investigate whether the degree of lexical cohesion between the contexts of query terms' occurrences in a document is related to its relevance to the query. Experiments suggest significant differences between the lexical cohesion in relevant and non-relevant document sets exist.

## Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: search process

## General Terms

Experimentation, Theory.

## Keywords

Lexical cohesion, Information Retrieval, Relevance, Collocation.

## 1. INTRODUCTION

Word instances in text depend to various degrees on each other for the realisation of their meaning. For example, closed-class words (such as pronouns or prepositions) rely entirely on their surrounding words to realise their meaning, while open-class words, having meaning of their own, rely on other open-class words in the document to realise their contextual meaning. As we read, we process the meaning of each word we see in the context of the meanings of the preceding words in text, thus relying on the lexical-semantic relations between words to understand it. Lexical-semantic relations between open-class words form the *lexical cohesion* of text, which helps us perceive text as a continuous entity, rather than as a set of unrelated sentences.

Lexical cohesion is a major characteristic of natural language texts, which is achieved through semantic connectedness between words in text, and expresses continuity between the parts of text [1]. Lexical cohesion is not the same throughout the text. Segments of text which are about the same or similar subjects (topics) have higher lexical cohesion, i.e. share a larger number of semantically-related or repeating words, than unrelated segments.

In this paper we investigate the lexical cohesion property of texts, specifically, whether there is a relationship between relevance and lexical cohesion between query terms in documents. We also report preliminary experiments to investigate whether lexical cohesion property of texts can be useful in helping IR systems to predict the likelihood of a document's relevance. From a linguistic point of view, the main problem in ad-hoc IR can be seen as matching two imperfect textual representations of meaning: a query, representing user's information need, and a document, representing author's intention. Obviously, the fact that a document and a query have matching words does not mean that they have similar meanings. For example, query terms may occur in semantically unrelated parts of text, talking about different subjects. Intuitively, it seems plausible that if we take into consideration lexical-semantic relatedness of the contexts of different query terms in a document, we may have more evidence to predict the likelihood of the document's relevance to the query. This paper sets to empirically investigate this idea.

We hypothesise that relevant documents tend to have a higher level of lexical cohesion between different query terms' contexts than non-relevant documents. This hypothesis is based on the following premise: In a relevant document, all query terms are likely to be used in related contexts, which tend to share many semantically-related words. In a non-relevant document, query terms are less likely to occur in related contexts, and hence share fewer semantically-related words.

The goal of this study is to explore whether the level of lexical cohesion between different query terms in a document can be linked to the document's relevance property, and if so, whether it can be used to predict the document's relevance to the query. Initially we formulated a hypothesis to investigate whether there is a statistically significant relation between two document properties – its relevance to a query and lexical cohesion between the contexts of different query terms occurring in it.

*Hypothesis 1:* There exists statistically significant association between the level of lexical cohesion of the query terms' contexts in documents and relevance.

We conducted a series of experiments to test the above hypothesis. The results of the experiments show that there is a statistically significant association between the lexical cohesion of query terms in documents and their relevance to the query. This result suggested the next step of our investigation: evaluation of the usefulness of lexical cohesion in predicting documents' relevance. We hypothesised that re-ranking document sets retrieved in response to the user's query by the documents' lexical cohesion property can yield better performance results than a term-based document ranking technique:

*Hypothesis 2:* Ranking of a document set by lexical cohesion scores results in significant performance improvement over term-based document ranking techniques.

The rest of the paper is organised as follows: in the next section we discuss the concept of lexical cohesion and review related work in detail; in section 3 we present the experiments comparing the degrees of lexical cohesion between sample sets of relevant and non-relevant documents; in section 4 we describe experiments studying the use of lexical cohesion in document ranking; finally, section 5 concludes the paper and provides suggestions for future work.

## 2. LEXICAL COHESION IN TEXT

Halliday and Hasan introduced the concept of 'textual' or 'text-forming' property of the linguistic system, which they define as a "set of resources in a language whose semantic function is that of expressing relationship to the environment" [1, p.299]. They claim that it is the meaning realised through text-forming resources of the language that creates text, and distinguishes it from the unconnected sequences of sentences. They refer to text forming resources in language by the broad term of c*ohesion*. The continuity created by cohesion consists in "expressing at each stage in the discourse the points of contact with what has gone before" [1, p.299]. There are two major types of cohesion: (1) *grammatical*, realised through grammatical structures, and consisting of the cohesion categories of reference, substitution, ellipsis and conjunction; and (2) *lexical* cohesion, realised through lexis [1]. Halliday and Hasan distinguished two broad categories of lexical cohesion: *reiteration* and *collocation*. Reiteration, as defined in [1], refers to a broad range of relations between a lexical item and another word occurring before it in text, where the second lexical item can be an exact repetition of the first, a general word, its synonym or near-synonym or its superordinate. As for the second category – collocation, Halliday and Hasan understand that this is a relationship between lexical items that occur in the same environment, but they fail to formulate a more precise definition.

Later, some linguists narrowed down the meaning of collocation to refer only to restricted type of collocations, whose meaning cannot be completely derived from the meaning of their elements. For example Manning and Schütze [2] defined collocation as grammatically bound elements occurring in a certain order which are characterised by limited compositionality, i.e. the impossibility of deriving the meaning of the total from the meanings of its parts.

We recognise two major types of collocation:

1. Collocation due to lexical-grammatical or habitual restrictions. These restrictions limit the choice of words that can be used in the same grammatical structures with the word in question. Collocations of this type occur within short spans, i.e. within the bounds of a syntactic structure, such as a noun phrase, (e.g. “rancid butter”, “white coffee”, “mad cow disease”).
2. Collocation due to a typical occurrence of a word in a certain thematic environment: two words hold a certain lexical-semantic relation, i.e. their meanings are close semantically, therefore they tend to occur in the same topics in texts. Beeferman et al. experimentally determined that long-span collocation effects can extend in text up to 300 words [3]. Vechtomova et al. report examples of long span collocates identified using the Z-score such as “environment–pollution”, “gene–protein” [4].

Hoey [5] gave a different classification of lexical cohesive relationships under a broad heading of *repetition*: (1) simple lexical repetition, (2) Complex lexical repetition, (3) Simple partial paraphrase, (4) Simple mutual paraphrase, (5) Complex paraphrase, (6) Superordinate, hyponymic and co-reference repetition.

In this work we investigate the relationship between relevance and the level of lexical cohesion among query terms based on the simple lexical repetition of their long span collocates.

### 2.1 LEXICAL LINKS AND CHAINS

A single instance of a lexical cohesive relationship between two words is usually referred to as a *lexical link* [5, 6, 7, 8]. Lexical cohesion in text is normally realised through sequences of linked words – *lexical chains*. The term *'chain'* was first introduced by Halliday and Hasan [1] to denote a relation where an element refers to an earlier element, which in turn refers to an earlier element and so on.

Morris and Hirst [6] define lexical chains as sequences of related words, which have distance relations between them. One of the prerequisites for the linked words to be considered units of a chain is that they should co-occur within a certain span. Hoey [5] suggested using only information derivable from text to locate links in text, Morris and Hirst used Roget's thesaurus in identifying lexical chains. Morris and Hirst’s algorithm was later implemented for various tasks: IR [9], text segmentation [10] and summarisation [11].

### 2.2 LEXICAL BONDS

Hoey [5] pointed that text cohesion is built not only of links between words, but also of semantic relationships between sentences. He argued that if sentences are not related as whole units, even though there are some lexically linked words found in them, they are no more than a disintegrated sequence of sentences sharing a lexical context. He emphasised that it is important to interpret cohesion by taking into account the sentences where it is realised. For example, two sentences in text can enter the relation, where the second one exemplifies the statement expressed in the previous sentence. Sentences do not have to be adjacent to be related, and lexical cohesive relation can connect several sentences.

A cohesive relation between sentences was termed by Hoey as a *lexical bond*. He defines a bond between sentences as a sufficient number of lexical links between them. The number of lexical links the sentences must have to be bonded is a relative parameter, according to Hoey, depending indirectly on the relative length and the lexical density of the sentences. Hoey argues that an empirical method for estimating a minimum number of links the sentences need to have to form a bond must rely on the proportion of sentence pairs that form bonds in text. If the proportion of sentences linked by any given number of links is too high, then it is important to increase the cut-off point, until the degree of connection is not above average. In practice two or three links are considered sufficient to constitute a bond between a pair of sentences.

It is notable that in Hoey's experiments, only 20% of bonded sentences were adjacent pairs. Analysing non-adjacent sentences, Hoey made and proved two claims about the meaning of bonds. The first claim is that bonds between sentences are indicators of semantic relatedness between sentences, which is more than the sum of relations between linked words. The second claim is that a large number of bonded sentences are intelligible without recourse to the rest of the text, as they are coherent and can be interpreted on their own [5].

## 3. COMPARISON OF RELEVANT AND NON-RELEVANT SETS BY THE LEVEL OF LEXICAL COHESION

### 3.1 EXPERIMENTAL DESIGN

Our method of estimating the level of lexical cohesion between query terms was inspired by Hoey's method [5] of identifying lexical bonds between sentences. There is, however, a substantial difference between the aims of these two methods. Sentence bonds analysis is aimed at finding semantically related sentences. Our method is aimed at predicting whether query terms occurring in a document are semantically related, and measuring the level of such relatedness.

In both methods the similarity of local context environments is compared: in our method – fixed-size windows around query terms; in Hoey's method – sentences. Hoey's method identifies semantic relatedness between sentences in a text, whereas the objective of our method is to determine the semantic similarity of the contextual environments, i.e., collocates, of different query terms in a document.

To determine semantic similarity of the contextual environments of query terms we combine all windows for one query term, building a merged window for it. Each query term's merged window represents its contextual environment in the document. We then determine the level of lexical cohesion between the contextual environments of query terms**.** We experimented with two methods to determine the level of lexical cohesion between different query terms: (a) How many lexical links connect them, and (b) How many types they have in common. Each document is then assigned a *lexical cohesion score* (*LCS*), based on the level of lexical cohesion between different query terms' contexts.

In more detail, the algorithm for building merged windows for a query term is as follows: Fixed-size windows are identified around every instance of a query term in a document. A window is defined as *n* number of stemmed[1] non-stopwords to the left and right of the query term. We refer to all stemmed non-stopwords extracted from each window surrounding a query term as its *collocates*. In our experiments different window sizes were tested: 10, 20 and 40. These window sizes are large enough to capture collocates related topically, rather than syntactically.

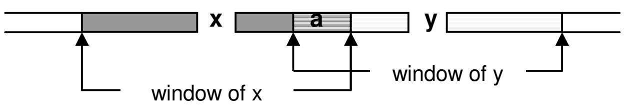


**Figure 1: Overlapping windows around query terms x and y.**

In this windowing technique we can encounter a situation where windows of two different query terms overlap. In such a case, we run into the following problem: let us assume that query terms *x* and *y* have overlapping windows and, hence, both are considered to collocate with term *a* (see Figure 1). We could simply add this instance of the term *a* into the merged windows of both *x* and *y*. However, when we compare these two merged windows, we would count this instance of *a* as a common term between them. This would be wrong, for we refer to the same instance of *a*, as opposed to a genuine lexical link by two different instances of *a*. Our solution to this problem is to attribute each instance of a word in an overlapping window to only one query term (node) – the nearest one.

[1] We used the weak stemming function in Okapi.

#### 3.1.1 ESTIMATING SIMILARITY BETWEEN THE QUERY TERMS' CONTEXTS

After merged windows for all query terms in a document are built, the next step is to estimate their similarity by the collocates they have in common. We do pairwise comparisons between query terms, using the following two methods:

*Method 1:* Comparison by the number of lexical links they have.

*Method 2:* Comparison by the number of types they have in common.

##### 3.1.1.1 METHOD 1

The first method takes into account how many instances of common collocates each query term has. In Figure 2, the first column contains collocates in the merged window of the query term *x*, the second column contains collocates in the merged window of the query term *y*. The lines between instances of the common collocates in the figure represent lexical links.

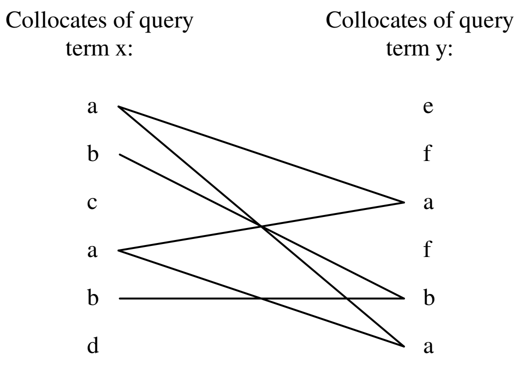


**Figure 2: Links between instances of common collocates in merged windows of query terms x and y.**

In this example there are altogether 6 links. If there are more than 2 query terms in a document, a comparison of each pair is done. The number of links are recorded for each pair, and summed up to find the total number of links in the document.

In our experiments, we only counted links formed by simple lexical repetition. We recognise that semantic similarity between contexts of terms might be more accurately estimated if we take into account other lexical-semantic relations between words, for example hyponymy, hypernymy, synonymy, etc. This would, require recourse to dictionaries and thesauri, such as WordNet. We plan to extend our work using such resources in the future.

A document's lexical cohesion score, calculated using method 1, will be referred to as $LCS_{links}$. To compare the scores across documents we need to normalise the total number of links in a document by the total size of all merged windows in a document. The normalised $LCS_{links}$ score is:

$$LCS_{links} = \frac{L}{V} \quad (1)$$

*where:*

$L$ – the total number of lexical links in a document;

$V$ – the size (in words) of all merged windows in a document, excluding stopwords.

#### *3.1.1.2 METHOD 2*

In method 2 no account is taken of the number of common collocate *instances* each query term co-occurs with. Instead only the number of common *types* between each pair of merged windows is counted.

Comparison of merged windows in Figure 2 will return 2 types that they have in common: *a* and *b*. Again, if there are more than 2 query terms, a pairwise comparison is done. For each document we record the number of types common between each pair of merged windows, and sum them up.

A document's lexical cohesion score estimated using this method is $LCS_{types}$, and is calculated by normalising the total number of common types by the total number of types in the merged windows in a document:

$$LCS_{types} = \frac{T}{U} \quad (2)$$

*Where:*

$T$ – the total number of common types in a document;

$U$ – the total number of types in all merged windows in a document.

### 3.2 CONSTRUCTION OF SETS OF RELEVANT AND NON-RELEVANT DOCUMENTS

To test the hypothesis that lexical cohesion between query terms in a document is related to a document's property of relevance to the query, we calculated average lexical cohesion scores for sets of relevant and non-relevant documents.

We conducted our experiments on two datasets:

1) A subset of the TREC ad-hoc track dataset: FT 96[2] database, containing 210,158 Financial Times news articles from 1991 to 1994, and 50 ad-hoc topics (251 – 300) from TREC-5. We will refer to this dataset in this paper as "FT".
2) The HARD track dataset of TREC-12: 652,710 documents from 8 newswire newswire corpora (New York Times, Associated Press Worldstream and Xinghua English, among others), and 50 topics (401-450). This dataset will be referred to as "HARD".

Short queries were created from all non-stopword terms in the 'Title' fields of TREC topics. Such requests are similar to the queries that are frequently submitted by average users in practice. The queries were run in the Okapi IR system using BM25 document ranking function to retrieve top N documents for analysis. BM25 is based on the Robertson & Spärck-Jones probabilistic model of retrieval [12]. The sets of relevant and nonrelevant documents are then built using TREC relevance judgements for the top N documents retrieved.

We need to ascertain that the difference between the average lexical cohesion scores in the relevant and non-relevant document sets is not affected by the difference between the average BM25 document matching scores. To achieve this we need to build the relevant and non-relevant sets, which have similar mean and standard deviation of BM25 scores for each topic. This is achieved as follows: first all documents among the top *N* BM25-ranked documents are marked as relevant and non-relevant using TREC relevance judgements. Then each time a relevant document is found it is added to the relevant set and the nearest scoring non-relevant document is added to the non-relevant set. After the sets are composed, the mean and standard deviation of BM25 document matching scores are calculated for each topic in the relevant and non-relevant sets. If there is a significant difference between the mean and standard deviation in the two sets for a particular topic, then the sets are edited by changing some documents until the difference is minimal. We will refer to the relevant and non-relevant document sets constructed using this technique as *aligned sets*.

We created two pairs of aligned sets for FT and HARD corpora: using the top 100 BM25-ranked documents and using the top 1000 BM25-ranked documents. The sets and their sizes are presented in Table 1.

**Table 1: Statistics of the aligned relevant and nonrelevant sets.**

| Data set | FT | | HARD | |
|---|---|---|---|---|
| | Relevant | Non-relevant | Relevant | Non-relevant |
| Top100 | | | | |
| Number of documents | 176 | 176 | 600 | 600 |
| Mean BM25 document score | 13.350 | 13.230 | 13.939 | 13.674 |
| Stdev BM25 document score | 2.200 | 1.905 | 4.254 | 3.864 |
| Top1000 | | | | |
| Number of documents | 268 | 268 | 1897 | 1897 |
| Mean BM25 document score | 11.515 | 11.472 | 11.306 | 11.219 |
| Stdev BM25 document score | 2.502 | 2.375 | 3.519 | 3.311 |

Comparison between the corresponding relevant and non-relevant sets was done by average lexical cohesion score, which was calculated as:

$$Average\ LCS = \frac{\sum_{i=1}^{S} LCS_i}{S} \quad (3)$$

*where:*

$LCS_i$ – lexical cohesion score of *i*th document in the set, calculated using either formula (1), or (2) above.

$S$ – number of documents in the set.

[2] TREC research collection, volume 4.

## 3.3 ANALYSIS OF RESULTS

Comparisons of pairs of relevant and non-relevant aligned sets derived from 100 and 1000 BM25-ranked documents showed large differences between the sets on some measures (Table 2). In particular, average Lexical Cohesion Scores of the relevant and non-relevant documents selected from the top 1000 BM25-ranked document sets, calculated using the Links method ($LCS_{links}$) have statistically significant differences (Wilcoxon[3] test at 0.05 significance level). Average $LCS_{types}$ are also significantly different in most of the experiments.

**Table 2: Difference between the aligned relevant and non-relevant sets (FT dataset)**

| Method | Window | Rel | Nonrel | Difference (%) | Wilcoxon P(2-tail) | Significant |
|---|---|---|---|---|---|---|
| FT, Top 1000 | | | | | | |
| Links | 10 | 0.097 | 0.076 | 28.795 | 0.025 | Y |
| Links | 20 | 0.151 | 0.119 | 26.727 | 0.002 | Y |
| Links | 40 | 0.197 | 0.165 | 19.868 | 0.008 | Y |
| Types | 10 | 0.056 | 0.043 | 30.454 | 0.009 | Y |
| Types | 20 | 0.071 | 0.057 | 24.733 | 0.001 | Y |
| Types | 40 | 0.082 | 0.071 | 14.333 | 0.031 | Y |
| FT, Top 100 | | | | | | |
| Links | 10 | 0.091 | 0.069 | 31.562 | 0.061 | N |
| Links | 20 | 0.144 | 0.109 | 32.703 | 0.001 | Y |
| Links | 40 | 0.187 | 0.146 | 28.016 | 0.001 | Y |
| Types | 10 | 0.048 | 0.036 | 33.920 | 0.024 | Y |
| Types | 20 | 0.063 | 0.047 | 32.928 | 0.001 | Y |
| Types | 40 | 0.074 | 0.061 | 21.010 | 0.005 | Y |
| HARD, Top 1000 | | | | | | |
| Links | 10 | 0.090 | 0.074 | 21.39 | 0.000 | Y |
| Links | 20 | 0.145 | 0.122 | 15.76 | 0.000 | Y |
| Links | 40 | 0.195 | 0.166 | 17.49 | 0.000 | Y |
| Types | 10 | 0.053 | 0.050 | 7.17 | 0.003 | Y |
| Types | 20 | 0.071 | 0.069 | 2.65 | 0.167 | N |
| Types | 40 | 0.086 | 0.084 | 1.36 | 0.387 | N |
| HARD, Top 100 | | | | | | |
| Links | 10 | 0.102 | 0.089 | 15.66 | 0.032 | Y |
| Links | 20 | 0.167 | 0.143 | 16.68 | 0.003 | Y |
| Links | 40 | 0.218 | 0.188 | 16.24 | 0.000 | Y |
| Types | 10 | 0.059 | 0.054 | 9.01 | 0.087 | N |
| Types | 20 | 0.080 | 0.075 | 5.91 | 0.175 | N |
| Types | 40 | 0.095 | 0.091 | 4.32 | 0.105 | N |

The first method of comparison by counting the number of links between merged windows appears to be somewhat better than the second method of comparison by types. This suggests that the density of repetition of common collocates in the contextual environments of query terms offers some extra relevance discriminating information.

To investigate other possible differences between the documents in the relevant and non-relevant sets, we have calculated various document statistics (Table 3). In both FT and HARD document collections the relevant documents, on average are longer, have more query term occurrences, and consequently have more collocates per query term. The latter finding is interesting, given that we selected relevant and non-relevant document pairs with the similar BM25 scores. However, BM25 scores do not depend on query term occurrences only. A number of other factors affect BM25 score: a) document length; b) *idf* weights of the query terms; c) non-linear within-document term frequency function which progressively reduces the contribution made by the repeating occurrences of a query term to the document score, on the assumption of verbosity[4].

**Table 3: Averaged document characteristics (FT and HARD document sets created from top1000 documents)**

| | Rel | Nonrel | Difference (%) | t-test P |
|---|---|---|---|---|
| FT (Top 1000) | | | | |
| Ave. number of collocate tokens per query term | 95.900 | 71.331 | 34.444 | 0.000 |
| Ave. query term instances | 11.704 | 8.719 | 34.230 | 0.000 |
| Ave. document length | 332.012 | 224.658 | 47.786 | 0.000 |
| Ave. distance between query terms | 19.444 | 14.976 | 29.832 | 0.027 |
| Ave shortest distance between query terms | 6.533 | 4.617 | 41.498 | 0.085 |
| HARD (Top 1000) | | | | |
| Ave. number of collocate tokens per query term | 86.848 | 66.561 | 30.479 | 0.000 |
| Ave. query term instances | 11.297 | 8.693 | 29.962 | 0.000 |
| Ave. document length | 282.740 | 220.419 | 28.274 | 0.000 |
| Ave. distance between query terms | 18.077 | 17.705 | 2.099 | 0.633 |
| Ave shortest distance between query terms | 6.164 | 7.113 | 15.389 | 0.091 |

An interesting, though somewhat counter-intuitive, finding is the *average distance* between query term instances, which is significantly longer in relevant documents. To calculate the average distance between query terms, we take all possible pairs of different query term instances, and for each pair find the shortest matching strings, using the cgrep program [13]. The shortest matching string is a stretch of text between two different query terms (say, *x* and *y*) that do not contain any other query term instance of the same type as either of the query terms (i.e., *x* or *y*). Once the shortest matching strings are extracted for each pair of query terms, the distances between them are calculated (as the number of non-stopwords) and averaged over the total number of pairs. The closer the query terms occur to each other, the more their windows overlap, and hence the fewer collocates they have. In the nonrelevant documents query terms occur on average closer to each other (Table 3), which may contribute to the fact that they have fewer collocates. Longer distances between query terms in the relevant documents may be explained by the higher document length values in the relevant set, compared to the nonrelevant set.

Another statistic, *average shortest distance* between query terms, is calculated by finding the shortest matching string for each distinct query term combination. In this case, only one value, the shortest distance between each distinct pair, is returned. The shortest distances of all distinct pairs are then summed and

[3] The distribution of the data is non-Gaussian.

[4] The term frequency effect can be adjusted in BM25 by means of the tuning constant $k_1$. In our experiments we used $k_1$=1.2, which showed optimal performance on TREC data [12]. This chosen value means that repeating occurrences of query terms contribute progressively less to the document score.

averaged. As Table 3 shows, this value is larger in the relevant documents than in the nonrelevant in the FT corpus, and smaller in the HARD corpus. The differences are not statistically significant, though.

The above analysis clearly shows that relevant documents are longer and have more query term occurrences. So, could any of these factors possibly be the reason for the higher average Lexical Cohesion Scores in relevant documents? As instances of the original query terms can be collocates of each other, and form links between the collocational contexts of each other or other query terms, we need to find out what is the number of link-forming collocates (i.e. those which form links with collocates of other query terms), which are not query terms themselves. The following hypothesis was formulated to investigate this possibility:

*Hypothesis 1.1:* Collocational environments of different query terms are more cohesive in the relevant documents than in the nonrelevant, and this difference is not due to the larger number of query terms.

To investigate the above hypothesis, we counted in each document the total number of link-forming collocate instances (excluding the query terms) and normalised this count by the total number of all collocates in the windows of all query term instances. We refer to the normalised link-forming collocate count per document as *link_cols*, and the total number of collocates of query terms in the document as *total_cols*. The data (Table 4) shows that there exist large differences between the relevant and nonrelevant sets. Seven out of twelve experiments demonstrate statistically significant differences. This indicates that the contexts of different query terms in the relevant documents on average are more cohesive than in the non-relevant documents, and that this difference is not due to the higher number of query term instances. The fact that we normalise the count by the total number of collocates of query terms in the document eliminates the possibility of larger collocate numbers affecting this difference.

**Table 4: Average number of link-forming collocates (excluding original query terms), normalised by the total number of collocates of query terms in the document**

| Window | Rel | Nonrel | Difference (%) | Wilcoxon P(2-tail) | Significant? |
|---|---|---|---|---|---|
| FT, Top1000 | | | | | |
| 10 | 0.071 | 0.065 | 9.607 | 0.000 | Y |
| 20 | 0.100 | 0.095 | 5.849 | 0.002 | Y |
| 40 | 0.123 | 0.118 | 4.636 | 0.010 | Y |
| FT, Top100 | | | | | |
| 10 | 0.070 | 0.065 | 7.630 | 0.067 | N |
| 20 | 0.101 | 0.096 | 5.019 | 0.300 | N |
| 40 | 0.123 | 0.115 | 6.963 | 0.045 | N |
| HARD, Top1000 | | | | | |
| 10 | 0.063 | 0.055 | 14.408 | 0.066 | N |
| 20 | 0.085 | 0.071 | 19.567 | 0.009 | Y |
| 40 | 0.103 | 0.090 | 14.465 | 0.013 | Y |
| HARD, Top100 | | | | | |
| 10 | 0.063 | 0.053 | 18.441 | 0.083 | N |
| 20 | 0.086 | 0.067 | 27.904 | 0.004 | Y |
| 40 | 0.105 | 0.086 | 21.992 | 0.002 | Y |

To find out whether the normalised link-forming collocate count can be statistically predicted by the number of query term instances we conducted linear regression analysis on the data of one of the experiments (HARD, top 1000 document dataset, window size 10), with the normalised link-forming collocate count per document (*link_cols*) as the dependent variable, and the number of query term instances in the document (*qterms*) as the independent variable. The R Square for the relevant document set was found to be 0.182, and for the nonrelevant document set, R Square was 0.122. Rather low R Square values support the Hypothesis 3 stated above. The result of the analysis indicates that the linear model using *qterms* can predict only about 18% of the *link_cols* values.

## 4. RE-RANKING OF DOCUMENT SETS BY LEXICAL COHESION SCORES

### 4.1 EXPERIMENTAL DESIGN

Statistically significant differences in the average lexical cohesion scores between relevant and non-relevant sets, discovered in the previous experiments, prompted us to evaluate LCS as a document ranking function.

It was decided to conduct experiments on re-ranking the set of top 1000 BM25-ranked documents by their LCS scores. Document sets were formed by using weighted search with the queries for $45^5$ topics of the HARD corpus. The queries were created from all non-stopword terms in the 'Title' fields of the TREC topics. Okapi IR system with the search function set to BM25 (without relevance information) was used for searching. Tuning constant $k_1$ (controlling the effect of within-document term frequency) was set to 1.2 and $b$ (controlling document length normalisation) was set to 0.75 [12].

BM25 function outputs each document in the ranked set with its document matching score (MS). We decided to test re-ranking with a simple linear combination function (*COMB-LCS*) of MS and LCS. Tuning constant $x$ was introduced into the function to

$$COMB\text{-}LCS = MS + x * LCS \qquad (4)$$

regulate the effect of LCS:

The following values of $x$ were tried: 0.25, 0.5, 0.75, 1, 1.5, 3, 4, 5, 6, 7, 8, 10 and 30.

We conducted experiments with both types of lexical cohesion scores:

$LCS_{links}$ – calculated using method 1 of comparing query terms' collocation environments by the number of links they have;

$LCS_{types}$ – calculated using method 2 of comparing query terms' collocation environments by the number of types they have in common.

The window sizes tested were 40, 20 and 10.

5 Five of the 50 topics had no relevant documents and were excluded from the official HARD 2004 evaluation [15].

## 4.2 ANALYSIS OF RESULTS

Precision results of re-ranking with the combined linear function of MS and LCS with different values for the tuning constant *x* are presented in Table 5.

**Table 5: Results of re-ranking BM25 document sets by *COMB-LCS* (HARD corpus)**

| Runs with different *x* values | Window size 40 | | Window size 20 | | Window size 10 | |
|---|---|---|---|---|---|---|
| | AveP | P@10 | AveP | P@10 | AveP | P@10 |
| BM25 | 0.2196 | 0.3089 | | | | |
| Method 1 (links) | | | | | | |
| 0.25 | 0.2201 | 0.3156 | 0.2199 | 0.3178 | 0.2198 | 0.3156 |
| 0.5 | 0.2208 | 0.3200 | 0.2207 | 0.3200 | 0.2200 | 0.3178 |
| 0.75 | 0.2213 | 0.3222 | 0.2217 | 0.3156 | 0.2202 | 0.3178 |
| 1 | 0.2213 | 0.3200 | 0.2217 | 0.3133 | 0.2209 | 0.3156 |
| 1.5 | 0.2217 | 0.3244 | 0.2223 | 0.3156 | 0.2214 | 0.3200 |
| 3 | 0.2242 | 0.3267 | 0.2241 | 0.3200 | 0.2230 | 0.3222 |
| 4 | 0.2240 | 0.3311 | 0.2268 | 0.3222 | 0.2230 | 0.3133 |
| 5 | 0.2205 | 0.3400 | 0.2322 | 0.3333 | 0.2231 | 0.3244 |
| 6 | 0.2227 | 0.3444 | 0.2316 | 0.3378 | 0.2230 | 0.3267 |
| 7 | 0.2227 | 0.3489 | 0.2314 | 0.3356 | 0.2258 | 0.3289 |
| 8 | 0.2265 | 0.3556 | 0.2311 | 0.3422 | 0.2258 | 0.3356 |
| 10 | 0.2217 | 0.3556 | 0.2303 | 0.3356 | 0.2254 | 0.3333 |
| 30 | 0.1964 | 0.3200 | 0.2097 | 0.3244 | 0.2179 | 0.3156 |
| Method 2 (types) | | | | | | |
| 0.25 | 0.2196 | 0.3089 | 0.2196 | 0.3067 | 0.2196 | 0.3111 |
| 0.5 | 0.2197 | 0.3133 | 0.2197 | 0.3111 | 0.2196 | 0.3133 |
| 0.75 | 0.2199 | 0.3133 | 0.2197 | 0.3111 | 0.2197 | 0.3111 |
| 1 | 0.2200 | 0.3133 | 0.2198 | 0.3156 | 0.2197 | 0.3133 |
| 1.5 | 0.2201 | 0.3133 | 0.2200 | 0.3178 | 0.2199 | 0.3178 |
| 3 | 0.2200 | 0.3044 | 0.2203 | 0.3156 | 0.2209 | 0.3200 |
| 4 | 0.2199 | 0.3044 | 0.2203 | 0.3156 | 0.2210 | 0.3200 |
| 5 | 0.2200 | 0.2978 | 0.2205 | 0.3133 | 0.2216 | 0.3244 |
| 6 | 0.2199 | 0.3022 | 0.2203 | 0.3133 | 0.2216 | 0.3200 |
| 7 | 0.2172 | 0.3022 | 0.2207 | 0.3133 | 0.2216 | 0.3222 |
| 8 | 0.2168 | 0.3022 | 0.2217 | 0.3111 | 0.2213 | 0.3244 |
| 10 | 0.2161 | 0.3044 | 0.2215 | 0.3111 | 0.2211 | 0.3244 |
| 30 | 0.2030 | 0.3178 | 0.2133 | 0.3200 | 0.2142 | 0.3089 |

The results show that there is a significant increase in precision at the cut-off point of 10 documents (P@10) when LCS scores are combined with the MS as given by equation 4 above, with $x$=8 and window size of 40. The precision @10 for BM25 and LCS scores are 0.3089 and 0.3556, respectively. The 15% increase is statistically significant (Wilcoxon test at P=0.001). Thirteen topics have higher precision and none – lower. Average precision (AveP) also increases, although by a smaller amount when documents are re-ranked with equation 4. The highest gain in average precision (5.7%) is achieved when *x* is 5 and window size is 20. This result is not, however, statistically significant. It is also worth mentioning that 5 out of 45 topics used in evaluation have only one query term in the topic title, and our method can only be applied to queries with two or more query terms.

A number of factors need to be considered in the context of the re-ranking experiments: 65.39% of documents have LCS score of zero. This is mainly because a very large proportion of documents (52.64%) only have one distinct query term. Also, we only compared lexical environments of query terms through the repetition of their collocates. It is likely that only a certain proportion of lexical links is determined in this way. For a fuller analysis, other types of lexical-semantic relations should be investigated. The above factors may have a significant impact on the results of re-ranking, and we expect to have better results if the above points are successfully addressed in future studies. It should also be noted that the combined function used in re-ranking is rather simple and alternatives (e.g. non-linear functions) are worth investigating.

## 5. CONCLUSIONS

In this study we explored the property of lexical cohesion between query terms in documents: whether it is related to relevance, and whether it can be used to predict relevance in document ranking. Two hypotheses were put forward. The first hypothesis we studied was:

*Hypothesis 1:* There exists statistically significant association between the level of lexical cohesion of the query terms in documents and relevance.

We conducted experiments by building sets of relevant and non-relevant documents, calculating their lexical cohesion scores and comparing the averages of these scores. The experiments showed that there exists a statistically significant difference between the average lexical cohesion scores of relevant and non-relevant documents extracted from the top 100 and top 1000 BM25-ranked sets. We also proved that this difference is genuine, and is not affected by differences in BM25 scores or other document characteristics.

The experimental results provided support for Hypothesis 1, demonstrating that there exists a statistically significant relation between relevance and the level of lexical cohesion between query terms.

Having discovered that on the whole relevant documents have more instances of query terms than non-relevant documents, we explored another hypothesis:

*Hypothesis 1.1:* Collocational environments of different query terms are more cohesive in the relevant documents than in the nonrelevant, and this difference is not due to the larger number of query terms.

Our experiments supported the above hypothesis and showed that on average relevant documents have larger numbers of link-forming collocates, which are not original query terms, compared to nonrelevant documents. Following these experiments, we explored another hypothesis:

*Hypothesis 2:* Ranking of a document set by lexical cohesion scores results in significant performance improvement over term-based document ranking techniques.

We conducted experiments on re-ranking BM25-ranked document sets with a simple linear combination function of BM25 document matching score and the lexical cohesion score. Different values of a tuning constant *x*, regulating the effect of LCS were tried. The results suggested that there are some significant improvements over BM25 document ranking function, thus providing support for Hypothesis 2. We are aware of the fact that the function used in re-ranking the documents is simple and more elaborate methods need to be investigated.

Results achieved in the first half of this study – i.e., difference between relevant and non-relevant documents by their average lexical cohesion scores are promising. Our approach to using LCS in document ranking in the second half of the study also proved to be useful. The experiments reported suggest that the concept of lexical cohesion has strong association with document relevance, and therefore is worth further investigation. To achieve further benefit from lexical cohesion in document ranking, more experimentation is needed. In particular, problems of documents with zero LCS score and better ways of combining LCS with BM25 scores need to be investigated.

Lexical cohesion, as a text property, is formed not only through word repetition, but other more complex lexical relations. So far we looked into lexical cohesion between query terms achieved only through repetition of their collocates. Other lexical cohesion forming phenomena, such as synonymy, antonymy, hyponymy and meronymy [14] could also be taken into account in identifying lexical cohesive links between the environments of query terms. A more complete analysis of lexical environments of query terms can be expected to provide more support to the ideas behind this study. It is noteworthy to mention that an earlier analysis of lexical link distribution by Ellman [8] showed that the most common link type, repetition of the same word, is closely followed by the type of links, formed by words belonging to the same thesaurus category. A possible future development of our method could, thus, consist of defining links on the basis of repeated words and words related through either manually or automatically constructed lexical resources and thesauri.

In the reported work, all links formed by repetition are treated equally. Arguably, links formed by collocates with high inverse document frequency (*idf*) are more indicative of a strong lexical cohesion between the contexts of query terms, than links formed by words with low *idf*. For example, some collocates could be discourse-forming or topic-neutral words (e.g., "say", "report", "argue"), which tend to have low *idf*. One possible future extension of this work is to weight links using *idf* weights of the terms forming them.

Apart from being a potential aid as a ranking function, the proposed method of estimating the degree of lexical cohesion between query terms could be useful in other tasks such as query expansion and summarisation. It is likely that query terms with a strong lexical cohesion belong to the same topic, therefore they are more likely to collocate with relevant query expansion terms, than query terms with weak lexical cohesion.

## 6. REFERENCES

[1] Halliday, M.A.K. and Hasan, R. *Cohesion in English.* Longman, 1976.

[2] Manning C.D., Schütze H. *Foundations of Statistical Natural Language Processing*, The MIT Press, Cambridge, Massachusetts, 1999.

[3] Beeferman D., Berger A., Lafferty J. A model of lexical attraction and repulsion. *In Proc. ACL-EACL Joint Conference*, Madrid, Spain, 1997.

[4] Vechtomova O., Robertson S., Jones S. Query expansion with long-span collocates. Information Retrieval, 6(2), pp. 251-273, 2003.

[5] Hoey, M. *Patterns of Lexis in Text.* Oxford University Press; 1991.

[6] Morris, J. and Hirst, G. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics,* 17, 1991, pp. 21-48.

[7] Hirst, G. and St-Onge, D. Lexical chains as representation of context for the detection and correction of malapropisms. *Wordnet. An Electronic Lexical Database.* C.Fellbaum (ed.), MIT Press, 1997, pp.305-332.

[8] Ellman, J. and Tait, J. On the generality of thesaurally derived lexical links. *In the Proceedings of 5th JADT*, 2000, pp.147-154.

[9] Stairmand M.A. Textual context analysis for information retrieval. *In Proc. ACM-SIGIR*, 1997, pp. 140-147.

[10] Hearst, M. Multi-paragraph segmentation of expository text. In the Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics. 1994.

[11] Manabu O., Hajime M. Query-biased summarization based on lexical chaining. *In Computational Intelligence*, Vol. 16, N 4, 2000, pp. 578-585.

[12] Spärck Jones, K., Walker, S. and Robertson, S.E. (2000). A probabilistic model of information retrieval: development and comparative experiments. *Information Processing and Management,* 36(6), 779-808 (Part 1); 809-840 (Part 2).

[13] Clarke, C.L.A. and Cormack, G.V. On the use of regular expressions for searching text. University of Waterloo Computer Science Department Technical Report number CS-95-07, University of Waterloo, Canada, February 1995.

[14] Hasan, R. Coherence and cohesive harmony. *In Flood, J. (ed.) Understanding Reading Comprehension*. 1984. pp.181-219. Delaware: International Reading Association.

[15] Allan, J. HARD Track overview in TREC 2004 (Notebook). High Accuracy Retrieval From Documents. In Voorhees, E. and Buckland, L. (Eds.) TREC 2004 Conference Notebook Proceedings, November 2004.

# Human annotation of lexical chains: coverage and agreement measures

Bill Hollingsworth
University of Cambridge Computer Laboratory
william.hollingsworth@cl.cam.ac.uk

Simone Teufel
University of Cambridge Computer Laboratory
simone.teufel@cl.cam.ac.uk

## ABSTRACT

Lexical chains have been successfully used in several previous applications, e.g. topic segmentation and summarization. In this paper, we address the problem of how to directly evaluate the quality of lexical chains, in comparison to a human gold standard. This is in contrast to previous work, where the formal evaluation either relied on a word sense disambiguation task or concentrated on the final application result (the summary or the text segmentation), rather than the lexical chains themselves. We present a small user study of human annotation of lexical chains, and a set of measures to measure how much agreement between sets of lexical chains there is. We also perform a small meta-evaluation to compare the best of these metrics, a partial overlap measure, to rankings of chains derived by introspection, which shows that our measure agrees reasonably well with intuition. We also describe our algorithm for chain creation, which varies from previous work in several aspects (for instance the fact that it allows for adjective attribution), and report its agreement with our human annotators in terms of our new measure.

## 1. INTRODUCTION

An algorithm for creating lexical chains was first proposed by Morris and Hirst [11] and relies on the theory of lexical cohesion [4]. A lexical chain is a collection of terms that are related within a text by lexico-semantic relations, such as synonymy or similar relations.

Lexical chains have been used in various applications, such as automatic text summarization [1, 13], text segmentation [5, 14], correction of malapropisms [6], and automatic generation of hypertext links [3]. The exact mechanisms by which lexical chains are used in these approaches differs from application to application. In Barzilay et al, for instance, particularly relevant sentences useful for presentation in a summary are defined as those that contain many intersecting chains. In [5], a gap between pseudo-sentences is more likely to be a topic shift if the number of lexical chains spanning the gap is low. Importantly, the user never sees the lexical chains directly; end evaluation of these approaches is performed in terms of either a word sense disambiguation task [1, 6] or the end product (summaries, text segmentation, hypertext output). The quality of the lexical chains themselves is thus typically not formally evaluated in the existing work.

We propose lexical chains as a framework for a different task, automatic text skimming. The goal of an automatic skimmer is to provide an online user with the ability to browse the text in a document in a way similar to that of a reader of a printed document. This would allow, for example, a blind or sight-impaired person to (a) quickly decide whether a document is worth reading (or listening to) and (b) quickly find specific information within a paper.

Such a skimming system must extract topics, or concepts, that are important in a document and then organize them in such a way that they can be browsed quickly. Following the premise that a lexical chain can represent a concept expressed in a document [1], we use lexical chains to build browsable topic maps for scientific papers.

The domain for our skimmer is scientific articles. We have built a system for detecting lexical chains in these texts. The algorithm essentially follows the Silber and McCoy methodology [13], but uses a few modifications that are important in our text type. For instance, we have found in scientific papers that adjective modification is essential to characterize topics well.

Our users have direct contact with the lexical chains that our program outputs. We thus require a different kind of evaluation from previous work, namely one that formally evaluates the quality of the chains per se.

For example, consider the two lexical chains in Figure 1 that were automatically generated by our system[1]. Numbers appearing in parentheses represent the frequency of the preceding term in the paper.

We perceive the first chain in Figure 1 as a good representation of an important topic in the paper (*probabilities*), but not the second one. We want our evaluation method, which is intrinsic, to pick up on this intuitive difference. We could choose an extrinsic evaluation of the final product (e.g. to determine if a user can solve a search task faster with our system output, than with a different document such as an abstract), but this evaluation method, like the ones in previous work, does not directly tell us to which degree lexical chains are intuitive to humans, and which chains describe

[1]The example paper used throughout this article is: Lee, Lillian (1999). "Measures of Distributional Similarity". *37th Annual Meeting of the ACL*, pp. 25-32.

1. confusion probability (7), probability (3), positive probability (1), conditional cooccurrence probability (1), arbitrary probability distributions (1), probability estimate (1), probability distributions (1), probability estimation (1), probabilities (3), base probabilities (1), base language model probabilities (1), verb cooccurrence probabilities (1), unigram probabilities (1), correct probabilities (1), smooth word cooccurrence probabilities (1), conditional verb cooccurrence probabilities (1)
2. values (2), appropriate values (1), actual values (1), distribution values (1)

**Figure 1: Lexical chains automatically generated by our system for the example paper.**

the topics in a paper well. We thus opted to create a "gold standard" of lexical chains to compare against.

In this paper we report preliminary results from a small annotation study, where we asked human annotators to manually create lexical chains for two texts. We also developed and compared coverage and agreement measures that allow us to quantify the similarities between lexical chains created by different humans, and between humans and our system. We conducted these studies to shed more light on the question of how much difference there is in human intuition about lexical chains. Additionally we hope to use the human training material for a filtering component of our system, as by far the largest problem we encounter is the over-generation of lexical chains by our automatic method.

Our main contribution in this paper is the methodology of the human annotation study and the novel measures for reporting agreement between lexical chains created by different sources. We present measures for how much one single chain agrees with another single chain, and for how much a set of chains (representing one paper) agrees with another set of chains created by a different annotator.

The rest of the paper is structured as follows: the next section discusses a peculiarity of our task, namely the need to find local as well as global chains. Section 3 describes our system. The pilot study for lexical chain annotation is given in section 4. Section 5, the core of this paper, describes the coverage and agreement measures between lexical chains. We then report our system's results in terms of difference from each human annotator. The last section gives conclusions.

## 2. GLOBAL AND LOCAL CHAINS

A scientific paper will have main topics which describe the principal purpose(s) of the paper. It will also contain more localized topics. These may be associated with subsections of the paper, for the purpose of providing more detailed information. Given a paper, a concept associated with a chain need not run through the entire paper ("global chain"), but can also cover only a subset of the paper ("local chain"). Schematically, this is shown in Figure 3, where the shorter chains are local, as they refer to localized topics.

A real-world example of a global chain is chain #1 in Figure 2. This chain represents the topic *similarity measures/metrics*. Terms from this chain are used throughout the example paper by L. Lee, including the title. An example of a local chain is #2 in Figure 2. This chain represents the topic *distance-weighted averaging* and is only used in section 1 (*Introduction*) and section 3 (*Empirical Comparison.*

1. measures (15), distributional similarity measures (8), similarity functions (7), function (12), functions (14), divergence (15), similarity measures (8), similarity metric (3), similarity function (7), metric (8), similarity metrics (2), coefficient (7), measure (6), metrics (4)
2. distance-weighted (5), distance-weighted averaging (5), distance-weighted averaging model (1)

**Figure 2: Global and local chains.**

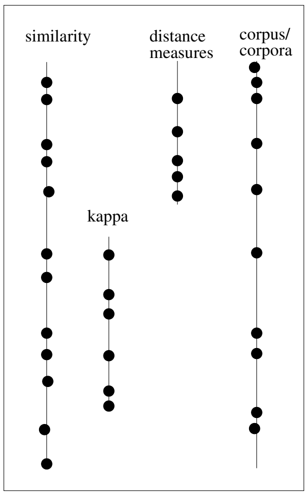


**Figure 3: Examples of local and global chains**

We have discussed local chains, and why they are important in our task. This motivates a slightly unorthodox approach to chain membership; we allow one term to occur in more than one chain, unlike previous definitions [1, 13]. This is necessary in order to allow for the parallel existence of global and local chains that cover similar aspects of a concept. For instance, the local chain in Figure 4 contains "divergence" as its most frequent term, a term which also belongs to a different, global chain. We would not want to have to exclude all occurrences of divergence from the global chain, just to enforce the uniqueness constraint.

## 3. THE SYSTEM

Our lexical chainer is based on the Silber and McCoy generation algorithm and uses the Barzilay and Elhadad scoring system. Both approaches used WordNet [10] to group together related words. The scoring system we use is due to Barzilay and Elhadad ([1]) scored chains independently from each other by assigning different scores to each type of relation. For instance, word repetition contributed the

divergence (15), total divergence (1), skew divergence (5), jensen-shannon divergence (7), kl divergence (5), outliers (1)

**Figure 4: An example local chain.**

| Text | A | B | C | Average |
|---|---|---|---|---|
| 1 | 3.5% | 0% | 1.4% | 2% |
| 2 | 10.2% | 9.4% | 0% | 7% |

**Figure 5: Presence of WordNet relations in human-generated chains.**

most points (7), followed by the synset relation (4). Each of the other relations contributed 1 point. We modify this scheme by only using synonymy and hypernymy/hyponymy, like Silber and McCoy.

We follow Silber and McCoy's ([13]) linear time algorithm for creating chains and performing word-sense disambiguation. This is done by first ranking the chains by score. If the score of the highest-scoring chain is above a set threshold, then the chain is added to the final set of lexical chains for the document. Each word in the chain is then removed from all other chains, and the chains are rescored and reranked. This is repeated until there is no chain left that has a score higher than the threshold and has not already been added to the final chain list. Each word, therefore, belongs to at most one lexical chain in the final chain list. In each approach, only nouns and noun compounds were considered for membership in lexical chains.

The differences between our lexical chainer and the Silber and McCoy system are discussed in the subsections that follow.

## 3.1 Multi-word scientific terms and WordNet

Scientific papers tend to use a large number of multi-word terms [7]. Such terms are usually not present in WordNet. For each noun compound which does not exist in WordNet, Barzilay and Elhadad assign the compound the WordNet synset value of the head noun. Analogous to this 'shared head' relation, we additionally define a 'shared modifier' relation. This allows two terms which share one or more modifiers to be included in the same lexical chain (e.g. *similarity measure* and *similarity distribution*. The decision of including a 'shared modifier' relation is supported by our annotation data (cf. Figure 6).

distributional similarity measures, similarity functions, similarity measures, similarity metric, similarity function, similarity metrics

**Figure 6: An example of a lexical chain whose terms are related by premodification and require a partial overlap relation.**

Stokes [14] found that the WordNet relations played less of a role in her lexical chains than repetition did. She attributes part of this effect to the sparcity of compound terms in WordNet. Our annotation results suggest that for scientific texts WordNet relations play an even smaller role than in news texts, as shown in Figure 5; on average, only 4% of the relations in our chains correspond to WordNet relations. This seems to be in line with Justeson and Katz's argument technical that terms tend to be repeated instead of substituted [7]. Indeed all of the WordNet relations we observed between single-word terms (e.g. *probability/chance* and *data/corpus*).

Additionally we also believe that the word sense disambiguation problem is less acute in scientific text, because (1) the terms are naturally longer and thus more specific, and (2) word sense variation is lower within a subfield.

## 3.2 Adjectives

The importance of adjectives as premodifiers in technical terms in scientific text is well-acknowledged [7]. In addition, Justeson and Katz report that 4% of the terms in their dictionary sample are single adjectives or adjective phrases. Our approach to lexical chaining allows adjectives as modifiers in noun phrases and as heads of adjective phrases to be chain candidates.

| Text | A | B | C | Average |
|---|---|---|---|---|
| 1 | 29% | 42% | 37% | 36% |
| 2 | 19% | 32% | 39% | 30% |

**Figure 7: Percentage of adjectives in human-generated lexical chains.**

The data produced by our annotators suggest that humans do indeed heavily incorporate adjectives into their lexical chains. Overall, 37% of all term types are adjectives or contain adjectives (30% for annotator A, 41% for annotator B, and 38% for annotator C). Of the two texts that we had annotated, Text 1 seems to have more adjectives than text 2 (cf. Figure 7). For text 1, for example, 81% of all of the human-generated chains contained at least one adjective.

Our decision to include adjectives into lexical chains is in contrast to previous work in lexical chains: Barzilay and Elhadad only allow nouns to be considered in creating terms for lexical chains, as do Morris and Hirst [11] (who work on Reader's Digest articles) and Silber and McCoy [13]. Stokes [15] uses adjectives which form part of a complex proper noun such as *Irish* in compound terms like *Irish journalist.* We believe that the importance of adjectives varies considerably with genre (Barzilay and Elhadad used newspaper texts, Morris and Hirst [12] Reader's Digest articles.

However, we believe that not all adjective in scientific text are equally important to represent a scientific text accurately. In a term such as *statistical significance*, *statistical* disambiguates the sense of *significance* whereas an adjective like *higher* in *higher significance* does not. Levi [9] calls adjectives such as *statistical* non-predicating. We have implemented an algorithm based on some of her linguistic tests to filter out predicating adjectives. This algorithm is however not the focus of the current paper; we will report about it elsewhere.

## 3.3 Non-uniqueness of chain membership

Silber and McCoy restrict each term to appearing in only one chain. The "best" chain for a given term is chosen and that term is removed from the rest of the chains. Our chainer allows a term to appear in multiple chains. In scientific papers, a term may intuitively belong to a global topic and to a local topic. For example, chain #1 in Figure 8 represents the global topic *probability* and contains the term *cooccurrence probability.* Chain #2 in the same figure represents the local topic *cooccurrence* and also contains the term *probability cooccurrence.*

The annotation guidelines allow the possibility of using a term in more than one chain but leaves the decision up to the annotators. All three of the current annotators used at

1. probability (15), probability estimation (1), conditional cooccurrences probability (2), cooccurrence probability (2), probability distribution (1), confusion probabilities (1), frequencies (3), conditional verb cooccurrence probabilities (2), relative frequencies (2), unigram probabilities (1), word cooccurrence probabilities (2), conditional probabilities (3), likelihoods (1), base probabilities (3), likely (2), probability estimate (2)
2. cooccurrences (3), cooccurrence probability (2), word cooccurrence probability (2), cooccurrence pair (1), conditional verb cooccurrence (1), verb-object cooccurrence pairs (1)

**Figure 8: Membership of a term in a global and a local chain.**

least one term in more than one chain.

## 4. PILOT STUDY

Measuring the extent to which human intuitions about lexical chains agree is an interesting task, both from a psycholinguistic viewpoint as well as from a practical one. A lexical chaining algorithm was first proposed by Morris and Hirst [11], based on the idea of lexical cohesion as in [4]. Even though it seems clear that most humans intuitively understand the concept of lexical chains, few experiments of the psycholinguistic plausibility of actual chain construction have been performed. Morris and Hirst [12] present a pilot study of the subjectivity of readers' perceptions of relations between words that make up lexical chains. The domain for this study was a collection of general-interest articles taken from Reader's Digest. Five subjects were asked to read the first 1.5 pages of an article and mark each word group that they perceived. For each word group, they identified pairs of related words and the relation between them. The subjects agreed on a subset of the word groups while also having individual variation.

They point out that the "degree of individual difference or subjectivity in text understanding is likely to vary with text type." It is thus necessary for us to collect annotators' perception and agreement data for the text type we work on, scientific domain.

As we already motivated, we also have practical reasons for creating a manual training set of lexical chains: we need them to directly evaluate the quality of our automatically created lexical chains, and we intend to use them as training material to learn to recognize weak chains in order to remove them from the final lexical chain set.

Because of our focus on scientific papers, we decided to also perform annotation, choosing to randomly select papers from the ACL anthology as our data.

Experimental design is as follows: We use three annotators, who are given a set of materials as described below. Annotator A is a doctoral student in computer science. Annotator B is the second author of this paper, and annotator C is the first author of this paper. The annotators are given unrestricted time to create sets of terms that they judge to be related given the context of the paper. Each set of terms then represents one lexical chain. The guidelines are four pages long and essentially describe the task as follows: A term can comprise a single word or a combination of words, all taken directly from the text. Words used in terms may be nouns, adjectives, or adverbs. Possible relationships between terms in a chain are mentioned which include inflectional variance, synonymy, hypernymy/hyponymy, holonymy, and meronymy.

There are no limits placed on the size of lexical chains or the number of chains needed to describe a document. We found that there are many intuitive similarities between chains created by our annotators. There are also many differences, such as in the number of chains used and in the exact terms that are used.

For the ongoing annotation experiment, human annotators are given a collection of materials including a list of all words in the paper together with part-of-speech tags generated by RASP [2]. Each annotator is also given a list of maximal noun phrases automatically extracted from the paper. Use of these lists is optional, but they are provided as different visualizations of the terms in the paper.

To measure the agreement between two annotators we need a metric that will do the following:

1. When comparing two lexical chains, one chain should be penalized for not covering its topic as well as a competing chain.
2. When comparing two sets of lexical chains, one chain set should be punished for not covering the paper as well as a competing chain set.
3. A chain set should be penalized for splitting chains (i.e. using two chains to describe the same topic), in comparison to having identical chains (non-split chains), but it should penalize it less than in a situation where one of the split chains is missing or replaced with irrelevant terms.
4. A chain set should be penalized for merging chains (i.e. combining multiple concepts into one chain); see above.

We use a token-based approach to comparing chains rather than a type-based approach because we believe term repetition in scientific texts to be a strong indicator of the relevance of topics.

Section 5 describes some coverage and agreement measures that we are using to evaluate lexical chains and sets of lexical chains.

## 5. COVERAGE AND AGREEMENT MEASURES

### 5.1 Comparing lexical chains

In this section we compare four measures for computing the similarity between two lexical chains. We discuss the properties of each measure and how they affect the usefulness of the measure for our task.

When comparing two lexical chains $x$ and $y$, two (not necessarily equal) agreement measurements are important:

1. The degree to which $y$ is covered by or similar to $x$
2. The degree to which $x$ is covered by or similar to $y$

To compute chain set agreement between two annotators (or chain set similarity between two papers), we find (for each chain in chain set $A$) the best match in chain set $B$, according to either measure detailed below. Adding together the agreement scores for each match gives us Equation 1.

$$m(A,B) = \sum_{x \in A} \frac{m_1(x,B)|x|}{|A|}. \quad (1)$$

$m(A,B)$ measures the degree to which all chains in $A$ cover any of the chains in $B$.

## 5.2 Cosine measure

For a baseline, we use the standard cosine metric. Each lexical chain is represented as a vector of term frequencies. Of the measures considered here, the cosine metric is the only one that is symmetric.

## 5.3 KL distance

Another comparison measure that we evaluated is the Kullback-Leibler (KL) distance [8]. It is a measure of similarity between two distributions, as defined in Equation 2.

$$KL(P,Q) = \sum_{i=0}^{n} p_i \log_2(\frac{p_i}{q_i}), \quad (2)$$

where $P = (p_0, ..., p_n)$ and $Q = (q_0, ..., q_n)$ are probability distributions.

We compare chains by representing each chain as a vector of relative term frequencies. Suppose we wish to compare chains $X$ and $Y$. Since both distributions in Equation 2 must contain the same number of points, we set the length of the vector for chain $X$ and the length of the vector for chain $Y$ equal to the order of the union of the terms in $X$ and $Y$. This means that for two chains that do not have exactly the same terms, their corresponding vectors will contain 0-values representing terms missing from the chain. Since each value in $P$ and $Q$ must be nonzero for KL, we perform simple add-one smoothing.

## 5.4 Term overlap

We also consider simple term overlap

$$c(x,y) = \frac{|x \cap y|}{|y|}. \quad (3)$$

Two chains $x$ and $y$ are treated as sets of tokens (with multiplicity).

We measure the coverage of $B$ by $x$ as

$$m_1(x,B) = \max_{y \in B} c(x,y). \quad (4)$$

Similarly, we measure the coverage of $x$ by $B$ as

$$m_2(x,B) = \max_{y \in B} c(y,x). \quad (5)$$

Note that $m_1(x,B)$ and $m_2(x,B)$ need not be maximized by the same $y \in B$.

## 5.5 Shared modifier relation

We modify our overlap measure by allowing partially overlapping terms to count as partial matches. The overlap measure in Equation 3 only recognizes exact term matches, but semantics is shared between terms even if there is a partial overlap (e.g., in modifiers or heads). We assign a weight of 0.3 to this relation.

| Measure | A→B | A→C | B→C |
|---|---|---|---|
| Cosine | 18% | 0% | 7% |
| KL | 82% | 62% | 71% |
| Overlap | 100% | 62% | 71% |
| Shared modifier | 100% | 69% | 86% |

**Figure 9: Agreement between automatically matched chains and manually matched chains for Text 1.**

| Measure | B→A | C→A | C→B |
|---|---|---|---|
| Cosine | 8% | 0% | 8% |
| KL | 38% | 42% | 71% |
| Overlap | 69% | 84% | 67% |
| Shared modifier | 69% | 84% | 67% |

**Figure 10: Agreement between automatically matched chains and manually matched chains for Text 1.**

## 5.6 Preliminary results

### 5.6.1 *Testing the measures*

To test the four measures described above, we look for the strongest chain matches between two annotators. That is, given two annotators $A$ and $B$, each chain from annotator $A$ is matched with the most similar chain from annotator $B$, and vice versa. Performing this task using each measure gives us four sets of chain matches for each annotator pair (going one direction). Each set of matches is then compared to a manually generated set of chain matches for the same annotator pair.

As we can see in Figures 9-13, the cosine metric performs badly when matching chains. This is primarily because a metric based on the inner product of two vectors does not issue a penalty when vectors of different lengths are compared (an attractive property in IR when comparing documents with queries). Thus, the chains that are found to match using this metric may have high frequency terms in common but may also contain several other terms not shared by other chains.

The shared-modifier algorithm had a slight improvement over the overlap measure when finding chain matches, and thus outperformed the KL distance and the baseline.

### 5.6.2 *Comparing rankings*

Using the chain match scores given by the measures, we can rank the strength of the chain matches. For each measure we compare the top five chains to a manual ranking of the top five chain matches for each annotator pair. We only consider the top five matches to avoid having to compare match strengths between chains with little in common. Since the cosine metric performed so poorly when finding matches, we only compared rankings for the other three

| Measure | A→B | A→C | C→B |
|---|---|---|---|
| Cosine | 29% | 17% | 27% |
| KL | 86& | 67% | 73% |
| Overlap | 86% | 83% | 64% |
| Shared modifier | 86% | 83% | 91% |

**Figure 11: Agreement between automatically matched chains and manually matched chains for Text 2.**

| Measure | B→A | B→C | C→A |
|---|---|---|---|
| Cosine | 0% | 30% | 13% |
| KL | 88% | 80% | 75% |
| Overlap | 6/8 | 80% | 75% |
| Shared modifier | 100% | 80% | 75% |

**Figure 12: Agreement between automatically matched chains and manually matched chains for Text 2.**

| Measure | Text 1 | Text 2 |
|---|---|---|
| Cosine | 6% | 20% |
| KL | 61% | 78% |
| Overlap | 74% | 76% |
| Shared modifier | 78% | 86% |

**Figure 13: Average agreement between automatically matched chains and manually matched chains.**

measures.

Figure 14 shows the agreement between top matches selected by the different measures and those selected manually.

## 6. CONCLUSIONS AND FUTURE WORK

Our main contributions in this paper are our methodology of the human annotation study and a comparison of four similarity measures (including a new measures based on shared modifiers) for reporting agreement between lexical chains created by different sources. Our annotation study covers the scientific domain with the goal of training a lexical chaining system for scientific papers.

This pilot study explores the extent to which human-generated lexical chains agree in the domain of scientific texts. In future work, we will investigate the role that non-uniqueness of term membership plays in creating local chains. As we build our gold standard we hope to determine the importance of adjectives in human-generated lexical chains in the scientific domain.

Limitations of our preliminary study are:

1. We have too few annotators and use too few papers for an extensive study of lexical chain agreement in the scientific domain. This will be expanded in later work.

2. Our coverage and agreement measures do not yet handle all of the cases that we want to consider (e.g. the merger of two chains). The comparison ranking produced by our measures and presented in this paper compares well with an intuitive ranking for the most important matches, but compares badly overall.

Future work will address the problems mentioned above.

| Match | KL | Overlap | Shared modifier |
|---|---|---|---|
| A→B | 2 | 3 | 3 |
| B→A | 2 | 2 | 2 |
| B→C | 2 | 2 | 2 |
| C→B | 3 | 2 | 2 |
| A→C | 2 | 4 | 4 |
| C→A | 4 | 2 | 2 |

**Figure 14: Number of chain matches ranking in the top five as compared to the manually ranked top five matches.**

## 7. REFERENCES

[1] R. Barzilay and M. Elhadad. Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS'97).* ACL Madrid, 1997.

[2] E. Briscoe and J. Carroll. Robust accurate statistical annotation of general text. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 1499–1504, Las Palmas, Canary Islands, May 2002. (LREC 2002).

[3] S. Green. Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering*, 1999.

[4] M. Halliday and R. Hasan. *Cohesion in English.* Longman, London, 1976.

[5] M. A. Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 1997.

[6] G. Hirst and St-Onge. Lexical chains as representation of context for the detection and correction of malapropisms. In *Fellbaum, C., ed., WordNet: An Electronic Lexical Database and Some of its Applications.* The MIT Press, 1998.

[7] J. S. Justeson and S. M. Katz. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27, 1995.

[8] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statsistics*, 22:79–86, 1951.

[9] J. Levi. *The Syntax and Semantics of Complex Nominals.* Academic Press, New York, 1978.

[10] R. F. C. G. D. Miller, G.; Beckwith and K. Miller. Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography (special issue)*, 3(4).

[11] J. Morris and G. Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1), 1991.

[12] J. Morris and G. Hirst. The subjectivity of lexical cohesion in text. In *Shanahan, James G.; Qu, Yan; and Wiebe, Janyce (Eds.) Computing attitude and affect in text*, Dordrecht, The Netherlands, 2005.

[13] H. G. Silber and K. F. McCoy. Efficiently computed lexical chains as an intermediate representation in automatic text summarization. *Computational Linguistics*, 28(4), 2002.

[14] N. Stokes. Spoken and written news story segmentation using lexical chaining. In *Proceedings of the Student Workshop at HLT-NAACL 2003, Companion Volume*, pages 49–54, Edmonton, Canada, 2003.

[15] N. Stokes. *Applications of Lexical Cohesion Analysis in the Topic Detection and Tracking Domain.* Department of Computer Science, University College Dublin, 2004.

## Appendix

Figures 15 and 16 show all of the chains constructed by two of the annotators for the example paper. Numbers appearing in parentheses represent the frequency of the preced-

1. **similarity**, measures (15), distributional similarity measures (8), similarity functions (7), function (12), functions (14), divergence (15), similarity measures (8), similarity metric (3), similarity function (7), metric (8), similarity metrics (2), coefficient (7), measure (6), metrics (4)
2. **support**, **regions**, supports (7), regions of positive probability (1), support-intersection data (1), support (3), support regions (1)
3. **similarity**, distributional similarity (7), similar (7), distance-weighted (5), distributional similarity (7), semantic similarity (1), distance (2), dissimilarity (1), similarity-based (1), similarity (28), commonality (2), differences (3)
4. **unseen**, unseen cooccurrences (2), sparse data (2), low frequency events (1), unseen events (2), unseen word pair (1), unseen (8), unseen pairs (1), sparseness (1)
5. **cooccurrence**, **cooccurrences**, cooccurrence (6), cooccurrences (3), word cooccurrence (2), neighborhood (1), closest neighbors (1), nearest neighbors (2)
6. **probability**, **estimate**, **estimation**, probability estimation (1), estimate (4), similarity-based estimation (1), probability estimate (2), estimates (2)
7. **comparison**, **empirical**, empirical comparison (3), comparison (5)
8. **distributions**, potential proxy distributions (2), distributional (8), probability distributions (2), distributions (8), potential proxy distributions (2), joint distribution (1), product distribution (1)
9. **training**, **corpus**, training corpus (2), training set (2), training partition (1), training corpus (2), training data (4)
10. **probabilities**, probability (15), conditional cooccurrence probability (2), probability distributions (2), chance (2), distributions (8), probabilities (10), verb cooccurrence probabilities (2), smooth word cooccurrence probabilities (2), base language model probabilities (2), confusion probability (7), unigram probabilities (1), likelihoods (1), conditional verb cooccurrence probabilities (1), mathematical certainty (1)
11. **events**, **data**, events (5), data (11), bigrams (1), words (7), word pair (1), cooccurrences (3), words (7), data (11), nouns (6), verbs (11), cooccurrence pair (1), noun (2), corpus (2), adjectives (1), pairs (4), noun-verb pair (1), noun-verb-verb triple (1), test triple tokens (1), test instance (1)
12. **probability**, **probability distributions**, probability (15), probability distributions (2), chance (2), average (6), statistically (2), insignificant (1), unsmoothed (1), frequencies (3), smooth (2), relative frequencies (2), likelihoods (1), joint distribution (1), product distribution (1), unigram frequencies (1), error rate (4), statistic (1), t-test (1), significance level (1), mathematical certainty (1)

**Figure 15: Annotator A's lexical chains for the example paper.**

ing term in the paper. Terms appearing in bold are chain representatives and were automatically extracted from the manual chains.

1. **similarity**, similarity (28), distributional similarity measures (8), similarity functions (7), distributional similarity (7), semantic similarity (1), similarity measures (8), similarity metric (3), similar words (5), new similarity metrics (2), extreme dissimilarity (1), similarity-based estimation (1), inherently better similarity ranking (2), good similarity metric (3)
2. **probabilities**, probability (15), probability estimation (1), conditional cooccurrences probability (2), cooccurrence probability (2), probability distribution (1), confusion probabilities (1), frequencies (3), conditional verb cooccurrence probabilities (2), relative frequencies (2), unigram probabilities (1), word cooccurrence probabilities (2), conditional probabilities (3), likelihoods (1), base probabilities (3), likely (2), probability estimate (2)
3. **distribution**, distribution (6), proxy distributions (2), probability distribution (1), average distribution (1), joint distribution (1), product distribution (1), empirical distribution (1)
4. **unseen**, sparse data (2), low frequency events (1), unseen cooccurrences (2), unseen word pair (1), unseen (8), unseen pairs (1), sparseness (1)
5. **training**, training set (2), estimate (4), training partition (1), test-set bigrams (1), training corpus (2), test sets (1), test-set performance (2)
6. **concurrence**, cooccurrences (3), cooccurrence probability (2), word cooccurrence probability (2), cooccurrence pair (1), conditional verb cooccurrence (1), verb-object cooccurrence pairs (1)
7. **method**, **backoff**, backoff method (2), interpolation method (1), backoff smoothing method (1)
8. **distance-weighted**, **averaging**, distance-weighted (5), distance-weighted averaging (5), distance-weighted averaging model (1)
9. **divergence**, divergence (15), total divergence (1), skew divergence (5), jensen-shannon divergence (7), kl divergence (5), outliers (1)
10. **significant**, statistically significant (2), significant (3), significance level (1)
11. **evaluation**, **pseudoword disambiguation task**, evaluation (3), pseudoword disambiguation task (1), empirical results (1), decision task (3), empirical comparison (3), evaluation methodology (1), binary decision task (3), experimental framework (1), correct answer (1), paired t-test (1), prediction tasks (1)
12. **information theoretic metric**, **similarity metric**, information-theoretic metric (1), similarity metric (3), similarity measures (8), cosine metric (2), jaccard coefficient (1), jensen-shannon divergence (7), kl divergence (5), nonparametric measure (1), correlation (1), mutual information (1), value difference metric (2), dice coefficient (1), l2 norm (1), l1 norm (3), statistic (1), euclidean distance (1), skew divergence (5), alpha - skew divergence (5), good similarity metric (3), similarity function schema (1)
13. **performance**, **average**, performance (10), precision (1), average performance (3), average error rate (4), test-set performance (2)
14. **nouns**, **verbs**, nouns (6), verbs (11), transitive verbs (1), head noun (1), direct object (1), similar adjectives (1), frequent nouns (1), noun-verb pair (1), noun-verb-verb triple (1)
15. **smoothing**, **unsmoothed**, smoothing (1), unsmoothed (1), smoothed base language model (5)
16. **model**, **language**, language model (5), language model probabilities (2), language modeling (1), smoothed base language model (5), model (9)
17. **neighbors**, neighborhood (1), neighbors (3), nearest neighbors (2)
18. **function**, **weighting**, **weight**, weighting (1), weight function (1)
19. **substitutability**, substitutability (1)
20. **generalization**, **asymmetric**, **novel**, **symmetric**, symmetric (2),s novel asymmetric generalization (1)
21. **information**, **negative**, negative information (1)
22. **translations**, **mutual**, translations (1), mutual translations (1)

**Figure 16: Annotator B's lexical chains for the example paper.**

# A method to calculate probability and expected document frequency of discontinued word sequences

Antoine Doucet
Department of Computer Science
P.O. Box 68
FIN-00014 University of Helsinki
doucet@cs.helsinki.fi

Helena Ahonen-Myka
Department of Computer Science
P.O. Box 68
FIN-00014 University of Helsinki
hahonen@cs.helsinki.fi

## ABSTRACT

In this paper, we present a novel technique for calculating the probability of occurrence of a *discontinued* sequence of $n$ words, that is, the probability that those words occur, and that they occur in a given order, regardless of which and how many other words may occur between them.

Our method relies on the formalization of word occurrences into a Markov chain model. Numerous techniques of probability and linear algebra theory are exploited to offer an algorithm of competitive computational complexity. The technique is further extended to permit the calculation of the *expected document frequency* of an $n$-words sequence in an efficient manner.

We finally present an application of this technique; A fast and automatic direct evaluation of the interestingness of word sequences, by comparing their expected and observed frequencies.

## Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

## General Terms

Algorithms, Experimentation, Theory

## 1. INTRODUCTION

The probability of occurrence of words and phrases is a crucial matter in all domains of information retrieval. All language models rely on such probabilities. However, while the probability of a word is frequently based on counting its total number of occurrences in a document collection (its *collection frequency*), calculating the probability of a phrase is far more complicated. Counting the number of occurrences of a multi-word unit is often intractable, unless restrictions are adopted, such as setting a maximal unit size, requiring word adjacency or setting a maximal distance between two words.

Due to the higher information content and specificity of phrases versus words, information retrieval researchers have always been interested in multi-word units. The first models, introduced until the late 1980's, came with numerous restrictions. Mitra et al. [10], for example, defined phrases as adjacent pairs of words occurring in at least 25 documents of the TREC-1 collection. Choueka et al. [2] later extracted adjacent word sequences of length up to 6. The extraction of sequences of longer size was then intractable. The adjacency constraint is regrettable, as natural language often permits to express similar concepts by introducing one or more words between two others. For example, the phrases "President John Kennedy" and "President Kennedy" are likely to refer to the same person.

A new trend started in the 1980's, as linguistic information started to be used to filter out "undesirable" patterns. The idea consists in using parts-of-speech (POS) analysis to automatically select (or skip) the phrases matching a given set of linguistic patterns. Most recent extraction techniques still rely on a combination of statistical and syntactical methods [13, 7].

However, at a time when multilingual information retrieval is in full expansion, we think it is of crucial importance to propose language-independent techniques. There is very few research in this direction, as was suggested by a recent workshop on multi-word expressions [14] where most of the 11 accepted papers presented monolingual techniques, in a total of 6 distinct languages.

Dias et al. [4, 5] introduced an elegant generalization of conditional probabilities to $n$-grams extraction. The normalized expectation of an $n$-words sequence is the average expectation to see one of the words occur in a position, given the position of occurrence of all the others. Their main metric, the mutual expectation, is a variation of the normalized expectation that rewards $n$-grams occurring more frequently. While the method is language-independent and does not require word adjacency, it still recognizes phrases as a very rigid concept. The relative word positions are fixed, and to recall our previous example, no relationship is taken into account between "President John Kennedy" and "President Kennedy".

We present a technique that permits to efficiently calculate the exact probability (respectively, the expected document frequency) of a given sequence of $n$ words to occur in this order in a document of size $l$, (respectively, in a document collection $D$) with an unlimited number of other words eventually occurring between them.

The main challenges we had to handle in this work were to avoid the computational issue of using a potentially unlimited distance between each two words, while not making those distances rigid (we do see an occurrence of "President Kennedy" in the text fragment "President John Kennedy"). Achieving language-independence (neither stoplists nor POS analysis are used) and dealing with document frequencies rather than term frequencies are further specificics of this work.

By comparing observed and expected frequencies, we can estimate the interestingness of a word sequence. That is, the more the actual number of occurrences of a phrase is higher than its expected frequency, the stronger the lexical cohesion of that phrase. This evaluation technique is entirely language-independent, as well as domain- and application-independent. It permits to efficiently rank a set of candidate multi-word units, based on statistical evidence, without requiring manual assessment of a human expert.

The techniques presented in this paper can be generalized further. The procedure we present for words and documents may indeed similarly be applied to any type of sequential data, e.g., item sequences and transactions.

In the next section, we will introduce the problem, present an approximation of the probability of occurrence of an $n$-words sequence, and describe our technique in full details before analyzing its computational complexity and showing how it outperforms naive approaches. In section 3, we will explain how the probability of occurrence of an $n$-words sequence in a document can be generalized to compute its expected document frequency in a document collection, with a very reasonable complexity. Section 4 explains and experiments the use of statistical testing as an automatic way to evaluate and rank general-purpose non-contiguous lexical cohesive relations. We conclude the paper in section 5.

## 2. PROBABILITY OF OCCURRENCE OF A DISCONTINUED WORD SEQUENCE

### 2.1 Problem Definition

Let $A_1 A_2 \dots A_n$ be an $n$-gram, and $d$ a document of length $l$ (i.e., $d$ contains $l$ word occurrences). Each word $A_i$ is assumed to occur independently with probability $p_i$.

**Problem:** In $d$, we want to calculate the probability $P(A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_n, l)$ of the words $A_1$, $A_2$, $\dots$, $A_n$ to occur at least once in this order, an unlimited number of interruptions of any size being permitted between each $A_i$ and $A_{i+1}$, $1 \leq i \leq (n-1)$.

#### *2.1.1 More definitions*

Let $D$ be the document collection, and $W$ the set of all distinct words occurring in $D$. The probability $p_w$ of occurrence of a word $w$ is its collection frequency divided by the total number of word occurrences in the document collection. One reason why we prefer collection frequency versus, e.g., document frequency, is that in this case, the set of all word probabilities $\{p_w \mid \forall w \in W\}$ is a (finite) probability space. Indeed, we have

$$\sum_{w \in W} p_w = 1, \text{ and } p_w \geq 0, \forall w \in W.$$

For convenience, we will simplify the notation of $p_{A_i}$ to $p_i$, and define $q_i = 1 - p_i$, the probability of non-occurrence of the word $A_i$.

#### *2.1.2 A running example*

Let there be a hypothetic document collection containing only three different words $A$, $B$, and $C$, each occurring with equal frequency. We want to find the probability that the bigram $A \rightarrow B$ occurs in a document of length 3.

For this simple example, we can afford a manual enumeration. There exists $3^3 = 27$ distinct documents of size 3, each occurring with equal probability $\frac{1}{27}$. These documents are: $\{AAA, \boxed{\text{AAB}}, AAC, \boxed{\text{ABA}}, \boxed{\text{ABB}}, \boxed{\text{ABC}}, ACA, \boxed{\text{ACB}}, ACC, BAA, \boxed{\text{BAB}}, BAC, BBA, BBB, BBC, BCA, BCB, BCC, CAA, \boxed{\text{CAB}}, CAC, CBA, CBB, CBC, CCA, CCB, CCC\}$
The seven framed documents contain the $n$-gram $AB$. Thus, we have $p(A \rightarrow B, 3) = \frac{7}{27}$.

### 2.2 A decent over-estimation in the general case

We can attempt to enumerate the number of occurrences of $A_1 \dots A_n$ in a document of size $l$, by separately counting the number of ways to form the $(n-1)$-gram $A_2 \dots A_n$, given the $l$ possible positions of $A_1$. For each of these possibilities, we can then separately count the number of ways to form the $(n-2)$-gram $A_3 \dots A_n$, given the various possible positions of $A_2$ following that of $A_1$. We repeat this process recursively until we need to find the number of ways to form the 1-gram $A_n$, given the various positions left for $A_{n-1}$.

This enumeration leads to $n$ nested sums of binomial coefficients:

$$\sum_{pos_{A_1}=1}^{l-n+1} \left( \sum_{pos_{A_2}=pos_{A_1}+1}^{l-n+2} \left( \dots \sum_{pos_{A_n}=pos_{A_{n-1}}+1}^{l} \binom{l - pos_{A_n}}{0} \right) \right), \tag{1}$$

where each $pos_{A_i}$, $1 \leq pos_{A_i} \leq l$, denotes the position of occurrence of $A_i$.

The following can be proven easily by induction:

$$\sum_{i=k}^{n} \binom{i}{k} = \binom{n+1}{k+1},$$

and we can use it to simplify formula (1) by observing that:

$$\sum_{pos_{A_i}=pos_{A_{i-1}}+1}^{l-n+i} \binom{l - pos_{A_i}}{n-i} = \sum_{pos_{A_i}=n-i}^{l-pos_{A_{i-1}}-1} \binom{pos_{A_i}}{n-i} = \binom{l - pos_{A_{i-1}}}{n-i+1}.$$

Therefore, leaving further technical details to the reader, the previous nested summation (1) interestingly simplifies to $\binom{l}{n}$, which permits to obtain the following result:

$$enum\_overestimate(A_1 \rightarrow \cdots \rightarrow A_n, l) = \binom{l}{n} \cdot \prod_{i=1}^{n} p_i,$$

where $\binom{l}{n}$ is the number of ways to form the $n$-gram, and $\prod_{i=1}^{n} p_i$ the probability of conjoint occurrence of the words $A_1, \dots, A_n$ (since we assumed that the probability of occurrence of a word in one position is independent of which words occur in other positions).

The big flaw of this result, and the reason why it is an approximation only, is that some of the ways to form the $n$-gram are obviously overlapping. Whenever we separate the alternative ways to form the $n$-gram, knowing that $A_k$ occurs in position $i$, we do ignore the fact that $A_k$ may also occur before position $i$. In this case, we enumerate each possible case of occurrence of the $n$-gram, but we count some of them more than once, since it is actually *the ways* to form the $n$-gram that are counted.

**Running Example.** This is better seen by returning to the running example presented in subsection 2.1.2. As described above, the upper-estimate of the probability of the bigram $A \to B$, based on the enumeration of the ways to form it in a document of size 3 is: $\binom{3}{2}\left(\frac{1}{3}\right)^2 = \frac{9}{27}$, whereas the actual probability of $A \to B$ is $\frac{7}{27}$. This stems from the fact that in the document $AAB$ (respectively $ABB$), there exists two ways to form the bigram $A \to B$, using the two occurrences of $A$ (respectively $B$). Hence, out of the 27 possible equiprobable documents, 9 ways to form the bigram $A \to B$ are found in the 7 documents that contain it.

With longer documents, the loss of precision due to those cases can be considerable. Still assuming we are interested in the bigram $A \to B$, we will count one extra occurrence for every document that matches *A*B*B*, where * is used as a wildcard. Similarly, 8 ways to form $A \to B$ are found in each document matching *A*A*B*B*B*B*.

## 2.3 An exact formalization based on Markov Chains

### 2.3.1 An absorbing Markov Chain.

One interesting way to formalize the problem is to consider it as a sequence of $l$ trials with outcomes $X_1, X_2, \ldots, X_l$. Let each of these outcomes belong to the set $\{0, 1, \ldots, n\}$, where the outcome $i$ signifies that the $i$-gram $A_1A_2 \ldots A_i$ has already occurred. This sequence of trials verifies the following two properties:

(i) All the outcomes $X_1, X_2, \ldots, X_l$ belong to a finite set of outcomes $\{0, 1, \ldots, n\}$ called the *state space* of the system. If $i$ is the outcome of the $m$-th trial ($X_m = i$), then we say that the system is in state $i$ at the $m$-th step. In other words, the $i$-gram $A_1A_2 \ldots A_i$ has been observed after the $m$-th word of the document.

(ii) The second property is called the *Markov property*: the outcome of each trial depends at most upon the outcome of the immediately preceding trial, and not upon any other previous outcome. In other words, *the future is independent of the past, given the present.* This is verified indeed; if we know that we have seen $A_1A_2 \ldots A_i$, we only need the probability of $A_{i+1}$ to determine the probability that we will see more of the desired $n$-gram during the next trial.

These two properties are sufficient to call the stochastic process we just defined a (finite) *Markov chain*. The problem can thus be represented by an $(n+1)$-states Markov chain $M$ (see figure 1). The state space of the system is $\{0, 1, \ldots, n\}$ where each state, numbered from 0 to $n$ tells how much of the $n$-gram has already been observed. Presence in state $i$ means that the sequence $A_1A_2 \ldots A_i$ has been observed. Therefore, $A_{i+1} \ldots A_n$ remains to be seen, and the following expected word is $A_{i+1}$. It will be the next word with probability $p_{i+1}$, in which case a state transition will occur from $i$ to $(i+1)$. $A_{i+1}$ will not be the following word with probability $q_{i+1}$, in which case we will remain in state $i$. Whenever we reach state $n$, we can denote the experience a success: the whole $n$-gram has been observed. The only outgoing transition from state $n$ leads to itself with associated probability 1 (such a state is said to be *absorbing*).

### 2.3.2 Stochastic Transition Matrix (in general).

Another way to represent this Markov chain is to write its transition matrix.

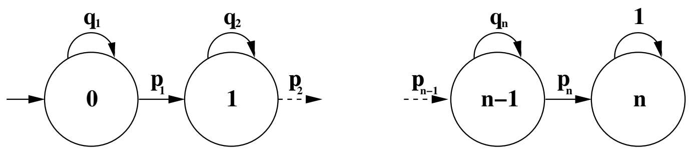


**Figure 1: State-transition diagram of the Markov Chain M.**

For a general finite Markov chain, let $p_{i,j}$ denote the transition probability from state $i$ to state $j$ for $1 \leq i, j \leq n$. The (one-step) stochastic transition matrix is:

$$P = \begin{pmatrix} p_{1,1} & p_{1,2} & \ldots & p_{1,n} \\ p_{2,1} & p_{2,2} & \ldots & p_{2,n} \\ & & \ldots & \\ p_{n,1} & p_{n,2} & \ldots & p_{n,n} \end{pmatrix}$$

THEOREM 2.1. *[6] Let $P$ be the transition matrix of a Markov chain process. Then the $m$-step transition matrix is equal to the $m$-th power of $P$. Furthermore, the entry $p_{i,j}(m)$ in $P^m$ is the probability of stepping from state $i$ to state $j$ in exactly $m$ transitions.*

### 2.3.3 Our stochastic transition matrix of interest.

For the Markov chain M defined above, the corresponding stochastic transition matrix is the following $(n+1) \times (n+1)$ square matrix:

$$M = \begin{array}{c} \text{states} \\ 0 \\ 1 \\ \vdots \\ \vdots \\ \vdots \\ n \end{array} \begin{array}{c} \begin{matrix} 0 & 1 & 2 & \ldots & n-1 & n \end{matrix} \\ \begin{pmatrix} q_1 & p_1 & 0 & \ldots & \ldots & 0 \\ 0 & q_2 & p_2 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & & \ddots & q_n & p_n \\ 0 & \ldots & \ldots & \ldots & 0 & 1 \end{pmatrix} \end{array}$$

Therefore, the probability of the $n$-gram $A_1 \to A_2 \to \cdots \to A_n$ to occur in a document of size $l$ is the probability of stepping from state 0 to state $n$ in exactly $l$ transitions. Following Theorem 2.1, this value resides at the intersection of the first row and the last column of the matrix $M^l$:

$$M^l = \begin{pmatrix} m_{1,1}(l) & m_{1,2}(l) & \ldots & \boxed{m_{1,n+1}(l)} \\ m_{2,1}(l) & m_{2,2}(l) & \ldots & m_{2,n+1}(l) \\ & & \ldots & \\ m_{n+1,1}(l) & m_{n+1,2}(l) & \ldots & m_{n+1,n+1}(l) \end{pmatrix}$$

Thus, the result we are seeking can be obtained by raising the matrix $M$ to the power $l$, and looking at the value in the upper-right corner. In terms of computational complexity, however, one must note that to multiply two $(n+1) \times (n+1)$ square matrices, we need to compute $(n+1)$ multiplications and $n$ additions to calculate each of the $(n+1)^2$ values composing the resulting matrix. To raise a matrix to the power $l$ means to repeat this operation $l-1$ times. The resulting time complexity is then $O(ln^3)$.

One may object that more time-efficient algorithms for matrix multiplication exist. The lowest exponent currently known is by Coppersmith and Winograd: $O(n^{2.376})$ [3]. Such results are achieved by studying how matrix multiplication depends on bilinear and trilinear combinations of factors. The strong drawback of those techniques is the presence of a constant factor so large that it removes the benefits of the lower exponent for all practical sizes of matrices [8]. For our purpose, the use of such an algorithm is typically more costly than to use the naive $O(n^3)$ matrix multiplication.

Linear algebra techniques, and a careful exploitation of the specificities of the stochastic matrix $M$ will, however, permit to perform a few transformations that will drastically reduce the computational complexity of $M^l$.

### 2.3.4 The Jordan normal form

**Definition:** A *Jordan block* $J_\lambda$ is a square matrix whose elements are zero except for those on the principal diagonal, which are equal to $\lambda$, and for those on the first superdiagonal, which are equal to unity. Thus:

$$J_\lambda = \begin{pmatrix} \lambda & 1 & & 0 \\ & \lambda & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda \end{pmatrix}.$$

THEOREM 2.2. *(Jordan normal form) [11] If $A$ is a general square matrix, then there exists an invertible matrix $S$ such that*

$$J = S^{-1}AS = \begin{pmatrix} J_1 & & 0 \\ & \ddots & \\ 0 & & J_k \end{pmatrix},$$

*where the $J_i$ are $n_i \times n_i$ Jordan blocks. The same eigenvalues may occur in different blocks, but the number of distinct blocks corresponding to a given eigenvalue is equal to the number of eigenvectors corresponding to that eigenvalue and forming an independent set. The number $k$ and the set of numbers $n_1, \dots, n_k$ are uniquely determined by $A$.*

In the following subsection we will demonstrate that $M$ is a matrix such that there exists only one block for each eigenvalue.

#### 2.3.4.1 Uniqueness of the Jordan block corresponding to any given eigenvalue of $M$.

THEOREM 2.3. *For the matrix $M$, no two eigenvectors corresponding to the same eigenvalue can be linearly independent.*

PROOF. Because $M$ is triangular, its characteristic polynomial is the product of the diagonals of $(\lambda I_{n+1} - M)$: $f(\lambda) = (\lambda - q_1)(\lambda - q_2)\dots(\lambda - q_n)(\lambda - 1)$. The eigenvalues of $M$ are the solutions of the equation $f(\lambda) = 0$. Therefore, they are the distinct $q_i$'s, and 1.

Now let us show that whatever the order of multiplicity of such an eigenvalue (how many times it occurs in the set $\{q_1, \dots, q_n, 1\}$), it has only one associated eigenvector. The eigenvectors associated to a given eigenvalue $e$ are defined as the non null solutions of the equation $M \cdot V = e \cdot V$. If we write the coordinates of $V$ as $[v_1, v_2, \dots, v_{n+1}]$, we can observe that $M \cdot V = e \cdot V$ results in a system of $(n+1)$ equations, where, for $1 \le j \le n$, the $j$-th equation permits to express $v_{j+1}$ in terms of $v_j$, and therefore in terms of $v_1$. That is,

$$\text{for } 1 \le j \le n: \; v_{j+1} = \frac{e - q_j}{p_j} v_j = \frac{(e - q_j)\dots(e - q_1)}{p_j \dots p_1} v_1.$$

In general (for all the $q_i$'s), $v_1$ can be chosen freely to have any non null value. This choice will uniquely determine all the values of $V$.

Since the general form of the eigenvectors corresponding to any eigenvalue of $M$ is $V = [v_1, v_2, \dots, v_{n+1}]$, where all the values can be determined uniquely by the free choice of $v_1$, it is clear that no two such eigenvectors can be linearly independent. Hence, one and only one eigenvector corresponds to each eigenvalue of $M$. $\square$

Following theorem 2.2, this means that there is a single Jordan block for each eigenvalue of $M$, whose size equals to the order of algebraic multiplicity of the eigenvalue, that is, its number of occurrences in the principal diagonal of $M$. In other words, there is a distinct Jordan block for every distinct $q_i$ (and its size equals the number of occurrences of $q_i$ in the main diagonal of $M$), plus a block of size 1 for the eigenvalue 1. Therefore we can write

$$J = S^{-1}MS = \begin{pmatrix} \boxed{J_{e_1}} & & 0 \\ & \ddots & \\ 0 & & \boxed{J_{e_q}} \end{pmatrix},$$

where the $J_{e_i}$ are $n_i \times n_i$ Jordan blocks, corresponding to the distinct eigenvalues of $M$. Following general properties of the Jordan normal form, we have

$$J^l = \begin{pmatrix} \boxed{J^l_{e_1}} & & 0 \\ & \ddots & \\ 0 & & \boxed{J^l_{e_q}} \end{pmatrix}.$$

Also,

$$\begin{aligned} M^l &= \overbrace{(SJS^{-1}) \cdot (SJS^{-1}) \dots (SJS^{-1})}^{l \text{ times}} \\ &= S \cdot J \cdot \overbrace{(S^{-1} \cdot S) \cdot J \cdot (S^{-1} \cdot S) \cdot J \dots (S^{-1} \cdot S) \cdot J}^{(l-1) \text{ times}} \cdot S^{-1} \\ &= S \cdot \overbrace{J \cdot J \dots J}^{l \text{ times}} \cdot S^{-1} \\ &= S \cdot J^l \cdot S^{-1}. \end{aligned}$$

Therefore, by multiplying the first row of $S$ by $J^l$, and multiplying the resulting vector by the last column of $S^{-1}$, we do obtain the upper right value of $M^l$, that is, the probability of the $n$-gram $(A_1 \to \dots \to A_n)$ to appear in a document of size $l$.

#### 2.3.4.2 Calculating powers of a Jordan block.

As mentioned above, to raise $J$ to the power $l$, we can simply write a direct sum of the Jordan blocks raised to the power $l$. In this section, we will show how to compute $J^l_{e_i}$ for a Jordan block $J_{e_i}$.

Let us define $D_{e_i}$ and $N_{e_i}$ such that $J_{e_i} = D_{e_i} + N_{e_i}$, where $D_{e_i}$ contains only the principal diagonal of $J_{e_i}$, and $N_{e_i}$ only its first superdiagonal. That is,

$$D_{e_i} = \begin{pmatrix} e_i & & & 0 \\ & e_i & & \\ & & \ddots & \\ 0 & & & e_i \end{pmatrix} \text{ and } N_{e_i} = \begin{pmatrix} 0 & 1 & & 0 \\ & & \ddots & \\ & & & 1 \\ 0 & & & 0 \end{pmatrix}.$$

Observing that $N_{e_i} D_{e_i} = D_{e_i} N_{e_i}$, we can use the binomial theorem:

$$J^l_{e_i} = (D_{e_i} + N_{e_i})^l = \sum_{k=0}^{l} \binom{l}{k} N^k_{e_i} D^{l-k}_{e_i}$$

Because $N_{e_i}$ is nilpotent ($N^k_{e_i} = 0, \forall k \geq n_i$), we can shorten the summation to:

$$J^l_{e_i} = (D_{e_i} + N_{e_i})^l = \sum_{k=0}^{n_i-1} \binom{l}{k} N^k_{e_i} D^{l-k}_{e_i}$$

Hence, to calculate $J^l_{e_i}$, one can compute the powers of $D_{e_i}$ and $N_{e_i}$ from 0 to $l$, which is a fairly simple task. The power of a diagonal matrix is easy to compute, as it is another diagonal matrix where each term of the original matrix is raised to the same power as the matrix. $D^j_{e_i}$ is thus identical to $D_{e_i}$, except that the main diagonal is filled with the value $e^j_i$ instead of $e_i$. To compute $N^k_{e_i}$ is even simpler. Each multiplication of a power of $N_{e_i}$ by $N_{e_i}$ results in shifting the non-null diagonal one row upwards.

The result of $N^k_{e_i} D^j_{e_i}$ resembles $N^j_{e_i}$, except that the ones on the only non-null diagonal (the $j$-th superdiagonal) are replaced by the value of the main diagonal of $D^j_{e_i}$, that is $e^j_i$. Therefore, we have:

$$N^k_{e_i} D^{l-k}_{e_i} = \begin{pmatrix} 0 & e_i^{l-k} & & 0 \\ & & \ddots & \\ & & & e_i^{l-k} \\ 0 & & & 0 \end{pmatrix}.$$

Since each value of $k$ corresponds to a distinct diagonal, the summation $\sum_{k=0}^{l} \binom{l}{k} N^k_{e_i} D^{l-k}_{e_i}$ is easily written as:

$$\begin{aligned} J^l_{e_i} &= \sum_{k=0}^{l} \binom{l}{k} N^k_{e_i} D^{l-k}_{e_i} \\ &= \begin{pmatrix} \binom{l}{0} \cdot e_i^l & \dots & \binom{l}{k} \cdot e_i^{l-k} & \dots & \binom{l}{n_i-1} \cdot e_i^{l-n_i+1} \\ & \ddots & & \ddots & \vdots \\ & & \binom{l}{0} \cdot e_i^l & & \binom{l}{k} \cdot e_i^{l-k} \\ & & & \ddots & \vdots \\ 0 & & & & \binom{l}{0} \cdot e_i^l \end{pmatrix}. \end{aligned}$$

### 2.3.5 Conclusion

The probability of the $n$-gram $(A_1 \to \dots \to A_n)$ in a document of size $l$ can be obtained as the upper-right value in the matrix $M^l$ such that:

$$M^l = S J^l S^{-1} = S \begin{pmatrix} \boxed{J^l_{e_1}} & & 0 \\ & \ddots & \\ 0 & & \boxed{J^l_{e_q}} \end{pmatrix} S^{-1},$$

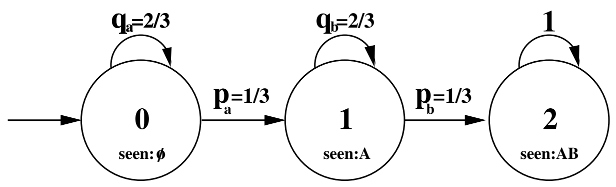


**Figure 2: State-transition diagram of the Markov Chain corresponding to our running example.**

where the $J^l_{e_i}$ blocks are as described above, while $S$ and $S^{-1}$ are obtained through the Jordan Normal Form theorem (Theorem 2.2). We actually only need the first row of $S$ and the last column of $S^{-1}$, as we are not interested in the whole matrix $M^l$ but in its upper-right value only.

In the next subsection we will calculate the worst case time complexity of the technique that we just presented. Before that, let us return to the running example presented in subsection 2.1.2.

### 2.3.6 Running Example.

The state-transition diagram of the Markov Chain corresponding to the bigram $A \to B$ has only three states (figure 2). The corresponding transition matrix is:

$$M_{re} = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & \frac{2}{3} & \frac{1}{3} \\ 0 & 0 & 1 \end{pmatrix}.$$

Following Theorem 2.2 on the Jordan normal form, there exists an invertible matrix $S_{re}$ such that

$$J_{re} = S_{re}^{-1} M_{re} S_{re} = \begin{pmatrix} \boxed{J_{\frac{2}{3}}} & 0 \\ 0 & \boxed{J_1} \end{pmatrix},$$

where $J_1$ is a block of size 1, and $J_{\frac{2}{3}}$ a block of size 2 since $q_a = q_b = \frac{2}{3}$. We can actually write $J_{re}$ as:

$$J_{re} = \begin{pmatrix} \frac{2}{3} & 1 & 0 \\ 0 & \frac{2}{3} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Since we seek the probability of the bigram $A \to B$ in a document of size 3, we need to calculate $J^3_{re}$:

$$J^3_{re} = \begin{pmatrix} \binom{3}{0}(\frac{2}{3})^3 & \binom{3}{1}(\frac{2}{3})^2 & 0 \\ 0 & \binom{3}{0}(\frac{2}{3})^3 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{8}{27} & \frac{4}{3} & 0 \\ 0 & \frac{8}{27} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

In the next subsection, we will give further details as to the practical computation of $S_{re}$ and the last column of its inverse $S^{-1}_{re}$. For now, let us simply assume they were calculated, and we can thus obtain the probability of the bigram AB in a document of length 3 as:

$$\begin{aligned} P(A \to B, 3) &= \overbrace{(1\ 0\ 1)}^{\text{1st row of } S} \begin{pmatrix} \frac{8}{27} & \frac{4}{3} & 0 \\ 0 & \frac{8}{27} & 0 \\ 0 & 0 & 1 \end{pmatrix} \overbrace{\begin{pmatrix} -1 \\ -\frac{1}{3} \\ 1 \end{pmatrix}}^{\text{last col. of } S^{-1}} \\ &= \frac{7}{27}. \end{aligned}$$

Our technique indeed obtains the right result. But how efficiently is it obtained? We overview an answer to this question in the following subsection.

## 2.4 Algorithmic Complexity

The process of calculating the probability of occurrence of an $n$-gram in a document of size $l$ consists of two main phases: calculating $J^l$, and computing the transformation matrix $S$ and its inverse $S^{-1}$.

Below, we will study the worst case time complexity, but it is interesting to observe that in practice, for a corpus big enough, the number of words equally-weighed should be small. This is especially true since, following Zipf's law, infrequent words are most likely to have equal weights, and they precisely are often pruned during preprocessing.

The following complexity analysis might be easier to follow, if studied together with the general formulas of $M^l$ and the Jordan blocks as presented in the conclusion of subsection 2.3 (subsection 2.3.5).

### 2.4.1 Time complexity of the $J^l$ calculation

Observing that each block $J_i^l$ contains exactly $n_i$ distinct values, we can see that $J^l$ contains $\sum_{1\leq k\leq q} n_k = n+1$ distinct values. Those $(n+1)$ values are $(n+1)$ multiplications of a binomial coefficient by a power of an eigenvalue.

The computation of the powers between 0 and $l$ of each eigenvalue is evidently achieved in $O(lq)$, because each of the $q$ distinct eigenvalues needs to be multiplied by itself $l$ times.

For every Jordan block $J_i^l$, the binomial coefficients to be computed are: $\binom{l}{0}, \binom{l}{1}, \ldots, \binom{l}{n_i-1}$. For the whole matrix $J^l$, we thus need to calculate $\binom{l}{k}$ where $0 \leq k \leq max_{block}$ and $\max_{block} = \max_{i=1}^{q} n_i$. Observing that $\binom{l}{j+1} = \binom{l}{j}\frac{l-j}{j+1}$, and thus, that $\binom{l}{j+1}$ can be computed from $\binom{l}{j}$ in a constant number of operations, we see that the set $\{\binom{l}{k} \mid 1 \leq k \leq \max_{block}\}$ can be computed in $O(\max_{block})$.

If $l < n$, the probability of occurrence of the $n$-gram in $l$ is 0, since the $n$-gram is longer than the document. Therefore, the current algorithm is only used when $l \geq n \geq \max_{block}$. We can therefore conclude that **the time complexity of the computation of $J^l$ is $O(lq)$.**

### 2.4.2 Calculating the transformation matrix $S$

Following general results of linear algebra [11], the $(n+1)\times(n+1)$ transformation matrix $S$ can be written as:

$$S = \left[S_1 S_2 \ldots S_q\right],$$

where each $S_i$ is an $n_i \times (n+1)$ matrix corresponding to the eigenvalue $e_i$, and such that $S_i = \left[v_{i,1} v_{i,2} \ldots v_{i,n_i}\right]$, where:

- $v_{i,1}$ is an eigenvector associated with $e_i$, thus such that, $Mv_{i,1} = e_i v_{i,1}$, and
- $v_{i,j}$, for all $j = 2 \ldots n_i$, is a solution to the equation $Mv_{i,j} = e_i v_{i,j} + v_{i,j-1}$.

The vectors $v_{i,1}v_{i,2}\ldots v_{i,n_i}$ are sometimes called *generalized eigenvectors* of $e_i$. We have already seen in section 2.3.4, that the first coordinate of each eigenvector can be assigned freely, and that every other coordinate can be expressed in function of its immediately preceding coordinate. Therefore it takes a constant number of operations to calculate the value of each coordinate of an eigenvector, and each eigenvector can be computed in $O(n)$. It is equally provable that the generalized eigenvectors can be expressed and calculated in a constant number of operations. Following this fact, each column of $S$ can be computed in $O(n)$, and thus **the whole matrix $S$ in $O(n^2)$.**

### 2.4.3 The inversion of $S$

The general inversion of an $(n+1)\times(n+1)$ matrix can be done in $O(n^3)$ through Gaussian elimination. To calculate only the last column of $S^{-1}$ does not help, since the resulting system of $(n+1)$ equations still requires $O(n^3)$ operations to be solved by Gaussian elimination.

However, some specificities of our problem will again permit an improvement over this general complexity. When calculating the similarity matrix $S$, we can ensure that $S$ **is a column permutation of an upper-triangular matrix**. It is therefore possible to calculate the last column of $S^{-1}$ by solving a triangular system of linear equations through a backward substitution mechanism. **The computation of the last column of $S^{-1}$ is thus achieved in** $O(n^2)$.

### 2.4.4 Conclusion

To obtain the final result, the probability of occurrence of the $n$-gram in a document of size $l$, it remains to multiply the first row of $S$ by $J^l$, and the resulting vector by the last column of $S^{-1}$. The second operation takes $(n+1)$ multiplications and $n$ additions. It is thus $O(n)$.

The general multiplication of a vector of size $(n+1)$ by an $(n+1)\times(n+1)$ square matrix takes $(n+1)$ multiplications and $n$ additions for each of the $(n+1)$ values of the resulting vector. This is thus $O(n^2)$. However, we can use yet another trick to improve this complexity. When we calculated the matrix $S$, we could assign the first row values of each column vector freely. We did it in such a way that the only non-null values on the first row of $S$ are unity, and that they occur on the $q$ eigenvector columns (these same choices ensured that $S$ is a column permutation of an upper-triangular matrix). Therefore, to multiply the first row of $S$ by a column vector simply consists in the addition of the $q$ terms of index equal to the index of the eigenvectors in $S$. That operation of order $O(q)$ needs to be repeated for each column of $J^l$. The multiplication of the first row of $S$ by $J^l$ is thus $O(nq)$.

The worst case time complexity of the computation of the probability of occurrence of an $n$-gram in a document of size $l$ is finally $\max\{O(lq), O(n^2)\}$. Since our problem of interest is limited to $l \geq n$ (otherwise the probability of occurrence is 0), an **upper bound** of the complexity for computing **the probability of occurrence of an $n$-gram in a document of size $l$ is** $O(ln)$. This is clearly better than directly raising $M$ to the power $l$, which is $O(ln^3)$.

## 3. EXPECTED DOCUMENT FREQUENCY OF A WORD SEQUENCE

Now that we have defined a formula to calculate the probability of occurrence of an $n$-gram in a document of size $l$, we can use it to calculate the expected document frequency of the $n$-gram in the whole document collection $D$. Assuming the documents are mutually independent, the expected frequency in the document collection is the sum of the probabilities of occurrence in each document:

$$Exp\_df(A_1 \to \ldots A_n, D) = \sum_{d\in D} p(A_1 \to \ldots A_n, |d|),$$

where $|d|$ stands for the number of word occurrences in the document $d$.

**Naive Computational Complexity.** We can compute the probability of an $n$-gram to occur in a document in

$O(ln)$. A separate computation and summation of the values for each document can thus be computed in $O(|D|ln)$, where $|D|$ stands for the number of documents in $D$.

In practice, we can improve the computational efficiency by counting the number of documents of same length and multiplying this number by the probability of occurrence of the $n$-gram in a document of that size, rather than reprocessing and summing up the same probability for each document of equal size. But as we are currently considering the *worst case* time complexity of the algorithm, we are facing the *worst case* situation in which every document has a distinct length.

**Better Computational Complexity.** We can achieve better complexity by summarizing everything we need to calculate and organizing the computation in a sensible way. Let $L = \max_{d \in D} |d|$ be the size of the longest document in the collection. We first need to raise the Jordan matrix $J$ to the power of every distinct document length, and then to multiply the (at worst) $|D|$ distinct matrices by the first row of $S$ and the resulting vectors by the last column of its inverse $S^{-1}$.

The matrix $S$ and the last column of $S^{-1}$ need to be computed only once, and as we have seen previously, this is achieved in $O(n^2)$, whereas the $|D|$ multiplications by the first row of $S$ are done in $O(|D|nq)$. It now remains to find the computational complexity of the various powers of $J$.

We must first raise each eigenvalue $e_i$ to the power $L$, which is an $O(Lq)$ process. For each document $d \in D$, we obtain all the terms of $J^{|d|}$ by $(n+1)$ multiplications of powers of eigenvalues by a set of combinatorial coefficients computed in $O(\max_{block})$. The total number of such multiplications is thus $O(|D|n)$, an upper bound for the computation of all combinatorial coefficients. The worst case time complexity for computing the set $\{ J^{|d|} \mid d \in D\}$, is thus $\max\{O(|D|n), O(Lq)\}$.

Finally, **the computational complexity for calculating the expected frequency of an $n$-gram in a document collection $D$ is** $\max\{O(|D|nq), O(Lq)\}$, where $q$ is the number of words in the $n$-gram having a distinct probability of occurrence, and $L$ is the size of the longest document in the collection. The improvement is considerable compared to the naive technique's $O(|D|ln^3)$.

## 4. DIRECT EVALUATION OF LEXICAL COHESIVE RELATIONS

The evaluation of lexical cohesion is a difficult problem. Attempts of direct evaluation are rare, simply due to the subjectivity of any human assessment and due to the wide acceptance that we first need to know what we want to do with a lexical unit before being able to decide whether or not it is relevant for that purpose. A common application of research in lexical cohesion is lexicography, where the evaluation is carried out by human experts who simply look at phrases to assess them as good or bad. This process permits to score the extraction process with highly subjective measures of precision and recall. However, a linguist interested in the different forms and uses of the auxiliary "to be" will have a different view of what is an interesting phrase than a lexicographer. What a human expert judges as uninteresting may be highly relevant to another.

Hence, most evaluation has been indirect, through question answering, topic segmentation, text summarization, and passage or document retrieval [12]. To pick the last case, such an evaluation consists in trying to figure out which are the phrases that permit to improve the relevance of the list of documents returned. A weakness of indirect evaluation is that it hardly shows whether an improvement is due to the quality of the phrases, or to the quality of the technique used to exploit them.

There is a need to fill the lack of a general purpose direct evaluation technique, one where no subjectivity or knowledge of the domain of application will interfere. Our technique permits exactly that, and this section will show how.

## 4.1 Hypothesis testing

A general approach to estimate the interestingness of a set of events is to measure their statistical significance. In other words, by evaluating the validity of the assumption that an event occurs only by chance (the *null hypothesis*), we can decide whether the occurrence of that event is interesting or not. If a frequent occurrence of a multi-word unit was to be expected, it is less interesting than if it comes as a surprise.

To estimate the quality of the assumption that an $n$-gram occurs by chance, we need to compare its (by chance) expected frequency and its observed frequency. There exists a number of statistical tests, extensively described in statistics textbooks, even so in the specific context of natural language processing [9]. In this paper, we will base our experiments on the *t-test*:

$$t = \frac{Obs\_df(A_1 \to \ldots A_n, D) - Exp\_df(A_1 \to \ldots A_n, D)}{\sqrt{|D| Obs\_DF(A_1 \to \ldots A_n)}}$$

## 4.2 Example of non-contiguous lexical units: MFS

Maximal Frequent Sequences (MFS) [1] are word sequences built with an unlimited gap, no stoplist, no POS analysis and no linguistic filtering. They are defined by two characteristics:

- A sequence is said to be *frequent* if its document frequency is higher than a given threshold.
- A sequence is *maximal*, if there exists no other frequent sequence that contains it.

Thus, MFSs correspond very well to our technique, since the extraction algorithm provides each extracted MFS with its document frequency. To compare the observed frequency of MFSs to their expected frequency is thus especially meaningful, and it will permit to rank a set of MFSs with respect to their statistical significance.

## 4.3 Experiments

### 4.3.1 Corpus

For experiments we used the publicly available Reuters-21578 newswire collection [1], which originally contains about 19,000 non-empty documents. We split the data into 106,325 sentences. The average size of a sentence is 26 word occurrences, while the longest sentence contains 260.

Using a minimum frequency threshold of 10, we extracted 4,855 MFSs, distributed in 4,038 2-grams, 604 3-grams, 141 4-grams, and so on. The longest sequences had 10 words.

[1]http://www.daviddlewis.com/resources/textcollections/reuters21578

| t-test | $n$-gram | expected | observed |
|---|---|---|---|
| 0.03109 | los angeles | 0.08085 | 103 |
| 0.02824 | kiichi miyazawa | 0.09455 | 85 |
| 0.02741 | kidder peabody | 0.04997 | 80 |
| 0.02666 | morgan guaranty | 0.20726 | 76 |
| 0.02485 | latin america | 0.65666 | 67 |

**Table 1: Overall Best 5 MFSs**

| t-test | $n$-gram | expected | observed |
|---|---|---|---|
| 9.6973-03 | het comite | 0.6430-03 | 10 |
| 9.6972-03 | piper jaffray | 0.8184-03 | 10 |
| 9.6969-03 | wildlife refuge | 0.0522-03 | 10 |
| 9.6968-03 | tate lyle | 0.1458-03 | 10 |
| 9.6968-03 | g.d searle | 0.1691-03 | 10 |
| 8.2981-03 | pacific security | 1.4434 | 10 |
| 8.2896-03 | present intervention | 1.4521 | 10 |
| 8.2868-03 | go go | 1.4551 | 10 |
| 8.2585-03 | bills holdings | 1.4843 | 10 |
| 8.2105-03 | cents barrel | 1.5337 | 10 |

**Table 2: The t-test applied to the 5 best and worst bigrams of frequency 10**

The expected document frequency and the $t$-test of all the MFSs were computed in 31.425 seconds on a laptop with a 1.40 Ghz processor and 512Mb of RAM. We used an implementation of a simplified version of the algorithm that does not make use of all the improvements presented in this paper.

### 4.3.2 *Results*

Table 1 shows the overall best-ranked MFSs. In Table 2, we can compare the best-ranked bigrams of frequency 10 to their last-ranked counterparts, noticing a difference in quality that observed frequency alone does not reveal.

It is important to note that our technique permits to rank longer $n$-grams amongst pairs. For example, the best-ranked $n$-gram of size higher than 2 lies in the $10^{th}$ position: *"chancellor exchequer nigel lawson"* with $t$-test value 0.023153, observed frequency 57, and expected frequency $0.2052e-07$.

In contrast to this high-ranked 4-gram, the last-ranked $n$-gram of size 4 occupies the $3,508^{th}$ position: *"issuing indicated par europe"* with $t$-test value 0.009698, observed frequency 10, and expected frequency $22.25e-07$.

## 5. CONCLUSION

We presented a novel technique for calculating the probability and expected document frequency of any given non-contiguous lexical cohesive relation. We found a Markov representation for the problem and exploited the specificities of that representation to obtain a low computational complexity.

We further described a method that compares observed and expected document frequencies through a statistical test as a way to give a direct numerical evaluation of the intrinsic quality of a multi-word unit (or of a set of multi-word units). This technique does not require work of a human expert, and it is fully language- and application-independent.

It is generally accepted that, in English, two words at a distance five or more are not connected. We can attempt to deal with this by using short documents, for example sentences, or even comma-separated units.

A weakness that our approach shares with most language models is the assumption that terms occur independently from each other. In the future, we hope to present more advanced Markov representations that will permit to account for term dependency.

## 6. REFERENCES

[1] H. Ahonen-Myka and A. Doucet. Data mining meets collocations discovery. In *print*, pages 1–10. CSLI Publications, Center for the Study of Language and Information, University of Stanford, 2005.

[2] Y. Choueka, T. Klein, and E. Neuwitz. Automatic retrieval of frequent idiomatic and collocational expressions in a large corpus. *Journal for Literary and Linguistic computing*, 4:34–38, 1983.

[3] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 1–6, 1987.

[4] G. Dias. Multiword unit hybrid extraction. In *Workshop on Multiword Expressions of the 41st ACL meeting. Sapporo. Japan.*, 2003.

[5] G. Dias, S. Guilloré, J.-C. Bassano, and J. G. P. Lopes. Extraction automatique d'unités complexes: Un enjeu fondamental pour la recherche documentaire. *Traitement Automatique des Langues*, 41(2):447–472, 2000.

[6] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley Publications, third edition, 1968.

[7] K. T. Frantzi, S. Ananiadou, and J. ichi Tsujii. The c-value/nc-value method of automatic recognition for multi-word terms. In *ECDL '98: Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries*, pages 585–604. Springer-Verlag, 1998.

[8] R. A. Horn and C. R. Johnson. *Topics in matrix analysis.* Cambridge University Press, New York, NY, USA, 1994.

[9] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge MA, second edition, 1999.

[10] M. Mitra, C. Buckley, A. Singhal, and C. Cardie. An analysis of statistical and syntactic phrases. In *Proceedings of RIAO97, Computer-Assisted Information Searching on the Internet*, pages 200–214, 1987.

[11] B. Noble and J. W. Daniel. *Applied Linear Algebra*, pages 361–367. Prentice Hall, second edition, 1977.

[12] V. O. The role of multi-word units in interactive information retrieval. In *Proceedings of the 27th European Conference on Information Retrieval, Santiago de Compostela, Spain*, pages 403–420, 2005.

[13] F. Smadja. Retrieving collocations from text: Xtract. *Journal of Computational Linguistics*, 19:143–177, 1993.

[14] T. Tanaka, A. Villavicencio, F. Bond, and A. Korhonen, editors. *Second ACL Workshop on Multiword Expressions: Integrating Processing*, 2004.

# Unsupervised Topic Segmentation Based on Word Co-occurrence and Multi-Word Units for Text Summarization

Gaël Dias
Centre of Human Language Technology and Bioinformatics
University of Beira Interior
+351 275 319 891
ddg@di.ubi.pt

Elsa Alves
Natural Language Research Group
Department of Computer Science
New University of Lisbon
+351 21 294 85 36
elsalves@zmail.pt

## ABSTRACT

Topic Segmentation is the task of breaking documents into topically coherent multi-paragraph subparts. In particular, Topic Segmentation is extensively used in Passage Retrieval and Text Summarization to provide more coherent results by taking into account raw document structure. However, most methodologies are based on lexical repetition that show evident reliability problems or rely on harvesting linguistic resources that are usually available only for dominating languages and do not apply to less favored and emerging languages. Moreover, most systems have been evaluated using Choi's data set [1] which is biased for systems using mostly lexical repetition. As a consequence, these systems are not tested in real-world environments and their application may prove worst results than presented in the literature. In order to tackle all these drawbacks, we present an innovative Topic Segmentation system based on a new informative similarity measure based on word co-occurrences and evaluate it on a set of web documents within which Multiword Units have previously been identified.

## Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – *abstracting methods.*

## General Terms

Algorithms, Measurement, Experimentation.

## Keywords

Unsupervised Topic Segmentation, Evaluation on Single Domain Web Documents, Text Summarization, Passage Retrieval.

## 1. INTRODUCTION

This paper introduces a new technique for improving access to information dividing lengthy documents into topically coherent sections. This research area is commonly called Topic Segmentation and can be defined as the task of breaking documents into topically coherent multi-paragraph subparts.

In order to provide solutions to access useful information from the ever-growing number of documents on the web, such technologies are crucial as people who search for information are now submerged with unmanageable quantities of texts.

For that purpose, Topic Segmentation has extensively been used in Information Retrieval and Text Summarization. In the context of Information Retrieval, it is clear that some user should prefer a document in which the occurrences of a word are concentrated into one or two paragraphs since such a concentration is more likely to contain a definition of the queried concept and as a consequence the system is more likely to retrieve useful information. This particular research domain is usually called Passage Retrieval and proposes techniques to extract fragments of texts relevant to a query [2][3][4]. In the context of Text Summarization, Topic Segmentation is usually used as the basic text structure in order to apply sentence extraction and sentence compression techniques [5][6][7].

In this paper, we present an innovative Topic Segmentation system based on a new informative similarity measure that takes into account word co-occurrence in order to avoid the accessibility to existing linguistic resources such as electronic dictionaries or lexico-semantic databases. In particular, our architecture solves three main problems evidenced by previous research. First, systems based uniquely on lexical repetition show reliability problems [8][9][10][11][12] as common writing rules prevent from using lexical repetition. Second, systems based on lexical cohesion, using existing linguistic resources that are usually only available for dominating languages like English, French or German, do not apply to less favored and emerging languages [13][14]. Third, systems that need previously existing harvesting training data [15] do not adapt easily to new domains as training data is usually difficult to find or build depending on the domain being tackled. Instead, our architecture proposes a language-independent unsupervised solution, similar to [16][17], defending that Topic Segmentation should be done "on the fly" on any text thus avoiding the problems of domain/genre/language-dependent systems that need to be tuned each time one of these parameters changes (domain, genre or language).

In order to show the results of our system in real-world conditions, we propose two different evaluations on a set of web documents: (1) one based only on words and (2) one based on the set of documents within which multiword units have previously been identified "on the fly". Unlike other methodology that have been evaluated on Choi's data set [1] which relies on small texts of different domains within which lexical repetition is high, we propose an evaluation on real-world texts where lexical distribution does not overuse repetition. In particular, we show

that the introduction of semantic information into the set of documents such as Multiword Units leads to better results.
This paper is divided into five sections. First, we show the main differences between our work and the existing ones, in particular the systems proposed by [16] and [17]. Second, we show the weighting process of each word of the input text corpus. Third, we introduce our main innovation i.e. the informative similarity measure. Fourth, we define how subparts can be elected from the values of the informative similarity measure. And fifth, we propose an evaluation on a real-world situation using "on the fly" identification of Multiword Units.

## 2. RELATED WORK

[8], [9] and [12] have proposed different architectures based on lexical item[1] repetition: respectively, TextTiling, Dotplotting and the Link Set Median Procedure. However, it has been proved that systems based on lexical repetition are not reliable when applied to non-technical texts without small controlled vocabularies. For instance, articles in newspapers tend to avoid word repetition. In fact, good writing should avoid word repetition. As a consequence, these techniques can only be applied to technical texts where synonyms rarely exist for a given concept so that word repetition is almost compulsory.
In order to avoid these limitations, [14] has proposed an architecture based on a Semantic Network built from the English Dictionary (LDOCE) from which lexical cohesion can be fine-grained induced. First, [13] had proposed a discourse segmentation algorithm based on lexical cohesion relations called lexical chains using Roget's thesaurus. However, these linguistic resources are not available for the majority of languages so that their application is drastically limited and as a consequence do not apply to less favored and emerging languages.
In order to avoid the use of huge linguistic resources, [15] have proposed a technique for identifying document boundaries using statistical techniques. So, they built statistical models within a framework which incorporated a number of cues about the story boundaries such as the appearance of particular words before a boundary and the appearance of cue words in the beginning of the previous sentence of a boundary. Unfortunately, this work is limited by the need of previously existing harvesting training data as it proposes a supervised solution to the problem of Topic Segmentation. Once more, it lacks in flexibility as new training is necessary when the genre/domain/language change.
It is clear that unsupervised language-independent techniques that automatically induce some degree of semantics propose a promising solution to solve all the exposed problems. [16] and [17] have proposed such techniques. [16] proposes to identify a lexical network based on word collocation frequency statistics and cluster analysis. However, he does not propose a classical Topic Segmentation technique but rather a Topic Detection system as he does not output boundaries in the text. [17] propose a Topic Segmentation technique based on the Local Content Analysis [18] allowing to substituting each sentence with words and phrases related to it. A pairwise similarity measure is then calculated between all transformed sentences and then introduced into a final score in order to find at each point in the corpus the best block that maximizes the score function. The important point to focus on is the use of the Local Content Analysis that introduces some degree of semantics to the system without requiring harvesting linguistic resources and thus reducing the problem of word repetition. In order to introduce endogenously acquired semantic knowledge, [19] has also proposed to automatically extract collocations from texts in order to compute semantic similarity measures[2].
Although our approach tends to stand to the basic ideas of these unsupervised methodologies, we differ from them as we clearly pose the problem of word weighting for the specific task of Topic Segmentation. Indeed, most of the presented systems only rely on frequency and/or the *tf.idf* measure proposed by [20][21] of their lexical items. However, we deeply think that better weighting measures can be proposed. For that purpose, we introduce a new weighting score based on three heuristics: the well-known *tf.idf* measure, the adaptation of the *tf.idf* measure for sentences, the *tf.isf*, and a new density measure that calculates the density of each word in the text. Moreover, in order to introduce a certain degree of semantics in our system, we propose a new informative similarity measure that includes in its definition the Equivalence Index Association Measure proposed by [22] so that word co-occurrence information is directly embedded in the calculation of the similarity between blocks of sentences. Thus, unlike [17], we propose a well-founded mathematical model that deals with the word co-occurrence factor. Finally, like classical methodologies, our system then calculates the similarity of each sentence in the corpus with the previous block of *k* sentences and the next block of *k* sentences and then elects the best text boundaries based on the standard deviation algorithm proposed by [8].

## 3. WEIGHTING SCORE

Our algorithm is based on the vector space model which determines the similarity of neighboring groups of sentences and places subtopic boundaries between dissimilar blocks. In our specific case, each sentence in the corpus is evaluated in terms of similarity with the previous block of *k* sentences and the next block of *k* sentences.
The simplest form of the vector space model treats a document (in our case, a sentence or a group of sentences) as a vector whose values correspond to the number of occurrences of the words appearing in the document as in [8]. Although [8] showed successful results with this weighting scheme, we strongly believe that the importance of a word in a document does not only depend on its frequency. Indeed, frequency can only be reliable for technical texts where ambiguity is drastically limited and word repetition largely used. But unfortunately, these documents are an exception in the global environment of the internet for example.
According to us, two main factors must be taken into account to define the relevance of a word for the specific task of Topic Segmentation: its semantic importance, based on its frequency but also on its inverse document frequency (*idf*) [20][21] and its distribution across the text. For that purpose, we propose a new weighting scheme based on three heuristics: the well-known *tf.idf* measure, the adaptation of the *tf.idf* measure for sentences, the *tf.isf*, and a new density measure that calculates the density of each word in the text.

[1] A lexical item can be a sequence of characters, a stem, a morphological root, a word or an ngram.

[2] We will show in our final section that this methodology proves to lead to encouraging results.

## 3.1 The *tf.idf* Score

The basic idea of the *tf.idf* score [21] is to evaluate the importance of a word within a document based on its frequency (i.e. frequent words within a document may reflect its meaning more strongly than words that occur less frequently) and its distribution across a collection of documents (i.e. terms that are limited to a few documents are useful for discriminating those documents from the rest of the collection). The *tf.idf* score is defined in equation 1 where *w* is a word and *d* a document.

$$tf.idf(w,d) = \frac{tf(w;d)}{|d|} \times \log_2 \frac{N}{df(w)} \quad \textbf{(1)}$$

For each *w* in document *d*, we compute its relative term frequency, i.e. the number of occurrences of *w* in *d*, *tf(w; d)*, divided by the number of words in *d*, |*d*|. We then compute the inverse document frequency of *w* [20] by taking the $\log_2$ of the ratio of *N*, the number of documents in our experiment, to the document frequency of *w*, i.e. the number of documents in which the word *w* occurs (*df(w)*).

However, not all relevant words in a document are useful for Topic Segmentation. For instance, relevant words appearing in all sentences will be of no help to segment the text into topics. For that purpose, we extend the idea of the *tf.idf* to sentences.

## 3.2 The *tf.isf* Score

The basic idea of the *tf.isf* score is to evaluate each word in terms of its distribution over the document. Indeed, it is obvious that words occurring in many sentences within a document may not be useful for Topic Segmentation purposes. So, we will define the *tf.isf* to evaluate the importance of a word within a document based on its frequency within a given sentence and its distribution across all the sentences within the document. The *tf.isf* score is defined in equation 2 where *w* is a word and *s* a sentence.

$$tf.isf(w,s) = \frac{stf(w;s)}{|s|} \times \log_2 \frac{Ns}{sf(w)} \quad \textbf{(2)}$$

For each *w* in *s*, we compute its relative sentence term frequency, that is the number of occurrences of *w* in *s*, s*tf(w; s)*, divided by the number of words in *s*, |*s*|. We then compute the inverse sentence frequency of *w* by taking the $\log_2$ of the ratio of *Ns*, the number of sentences within the document, to the sentence frequency of *w*, i.e. the number of sentences in which the word *w* occurs (*sf(w)*). As a result, a word occurring in all sentences of the document will have an inverse sentence frequency 0 giving it no chance to be a relevant word for Topic Segmentation. On the opposite, a word which occurs very often in one sentence but in very few other sentences will have a high inverse sentence frequency as well as a high sentence term frequency and thus a high *tf.isf* score. Consequently, it will be a strong candidate for being a relevant word within the document for the specific task of Topic Segmentation.

However, we can push even further our idea of word distribution. Indeed, a word *w* occurring 3 times in 3 different sentences may not have the same importance in all cases. Let's exemplify. If the 3 sentences are consecutive, the word *w* will have a strong influence on what is said in this specific region of the text. On the opposite, it will not be the case if the word *w* occurs in the first sentence, in the middle sentence and then in the last sentence. It is clear that we must take into account this phenomenon. For that purpose, we propose a new density measure that calculates the density of each word in a document.

## 3.3 The Word Density Score

The basic idea of the word density measure is to evaluate the dispersion of a word within a document. So, very disperse words will not be as relevant as dense words. In order to evaluate the word density, we propose a new measure based on the distance of all consecutive occurrences of the word in the document. We call this measure *dens* and is defined in equation 3.

$$dens(w,d) = \sum_{k=1}^{|w|-1} \frac{1}{\ln\big(dist(occur(k), occur(k+1)) + e\big)} \quad \textbf{(3)}$$

For any given word *w*, its density *dens(w,d)* in document d, is calculated from all the distances between all its occurrences, |*w*|. So, *occur(k)* and *occur(k+1)* respectively represent the positions in the text of two consecutive occurrences of the word *w* and *dist(occur(k), occur(k+1))* calculates the distance that separates them in terms of words within the document. Thus, by summing their inverse distances, we get a density function that gives higher scores to highly dense words. As a result, a word, the occurrences of which appear close to one another, will show small distances and as a result a high density. On the opposite, a word, the occurrences of which appear far from each other, will show high distances and as a result a small word density.

## 3.4 The Weighting Score

The weighting score of any word in a document can be directly derived from the previous three heuristics. As a matter of fact, by combining these three scores, we deal with the two main factors that must be taken into account to define the relevance of a word for the specific task of Topic Segmentation: its semantic importance and its distribution across the document. A straightforward definition of the weighting score is given in equation 4 where each score is normalized so that they can be combined.

$$weight(w,d) = \|tf.idf(w,d)\| \times \|tf.isf(w,s)\| \times \|dens(w,d)\| \quad \textbf{(4)}$$

The next step of the application of the vector space model aims at determining the similarity of neighboring groups of sentences. For that purpose, it is important to define an appropriate similarity measure. That is the objective of our next section.

## 4. SIMILARITY BETWEEN SENTENCES

There are a number of ways to compute the similarity between two documents, in our case, between a sentence and a group of sentences. Theoretically, a similarity measure can be defined as follows. Suppose that $Xi = (X_{i1}, X_{i2}, X_{i3}, \ldots, X_{ip})$ is a row vector of observations on *p* variables associated with a label *i*. The similarity between two units *i* and *j* is defined as $S_{ij} = f(X_i, X_j)$ where *f* is some function of the observed values. In the context of our work, the application of a similarity measure is straightforward. Indeed, $X_i$ may be regarded as the focus sentence and $X_j$ as a specific block of *k* sentences, each one being represented as *p*-dimension vectors, where *p* is the number of different words within the document and where $X_{ib}$ may represent the weighting score of the bth word in the document also appearing in the focus sentence $X_i$. Our goal here is to find the appropriate *f* function that will accurately evaluate the similarity between the focus sentence and the blocks of *k* sentences. Most applications in Natural Language Processing have used the cosine similarity measure. However, we will show that it evidences problems, like all other similarity measures proposed so far.

## 4.1 The Drawback of Similarity Measures

The cosine similarity (Equation 5) determines the angle between the vectors associated to two documents (in our case, the focus sentence and a group of *k* sentences). However, when applying the cosine similarity between two documents, only the identical indexes of the row vectors $X_i$ and $X_j$ will be taken into account i.e. if both documents do not have words in common, they will not be similar at all and will receive a cosine value of 0. However, this is not tolerable. Indeed, it is clear that both sentences (1) and (2) are similar although they do not share any word in common:

(1) *Ronaldo defeated the goalkeeper once more*.
(2) *Real Madrid striker scored again*.

The most interesting idea to avoid word repetition problems is certainly to identify lexical cohesion relationships between words.

$$S_{ij} = \cos(X_i, X_j) = \frac{\sum_{k=1}^{p} X_{ik} \times X_{jk}}{\sqrt{\sum_{k=1}^{p} X_{ik}^2} \times \sqrt{\sum_{k=1}^{p} X_{jk}^2}} \quad \textbf{(5)}$$

Indeed, systems should take into account semantic information that could, for instance, relate *Ronaldo* to *Real Madrid striker*. For that purpose, many authors have proposed to computationally identify these relationships (in particular, the synonym relation) using large linguistic resources such as Wordnet [6][23], Roget's thesaurus [13] or LDOCE [14]. However, these huge resources are only available for dominating languages and as a consequence do not apply to less favored languages.

## 4.2 The Informative Similarity Measure

A much more interesting research direction is proposed by [17] that propose a Topic Segmentation technique based on the Local Content Analysis [18], allowing substituting each sentence with words and phrases related to it. Our methodology is based on this same idea but differs from it as the word co-occurrence information is directly embedded in the calculation of the similarity between blocks of sentences thus avoiding an extra-step in the topic boundaries discovery. Another direct contribution is that, unlike [17], we propose a well-founded mathematical model that deals with the word co-occurrence factor. For that purpose, we propose a new informative similarity measure that includes in its definition the Equivalence Index Association Measure (*EI*) proposed by [22] that has shown successful results in our different research works [24] [25]. It is defined in equation 6.

$$EI(w_1, w_2) = p(w_1 \mid w_2) \times p(w_2 \mid w_1) = \frac{f(w_1, w_2)^2}{f(w_1) \times f(w_2)} \quad \textbf{(6)}$$

The Equivalence Index between words $w_1$ and $w_2$ is calculated within a word-context window in order to determine the frequency between $w_1$ and $w_2$ ($f(w_1, w_2)$) and from a collection of documents so that we can evaluate the degree of cohesiveness between two words outside the context of the document. This collection can be thought as the overall web, from which we are able to infer with maximum reliability the "true" co-occurrence between two words as it is done in [24].

So, the basic idea of our informative similarity measure is to integrate into the cosine measure the word co-occurrence factor inferred from a collection of documents with the Equivalence Index association measure. This can be done straightforwardly as defined in equation 7 where $EI(W_{ik}, W_{jl})$ is the Equivalence Index value between $W_{ik}$, the word that indexes the vector of the document *i* at position *k*, and $W_{jl}$, the word that indexes the vector of the document *j* at position *l*. In fact, the informative similarity measure can simply be explained as follows. Let's take the focus sentence $X_i$ and a block of sentences $X_j$. For each word in the focus sentence, then for each word in the block of sentences, we calculate the product of their weights and then multiply it by the degree of cohesiveness existing between those two words calculated by the *EI*. As a result, the more relevant the words will be and the more cohesive they will be, the more they will contribute for the cohesion within the text and will not contribute for a topic shift.

$$S_{ij} = \text{infosimba}(X_i, X_j) = \frac{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{ik} \times X_{jl} \times EI(W_{ik}, W_{jl})}{\sqrt{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{ik} \times X_{il} \times EI(W_{ik}, W_{il})} \times \sqrt{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{jk} \times X_{jl} \times EI(W_{jk}, W_{jl})}} \quad \textbf{(7)}$$

The next step of the application aims at placing subtopic boundaries between dissimilar blocks. For that purpose, we propose a detection methodology based on the standard deviation algorithm proposed by [8].

## 5. TOPIC BOUNDARY DETECTION

Different methodologies have been proposed to place subtopic boundaries between dissimilar blocks depending on the models used to determine similarity between blocks of sentences [8] [14] [15][17][26]. In fact, it is difficult to judge any methodology as they differ depending on the research approach. For that purpose, we propose a new methodology based on ideas expressed by different research. Taking as reference the idea of [17] who take into account the preceding and the following contexts of a segment, we calculate the informative similarity of each sentence in the corpus with its surrounding pieces of texts i.e. its previous block of *k* sentences and its next block of *k* sentences. The basic idea is to know whether the focus sentence is more similar to the preceding block of sentences or to the following block of sentences. In order to evaluate this preference in an elegant way, we propose a score for each sentence in the text in the same way [15] compare short and long-range models. Our preference score (*ps*) is defined in equation 8.

$$ps(S_i) = \log_2 \frac{\text{infosimba}(S_i, X_{i-1})}{\text{infosimba}(S_i, X_{i+1})} \quad \textbf{(8)}$$

So, if $ps(S_i)$ is positive, it means that the focus sentence $S_i$ is more similar to the previous block of sentences, $X_{i-1}$. Conversely, if $ps(S_i)$ is negative, it means that the focus sentence $S_i$ is more similar to the following block of sentences, $X_{i+1}$. In particular, when $ps(S_i)$ is near 0, it means that the focus sentence $X_i$ is similar to both blocks and so we may be in the continuity of a topic. In order to illustrate the variations of the *ps* score, we show, in Figure 1, an experiment made with five texts taken from the web with five different topics.

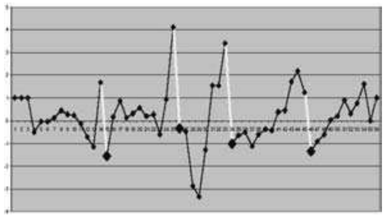


**Figure 1: Preference score variation**

In order to better understand the variation of the *ps* score, each time its value goes from positive to negative between two consecutive sentences, there exits a topic shift. We will call this phenomenon a downhill. In fact, it means that the previous sentence is more similar to the preceding block of sentences and the following sentence is more similar to the following block of sentences thus representing a shift in topic in the text. However, not all downhills identify the presence of a new topic in the text. Indeed, only deeper ones must be taken into account. They are represented in white in Figure 1 and represent the correct changes in topic. In order to automatically identify these downhills, and as a consequence the topic shifts, we adapt the algorithm proposed by [8] to our specific case. So, we propose a threshold that is a function of the average and the standard deviation of the downhills depths. A downhill is simply defined in equation 9 whenever the value of the *ps* score goes from positive to negative between two consecutive sentences $S_i$ and $S_{i+1}$.

$$downhill\left(S_i, S_{i+1}\right) = ps\left(S_i\right) - ps\left(S_{i+1}\right) \quad \mathbf{(9)}$$

Once all downhills in the text have been calculated, their mean $\bar{x}$ and standard deviation $\sigma$ are evaluated. The topic boundaries are then elected if they satisfy the constraint expressed in equation 10 where *c* is a constant to be tuned.

$$downhill\left(S_i, S_{i+1}\right) \geq \bar{x} + c\sigma \quad \mathbf{(10)}$$

By applying this threshold, we obtain promising results for the discovery of topic boundaries for the specific case of web news segmentation. We illustrate these results in the next section.

## 6. RESULTS

Topic Segmentation systems [19][27][28] have usually been evaluated on [1]'s data set that represents the gold standard for evaluation. However, many authors have discussed the validity of this test corpus [19][23][27][28] and proposed their own test corpus. Indeed, [1]'s data set, also called c99, evidences two major drawbacks: (1) it deals with segments of different domains and (2) lexical repetition is high within each segment. We propose an illustration of the c99 corpus in Figure 2.

```
The next question is whether board members favor their own social classes in their roles as educational policy-makers. On the whole, it appears that they do not favor their own social classes in an explicit way. Seldom is there an issue in which class lines can be clearly drawn. A hypothetical issue of this sort might deal with the establishment of a free public junior college in a community where there already was a good private college which served the middle-class youth adequately but was too expensive for working-class youth. In situations of this sort the board generally favors the expansion of free education.
```

```
Vincent G. Ierulli has been appointed temporary assistant district attorney, it was announced Monday by Charles E. Raymond, District Attorney. Ierulli will replace Desmond D. Connall who has been called to active military service but is expected back on the job by March 31. Ierulli, 29, has been practicing in Portland since November, 1959.
```

**Figure 2: Example of the C99 corpus (Directory 3-5, Text 7)**

However, it is clear that the c99 corpus does not apply for an evaluation oriented towards Text Summarization. Indeed, in this case, the texts must cover a single domain and intra-segment lexical repetitions are not used as much as in the c99 corpus. However, it is likely that there exist inter-segment lexical repetitions which unease the process of boundary detection. This situation is illustrated in Figure 3 where the inter-segments lexical repetitions are covered in yellow and the intra-segments lexical repetitions are covered in red. By tackling this particular situation, we propose a new challenge compared to other works that have been proposed so far and use test corpora based on multi-domain and multi-genre segments as in [19][23][28]. In fact, the most similar experiment, to our knowledge, is the one proposed by [27] who use the *Mars* novel. However, their segments are 2650 words-long while we deal with segments around 100 words each. In fact, we aim at proposing a fine-grained system capable of finding topic boundaries with high precision in a single domain and in short texts. To our knowledge, such a challenge has never been attempted so far.

```
O avançado brasileiro, novo reforço do Sporting, revelou hoje que vai viajar rapidamente para Lisboa, com o objectivo de assinar pelos «leões», cumprir os habituais exames médicos e começar a trabalhar às ordens do técnico José Peseiro. «O meu empresário está aí em Lisboa e disse-me que estava tudo acertado. Neste momento eu já me considero como jogador do Sporting», realçou Mota, em declarações à Renascença. O ponta-de-lança «canarinho», que está de férias no Brasil, revela que vai precisar de algum tempo para alcançar o mesmo nível físico dos restantes companheiro: «Vou procurar ficar bem fisicamente o mais rapidamente possível para entrar em campo e ajudar o Sporting a conquistar mais vitórias.» Para concluir, Mota, que vai viajar amanhã rumo a Portugal, admitiu que tem falado com os seus empresários para saber mais informações da cidade e dos jogadores do Sporting: «Tenho falado com os empresários para saber mais do clube e dos jogadores.».
```

```
O Nacional venceu esta noite na Choupana o Sporting por 3-2, na partida que marcou a saída de Casemiro Mior do clube insular. Com este resultado, os «leões» desperdiçaram o deslize de FC Porto e também a oportunidade de ascender ao primeiro lugar isolado do pódio. Os primeiros minutos de jogo davam sinais de que o Sporting estava a entrar bem no jogo e de pretendia «aceitar» a oportunidade da véspera proporcionada pelo FC Porto, - que foi empatar a Coimbra ante o último classificado (0-0) e voltar assim a reassumir a liderança da SuperLiga. Mas cedo essa imagem foi desfeita, a falta de ideias dos jogadores leoninos e a sua consequente ineficácia permitiram à equipa da casa, que pouco fazia para se abeirar da baliza adversária, aproveitar dois erros defensivos e chegar ao golo. Uma falha de Polga à passagem pelo minuto 18 permite a Adriano abrir a contagem na Choupana. Dois minutos volvidos Emerson, livre de marcação, recebe o esférico e dilata a vantagem, fazendo o 2-0.
```

**Figure 3: Our Test corpus**

In order to evaluate our system, we propose two distinct experiments. First, we propose an evaluation on a set of web documents about a unique domain using words as the basic textual information. In a second experiment, we show that semantic knowledge automatically acquired from the text, embodied by Multiword Units, can improve previous results. For that purpose, we use the SENTA Software proposed by [29] that can be run "on the fly" due to its efficient implementation [30] and flexibility as it does not need any previous knowledge.

In order to run our experiments, we built our own corpus by taking from two Portuguese soccer websites[3] a set of 100 articles of approximatively 100 words each. Then, we built 10 test corpora by choosing randomly 10 articles from our database of 100 articles[4] leading to 10 texts of around 1000 words-long[5].

A classical way of evaluating retrieval systems is to use Precision, Recall and F-measure. So, we show the results obtained by our system on our test corpus in Table 1.

**Table 1. Quantitative Results**

| | Without multiword units | | With multiword units | |
|---|---|---|---|---|
| | Measures | c=-1.5 | Measures | c=-2 |
| T1 | Precision | 0,64 | Precision | 0,58 |
| | Recall | 0,78 | Recall | 0,78 |
| | F-measure | 0,70 | F-measure | 0,66 |
| T2 | Precision | 0,67 | Precision | 0,73 |
| | Recall | 0,67 | Recall | 0,89 |
| | F-measure | 0,67 | F-measure | 0,80 |
| T3 | Precision | 0,80 | Precision | 1,00 |
| | Recall | 0,89 | Recall | 1,00 |
| | F-measure | 0,84 | F-measure | 1,00 |
| T4 | Precision | 0,73 | Precision | 0,64 |
| | Recall | 0,89 | Recall | 0,78 |
| | F-measure | 0,80 | F-measure | 0,70 |
| T5 | Precision | 0,60 | Precision | 0,64 |
| | Recall | 0,67 | Recall | 0,78 |
| | F-measure | 0,63 | F-measure | 0,70 |
| T6 | Precision | 0,73 | Precision | 0,62 |
| | Recall | 0,89 | Recall | 0,89 |
| | F-measure | 0,80 | F-measure | 0,73 |
| T7 | Precision | 0,80 | Precision | 0,82 |
| | Recall | 0,89 | Recall | 1,00 |
| | F-measure | 0,84 | F-measure | 0,90 |
| T8 | Precision | 0,64 | Precision | 0,64 |
| | Recall | 0,78 | Recall | 0,78 |
| | F-measure | 0,70 | F-measure | 0,70 |
| T9 | Precision | 0,60 | Precision | 0,45 |
| | Recall | 0,67 | Recall | 0,56 |
| | F-measure | 0,63 | F-measure | 0,50 |
| T10 | Precision | 0,70 | Precision | 0,80 |
| | Recall | 0,78 | Recall | 0,89 |
| | F-measure | 0,74 | F-measure | 0,84 |
| Average | Precision | 0,69 | Precision | 0,69 |
| | Recall | 0,79 | Recall | 0,84 |
| | F-measure | 0,73 | F-measure | 0,75 |

The results are surprisingly good considering the challenging task we were facing. Indeed, by using words as basic textual units, the average F-measure reaches 73% being Recall 79% and Precision 69%. After different tuning, the best results were obtained for the value c=-1.5.

By using Multiword Unit identification, the results show slight improvements with an average F-measure value of 75% being Recall improved by 5% (84%) and Precision remaining unchanged (69%). In this second experiment, the best results were obtained with c=-2.The introduction of Multiword Units allows a bigger number of correct decisions compared to single word processing in some cases (T3 and T7 specifically). However, in other ones, word units work better than with the introduction of Multiword Units like in T9. In fact, when texts gather many small sentences, the *ps(.)* function show bad behavior. In particular, T9 shows this particularity which is enhanced by the integration of Multiword Units leading to even worse results. In fact, by analyzing T9, we discovered that there were two sentences with 2 words and one sentence with only one word[6].

In any case, these global results hide most of the behavior of our system and a more detailed evaluation is needed.

**Table 2. Qualitative Results**

| | Without multiword units | | With multiword units | |
|---|---|---|---|---|
| | Match | c=-1.5 | Match | c=-2 |
| T1 | A | 7 | A | 7 |
| | ±1 | 2 | ±1 | 1 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 4 |
| T2 | A | 6 | A | 8 |
| | ±1 | 2 | ±1 | 1 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 1 | F | 2 |
| T3 | A | 8 | A | 9 |
| | ±1 | 1 | ±1 | 0 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 1 | F | 0 |
| T4 | A | 8 | A | 7 |
| | ±1 | 0 | ±1 | 1 |
| | ±2 | 1 | ±2 | 1 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 2 |
| T5 | A | 6 | A | 7 |
| | ±1 | 2 | ±1 | 1 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 3 |
| T6 | A | 8 | A | 8 |
| | ±1 | 1 | ±1 | 1 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 4 |
| T7 | A | 8 | A | 9 |
| | ±1 | 1 | ±1 | 0 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 1 | F | 2 |
| T8 | A | 7 | A | 7 |
| | ±1 | 2 | ±1 | 1 |
| | ±2 | 0 | ±2 | 1 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 2 |
| T9 | A | 6 | A | 5 |
| | ±1 | 2 | ±1 | 2 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 4 |
| T10 | A | 7 | A | 8 |
| | ±1 | 1 | ±1 | 1 |
| | ±2 | 0 | ±2 | 0 |
| | >2 | 0 | >2 | 0 |
| | F | 2 | F | 1 |

[3] http://www.abola.pt and http://www.ojogo.pt.

[4] We used the same methodology as [1] to build the test corpora although in a smaller scale.

[5] The chosen parameters of our experiments were the following: block size=2 sentences and EI window=10 words.

[6] We are already working on a normalization measure that takes into account sentence length.

As [9] evidences, Precision and Recall measures are overly strict. By taking into account only Precision and Recall, a hypothesized boundary close to a real segment boundary is equally detrimental to performance as one far from a boundary. This definitely should not be the case. In order to solve this problem, [15] proposed a metric that weights exact matches more than near misses and yields a single score. However, [15] observed that computing this metric requires some knowledge of the collection as parameters have to be tuned and as a consequence, performance comparison on different collections may be difficult. So, up-to-now, there is no standard evaluation measure that the community agrees on. As a consequence, we present, in Table 2, the qualitative results of our system where (1) A stands for the number of exact matches, (2) ± *n* stands for the number of boundaries that missed the true boundary for *n* sentences, (4) >2 stands for the number of boundaries that missed the true boundary for more than two sentences and (5) F stands for the boundaries that were proposed by the system that do not have any match in the test segmented text i.e. false boundaries.

We can see from these results, which by taking into account, as correct boundaries, all A and near misses ± 1, that we would obtain between 84% and 89% F-measure as shown in Table 3.

**Table 3. Estimated Results**

| Without multiword units | | With multiword units | |
|---|---|---|---|
| Precision | 0,83 | Precision | 0,77 |
| Recall | 0,95 | Recall | 0,93 |
| F-measure | 0,89 | F-measure | 0,84 |

The results presented in this section are promising as we deal with a very difficult challenge which is working without any linguistic knowledge, on the basis of small mono-domain texts with many inter-segments lexical repetitions. As we said earlier, to our knowledge, such a challenge has never been attempted so far. Although the quantitative and qualitative results show good figures, some work still need to be done, in particular, with respect to the sizes of the sentences in texts that cause some trouble in the topic boundary extraction.

## 7. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a language-independent unsupervised Topic Segmentation system based on word-co-occurrences that avoids the accessibility to existing linguistic resources such as electronic dictionaries or lexico-semantic databases. In particular, our architecture proposes a system that solves three main problems evidenced by previous research: systems based uniquely on lexical repetition that show reliability problems, systems based on lexical cohesion using existing linguistic resources that are usually available only for dominating languages and as a consequence do not apply to less favored and emerging languages and finally systems that need previously existing harvesting training data. To our point of view, our main contribution to the field is the definition of a new similarity measure, the informative similarity measure, *infosimba*, that proposes a well-founded mathematical model that deals with the word co-occurrence factor and avoids an extra step in the boundary detection compared to the solution introduced by [17]. Our evaluation has shown promising results both with word units and Multiword Units. Indeed, by using words as basic textual units, the average F-measure reaches 73% being Recall 79% and Precision 69%. Comparatively, by using Multiword Unit identification, the results show slight improvements with an average F-measure value of 75% being Recall improved by 5% (84%) and Precision remaining unchanged (69%).

However, the existence of three main parameters (the block size, the window size to calculate the association measure and the topic discovery threshold) may introduce some drawbacks in our solution, although it also provides interesting properties. We will start with the properties. Thanks to the existence of these parameters, fine-tuning of Topic Segmentation can be done. Indeed, depending on the type of the Topic Segmentation that is required (Topic Segmentation inside one main topic text or Topic Segmentation inside a webpage that contains drastically different news as in electronic newspapers), the adjustment of the parameters may allow a coherent segmentation. However, the existence of parameters is a drawback for totally flexible systems. Indeed, these parameters need to be tuned depending on the wanted application and are usually evaluated by experimentation which introduces partial judgment. It is clear that theoretical work should be carried out in order to avoid the tuning of these parameters; maybe following [17] and [15] that propose research directions to avoid the tuning by experimentation.

As immediate future work, we intend to test our system in different conditions of Topic Segmentation in order to find some clues that could help us in the definition of new theories to avoid parameter tuning. We will also experiment different association measures within the informative similarity measure in order to test whether drastically different results may be evidenced. Finally, we strongly think that more work must be done on the automatic boundary detection algorithm. In particular, we are convinced that better algorithms may be proposed based on the transformation of the representation of the *ps(.)* function into a graph or network. For that purpose, we would like to investigate possible solutions based on statistical mechanics of complex networks [33]. The system and its evolutions will be available for download as a GPL license at the following address: http://asas.di.ubi.pt.

## 8. REFERENCES

[1] Choi, F.Y.Y. 2000. Advances in Domain Independent Linear Text Segmentation. In Proceedings of NAACL'00, Seattle, April 2000. ACL.

[2] Salton, G., Allan, J. and Buckley, C. 1993. Approaches to passage retrieval in full text information systems. In Proceedings of ACM-SIGIR. 4--58.

[3] Kaszkiel, M. and Zobel, J. 1997. Passage retrieval revisited. In Proceedings of ACM-SIGIR, 178--185.

[4] Cormack, G.V., Clarke, C.L.A., Kisman, D.I.E. and Palmer, C.R. 1999. Fast Automatic Passage Ranking. MultiText Experiments for TREC-8. In Proceedings of TREC-8. 735-742.

[5] Boguraev, B. and Neff, M. 2000. Discourse segmentation in aid of document summarization. In Proceedings of Hawaii International Conference on System Sciences (HICSS- 33), Minitrack on Digital Documents Understanding, Maui, Hawaii. IEEE.

[6] Angheluta, R., De Busser, R., Moens, M-F. 2002. The Use of Topic Segmentation for Automatic Summarization. In Workshop on Text Summarization in Conjunction with the

ACL 2002 and including the DARPA/NIST sponsored DUC 2002 Meeting on Text Summarization. July 11-12, Philadelphia, Pennsylvania, USA.

[7] Farzindar, A. and Lapalme, G. 2004. Legal text summarization by exploration of the thematic structures and argumentative roles. In Text Summarization Branches Out Conference held in conjunction with ACL 2004, Barcelona, Spain, 27-38

[8] Hearst, M. 1994. Multi-Paragraph Segmentation of Expository Text, In Proceedings of the 32nd Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico, June, 9--16.

[9] Reynar, J.C. 1994. An Automatic Method of Finding Topic Boundaries. In Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics (Student Session), Las Cruces, New Mexico, USA.

[10] Richmond, K., Smith, A., and Amitay, E. 1997. Detecting subject boundaries within text: A language independent statistical approach. In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP--97), Providence, Rhode Island, August 1-2. 4--54.

[11] Yaari, Y. 1997. Segmentation of expository text by hierarchical agglomerative clustering. In Proceedings of the Conference on Recent Advances in Natural Language Processing, 59--65.

[12] Sardinha, T.B. 2002. Segmenting corpora of texts. DELTA, 2002, 18(2), 273--286. ISSN 0102-4450.

[13] Morris, J. and Hirst, G. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text, Computational Linguistics 17(1): 21--43.

[14] Kozima, H. 1993. Text Segmentation Based on Similarity between Words. In Proceedings of the 31th Annual Meeting of the Association for Computational Linguistics (Student Session), Colombus, Ohio, USA, 286--288.

[15] Beeferman, D., Berger, A., and Lafferty, J. 1997. Text segmentation using exponential models. In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, 35--46.

[16] Phillips, M. 1985. Aspects of Text Structure: An Investigation of the Lexical Organisation of Text, North Holland Linguistic Series, North Holland, Amsterdam.

[17] Ponte J.M. and Croft W.B. 1997. Text Segmentation by Topic. In Proceedings of the First European Conference on Research and Advanced Technology for Digitial Libraries.120--129.

[18] Xu, J. and Croft, W.B. 1996. Query Expansion Using Local and Global Document Analysis. In Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 4--11.

[19] Ferret, O. 2002.*Using Collocations for Topic Segmentation and Link Detection*. In Proceedings of COLING 2002, 19th International Conference on Computational Linguistics, August 24 - September 1, 2002, Howard International House and Academia Sinica, Taipei, Taiwan.

[20] Sparck-Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 28(1), 11--21.

[21] Salton, G., Yang, C.S., and Yu, C.T. 1975. A theory of term importance in automatic text analysis. Amer. Soc. Inf. Sc~ 26, 1, 33--44.

[22] Muller, C., Polanco, X., Royauté, J. and Toussaint, Y. 1997. Acquisition et structuration des connaissances en corpus: éléments méthodologiques. Technical Report RR-3198, Inria, Institut National de Recherche en Informatique et en Automatique. http://www.inria.fr/rrrt/rr-3198.html

[23] Moens, M-F. and De Busser, R. 2003. Generic topic segmentation of document texts. In Proceedings of the 24th annual international ACM SIGIR conference on Documentation. San Francisco, USA. 117--124.

[24] Cleuziou G., Clavier V., Martin L. 2003. Une méthode de regroupement de mots fondée sur la recherche de cliques dans un graphe de cooccurrences. In Proceedings of the 5èmes rencontres Terminologie et Intelligence Artificielle), LIIA - ENSAIS ed., pages 179--182, Strasbourg, France.

[25] Silva, J., Dias, G., Guilloré, S. and Lopes, J.G.P. 1999. Using LocalMaxs Algorithm for the Extraction of Contiguous and Non-contiguous Multiword Lexical Units. In the 9th Portuguese Conference in Artificial Intelligence. Pedro Barahona and Júlio Alferes (eds). Lecture Notes in Artificial Intelligence nº1695, Springer-Verlag, Universidade de Évora, Évora, Portugal, September, 113--132.

[26] Stokes, N., Carthy, J. and Smeaton, A.F. 2002. Segmenting Broadcast News Streams Using Lexical Chains. In Proceedings of 1st Starting AI Researchers Symposium (STAIRS 2002), volume 1, pp.145--154.

[27] Xiang, J. and Hongyuan, Z. 2003. Domain-independent Text Segmentation Using Anisotropic Diffusion and Dynamic Programming. In proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada. pp.322—329.

[28] Brants, T., Chen, F. and Tsochantaridis, I. 2002. Topic-Based Document Segmentation with Probabilistic Latent Semantic Analysis. In Proceedings of the CIKM 11th International Conference on Information and Knowledge Management, McLean, Virginia, USA. Pp.211-218.

[29] Dias, G., Guilloré, S., Bassano, J.C. and Lopes, J.G.P. 2000. *Extraction Automatique d'unités Lexicales Complexes: Un Enjeu Fondamental pour la Recherche Documentaire*. In Traitement Automatique des Langues, Vol 41:2, Christian Jacquemin (eds). Paris, France. pp. 447-473. ISBN: 2-7462-0225-5.

[30] Gil, A. & Dias, G. (2003). *Using Masks, Suffix Array-based Data Structures and Multidimensional Arrays to Compute Positional Ngram Statistics from Corpora*. In proceedings of the Workshop on Multiword Expressions of the 41st Annual Meeting of the Association of Computational Linguistics, Sapporo, Japan, July 7-12. pp. 25-33. ISBN: 1-932432-20-5.

[31] Albert, R. and Barabási, A-L. 2002. Statistical mechanics of complex networks. In Reviews of Modern Physics. Vol 74. The American Physical Society. January.

# Hypernyms Ontologies for Semantic Indexing

## Using Minimum Redundancy Cut as Indexing Set can Improve Information Retrieval

Florian Seydoux
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
florian.seydoux@epfl.ch

Jean-Cédric Chappelier
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
jean-cedric.chappelier@epfl.ch

## ABSTRACT

This paper presents a new method that improves semantic indexing while reducing the number of indexing terms. Indexing terms are determined using a minimum redundancy cut in a hierarchy of conceptual hypernyms provided by an ontology (e.g. *WordNet*, *EDR*). The results of some information retrieval experiments carried out on several standard document collections using the *EDR* ontology are presented, illustrating the benefit of the method.

## Categories and Subject Descriptors

H.3.1 [**Information storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods, Thesauruses*; H.1.1 [**Models and principles**]: Systems and Information Theory; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

## General Terms

ALGORITHMS, EXPERIMENTATION, THEORY

## Keywords

INFORMATION RETRIEVAL, SEMANTIC INDEXING, MINIMUM REDUNDANCY, TREE CUT, ONTOLOGY, EDR

## 1. INTRODUCTION

Using semantic knowledge within the Information Retrieval framework has already been addressed in the past. Three fields are mainly reported in the literature: *query expansion* [19, 12], *Word Sense Disambiguation* [7, 22, 1] and *semantic indexing.* This contribution relates to the latest.

The main idea of semantic indexing is to use word senses rather than (or in addition to) the words[1] for indexing documents, in order to improve both recall (by handling synonymy) and precision (by handling homonymy and polysemy). However, the related experiments reported in the literature lead to contradicting results: some claim that the addition or substitution, carried out in an automatic way, degrades the performance of their system [14, 5, 18, 20]; for others conversely, the gain seems significant [13, 17, 3, 4, 10].

Although it is definitely desirable for an IR system to take a maximum of (semantic) information into account, the resulting expansion of the data processed could happen to be counter-productive or at least not develop its full potential. Indeed, the growth of the number of index terms not only increases the processing time, but could also reduce the precision: discriminating documents by using a very large number of index terms is a hard task, the "distance"[2] between documents tending to become more or less the same ("*curse of dimensionality*" effect). This problem is not new, and various techniques aiming at reducing the size of the indexing set already exist: filtering by stoplist, part of speech tags, frequencies, but also through statistical techniques as in LSI [2] or PLSI [6]. However, most of these techniques are not adapted to the case where an explicit semantic information is available in the form of an ontology (i.e. with some underlying formal – not statistical – structure).

The focus of our work is to use external[3] structured semantic resources such as an ontology in order to limit the semantic indexing set. This relates, but from a different point of view, with experiments described in [4], [21] or [10], which uses the synsets (or hypernyms synsets [10]) of *WordNet* as indexing terms. We follow the *onto-matching* technique described in [8], but here selecting the indexing set using an information theory based criterion, the *Minimum Redundancy Cut* (*MRC*, see figure 1), applied to the inclusive "is-a" relation (hypernyms) provided by the *EDR* taxonomy [11].

[1] Usually lemmas or stems.

[2] Usually a similarity.

[3] By "external" we mean "not directly related to the document collection itself".

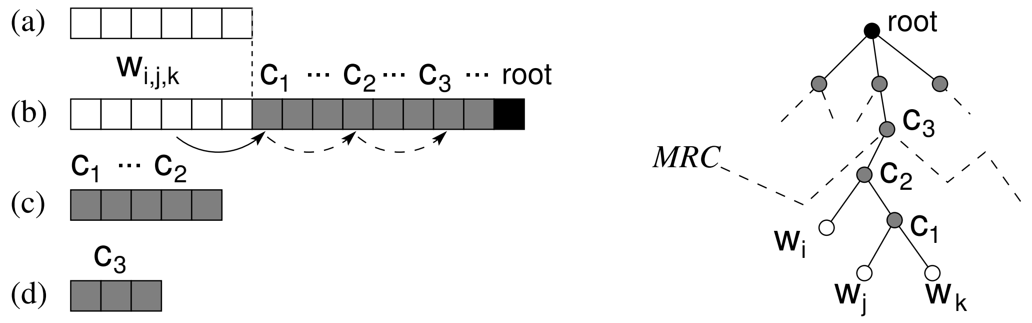


**Figure 1:** Several indexing scheme: (a) usual indexing with words, stems or lemmas; (b) using a semantic ontology (displayed on the right), each indexing term is *extended* with all/a part of the concepts that dominate this term; this leads to an explosion of the number of indexes for documents; (c) synset (or hypernyms synsets) indexing: each indexing term is *replaced* with its (hypernyms) synset; this, in principle, can reduce the size of the indexing set since all the indexing terms that are covered by the same hypernym are regrouped in one single indexing feature; (d) Minimum Redundancy Cut (*MRC*) indexing: each indexing term is *replaced* with its dominating concept chosen with *MRC*. This actually reduces the size of the indexing set since all the indexing terms that are subsumed by the same concept in the *MRC* are regrouped in one single indexing feature.

## 2. ONTOLOGY-CUT MODEL

### 2.1 Goals

The choice of the appropriate hypernym (a "concept" in the ontology) to be used for representing a word is not easy: be it too general, the performance of the system will degrade (lack of precision); be it too specific, the indexing set will not reduce enough, preserving some distinction between words with close senses (lack of recall).

To select the appropriate level of conceptual indexing, we consider cuts in the ontology. A cut in the directed acyclic graph (DAG) representing the ontology is defined as a minimal subset[4] of nodes in the ontology defining a coverage of all the leaf nodes (i.e. words). Each node in the cut then represents every leaf node it dominates.

The problem is to find a computable strategy to select an optimal cut. For a related task, Li and Abe [9] use the Minimum Description Length principle (MDL). Although easy to compute, this criterion has in practice the drawback of often selecting as a cut the root of the ontology, which is not really useful for document indexing. We rather propose to use a new criterion, based on information theory, that selects a cut for which the redundancy is minimal, i.e. a cut where the degree of description of the indexing features in the ontology is balanced as much as possible (maximum entropy).

### 2.2 Minimum Redundancy Criterion

Let $\mathcal{N} = \{n_i\}$ and $\mathcal{W}$ respectively be the set of nodes and the set of words in the ontology. A cut $\Gamma$ is defined as a minimal[4] subset of $\mathcal{N}$ which covers $\mathcal{W}$. A probabilized cut $M = (\Gamma, P)$ is a couple consisting of a cut $\Gamma$ and a probability distribution $P$ on $\Gamma$. Finally, $|\Gamma|$ denotes the number of nodes in the cut $\Gamma$ (and similarly $|M| = |\Gamma|$).

From now on, the probabilized cut $M = (\Gamma, P_f)$ is considered, where $P_f$ is defined using the relative frequencies of the words in the collection:

$$P_f(n_i) = \frac{f(n_i)}{|D|},$$

[4] "minimal subset" means that no node can be removed from the set without decreasing it's coverage.

$f(n_i)$ being the number of occurrences of the node $n_i$ in a document collection $D$. To compute $f(n_i)$, we consider that an occurrence of $n_i$ happens when any of the hyponym words of $n_i$ occurs.

The redundancy $R(M)$ of a probabilized cut $M = (\Gamma, P)$ is defined as [16]:

$$R(M) = 1 - \frac{H(M)}{\log |M|},$$
$$\text{where } H(M) = -\sum_{n \in \Gamma} P(n) \cdot \log P(n).$$

Minimizing the redundancy is thus equivalent to maximizing the ratio between the entropy $H(M)$ of the cut and its maximum possible value ($\log |M|$), i.e. balancing as much as possible the probabilities of the nodes in the cut.

Notice that $R$ does not necessarily have a unique minimum, but the ontology may rather have several equally minimal cuts. In practice, this can easily be overcome, considering for instance any of the minimal cuts, or those having a minimal number of nodes, or the minimum average depth of the nodes, etc.

In order to identify global *MRC*, the whole set of possible cuts has to be considered. We thus decided to give up global optimality for the sake of tractability and focussed rather on more efficient heuristics.

The algorithm we propose here consists in iteratively modifying a given cut, starting from the leaves, by systematically choosing the replacement of a node by its parent or its children that minimizes the redundancy. For each node $n_i$ in the current cut, we consider on one hand ${n_i}^{\downarrow}$ the (set of) children of $n_i$, and on the other hand ${n_i}^{\uparrow}$ the (set of) parents of $n_i$. Due to the DAG structure, this replacement can involve other nodes in the cut. In fact, when replacing $n_i$ by ${n_i}^{\downarrow}$, we must exclude those nodes which are already covered by other nodes in the cut, i.e. consider ${n_i}^{\downarrow} \setminus (\Gamma \setminus \{n_i\})^{\Downarrow}$ instead of ${n_i}^{\downarrow}$ (see fig. 2), where $n^{\Downarrow}$ is the transitive closure of $n^{\downarrow}$; and similarly for ${n_i}^{\uparrow}$ (see fig. 3).

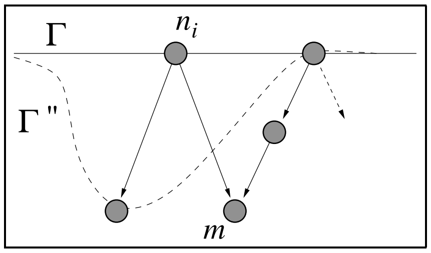


**Figure 2: Lower search: node $n_i$ is replaced by $n_i^{\downarrow}$ but those nodes already covered by other nodes in $\Gamma$ (e.g. $m$), i.e. $(\Gamma \setminus \{n_i\})^{\Downarrow}$.**

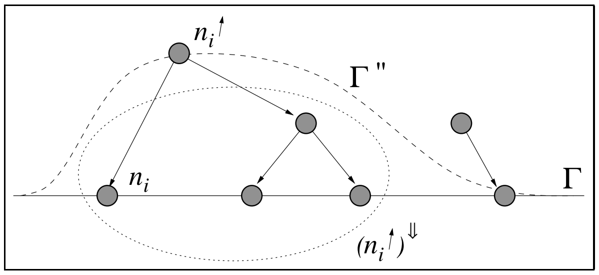


**Figure 3: Upper search: node $n_i$ is replaced by $n_i^{\uparrow}$, and all nodes covered by $n_i^{\uparrow}$ (i.e. $(n_i^{\uparrow})^{\Downarrow}$) are removed from $\Gamma$.**

Then, the cut with minimal redundancy among these new considered cuts and the current one is kept, and the search continue as long as better cuts are found. The full algorithm[5] is given hereafter (Algorithm 1).

This algorithm converges towards a local minimum redundancy cut close to the leaves. Furthermore, since this algorithm always works on a complete cut, it can be stopped at any time, if required.

## 2.3 Example

Let us illustrate the *MRC* on the toy example of the ontology given in figure 4. With this data, the redundancy of the example cut $\Gamma$ = [Animal, Plant, Transport] is given by:

| $n$ | Animal | Plant | Transport |
|---|---|---|---|
| $f(n)$ | 20 | 33 | 2 |
| $P_f(n)$ | 0.3704 | 0.5926 | 0.0370 |
| $-P_f(n)\log_2 P_f(n)$ | 0.5307 | 0.4473 | 0.1761 |
| $R(\Gamma) = 1 - \frac{1.1541}{\log_2(3)} = 0.2718$ | | | |

In this case, by examining each of the 2036 possible cuts, one can check that the global *MRC* is also the local one found by the local search algorithm (with only 117 evaluation), and for which the redundancy is 0.07092 (see figure 4).

Regarding the considered indexing schemas (see fig. 1), consider the following three documents:

| | |
|---|---|
| d₁ | myosotis tree bicycle myosotis lion |
| d₂ | lion cow carnivore |
| d₃ | violet car fir carnivore |

[5] In practice, several optimizations can be made, which do not conceptually change the algorithm and are thus not presented here for the sake of clarity.

**Algorithm 1** *MRC* local search algorithm

**Requires:** a hierarchy $\mathcal{N}$ (the leaves of which are $\mathcal{W}$)
**Provides:** a cut $\Gamma$ with (local) minimal redundancy

```
$\Gamma \leftarrow \mathcal{W}$ # current cut, start from the leaves
repeat
  $\Gamma' \leftarrow \varnothing$ # best new cut
  $\Gamma'' \leftarrow \varnothing$ # tested candidate
  continue ← false # search-loop control flag
  for all $n_i \in \Gamma$ do
    # Evaluate the children's cut:
    $\Gamma'' \leftarrow (\Gamma \setminus \{n_i\}) \cup \left(n_i^{\downarrow} \setminus (\Gamma \setminus \{n_i\})^{\Downarrow}\right)$
    $\Gamma' \leftarrow \mathrm{Argmin}\left(R(\Gamma'), R(\Gamma'')\right)$
    # Evaluate each parent's cut:
    for all $n_j \in n_i^{\uparrow}$ do
      $\Gamma'' \leftarrow (\Gamma \cup \{n_j\}) \setminus n_j^{\Downarrow}$
      $\Gamma' \leftarrow \mathrm{Argmin}\left(R(\Gamma'), R(\Gamma'')\right)$
  if $R(\Gamma') < R(\Gamma)$ then
    $\Gamma \leftarrow \Gamma'$ # keep the best cut
    continue ← true # the search goes on
    # some watchdog or timer can be put here
until continue is false
return $\Gamma$
```

with $R(\varnothing) = R(\{c\}) = 1$, by convention.

The baseline words indexing (scheme (a)) gives:

| | |
|---|---|
| $\mathrm{Id}_1^{(a)}$ | bicycle:1 lion:1 myosotis:2 tree:1 |
| $\mathrm{Id}_2^{(a)}$ | carnivore:1 cow:1 lion:1 |
| $\mathrm{Id}_3^{(a)}$ | car:1 carnivore:1 fir:1 violet:1 |

Indexing extended by all hypernyms (scheme (b)) gives:

| | |
|---|---|
| $\mathrm{Id}_1^{(b)}$ | bicycle:1 lion:1 myosotis:2 tree:1 Animal:1 BlueFl:2 Carnivore:1 Ecological:1 Entity:5 Flower:2 Mammal:1 Plant:3 Transport:1 Tree:1 |
| $\mathrm{Id}_2^{(b)}$ | carnivore:1 cow:1 lion:1 Animal:3 Carnivore:2 Entity:3 Herbivore:1 Mammal:3 |
| $\mathrm{Id}_3^{(b)}$ | car:1 carnivore:1 fir:1 violet:1 Animal:1 BlueFl:1 Carnivore:1 Entity:4 Flower:1 Mammal:1 Plant:2 Pollutant:1 Transport:1 Tree:1 |

Indexing by direct hypernyms (scheme (c)) gives:

| | |
|---|---|
| $\mathrm{Id}_1^{(c)}$ | BlueFl:2 Carnivore:1 Ecological:1 Tree:1 |
| $\mathrm{Id}_2^{(c)}$ | Carnivore:2 Herbivore:1 |
| $\mathrm{Id}_3^{(c)}$ | BlueFl:1 Carnivore:1 Pollutant:1 Tree:1 |

Indexing by example cut $\Gamma$ gives:

| | |
|---|---|
| $\mathrm{Id}_1^{(\text{ex. }\Gamma)}$ | Animal:1 Plant:3 Transport:1 |
| $\mathrm{Id}_2^{(\text{ex. }\Gamma)}$ | Animal:3 |
| $\mathrm{Id}_3^{(\text{ex. }\Gamma)}$ | Animal:1 Plant:2 Transport:1 |

Indexing by *MRC* cut (scheme (d)) gives:

| | |
|---|---|
| $\mathrm{Id}_1^{(d)}$ | BlueFl:2 Carnivore:1 Transport:1 Tree:1 |
| $\mathrm{Id}_2^{(d)}$ | cow:1 Carnivore:2 |
| $\mathrm{Id}_3^{(d)}$ | BlueFl:1 Carnivore:1 Transport:1 Tree:1 |

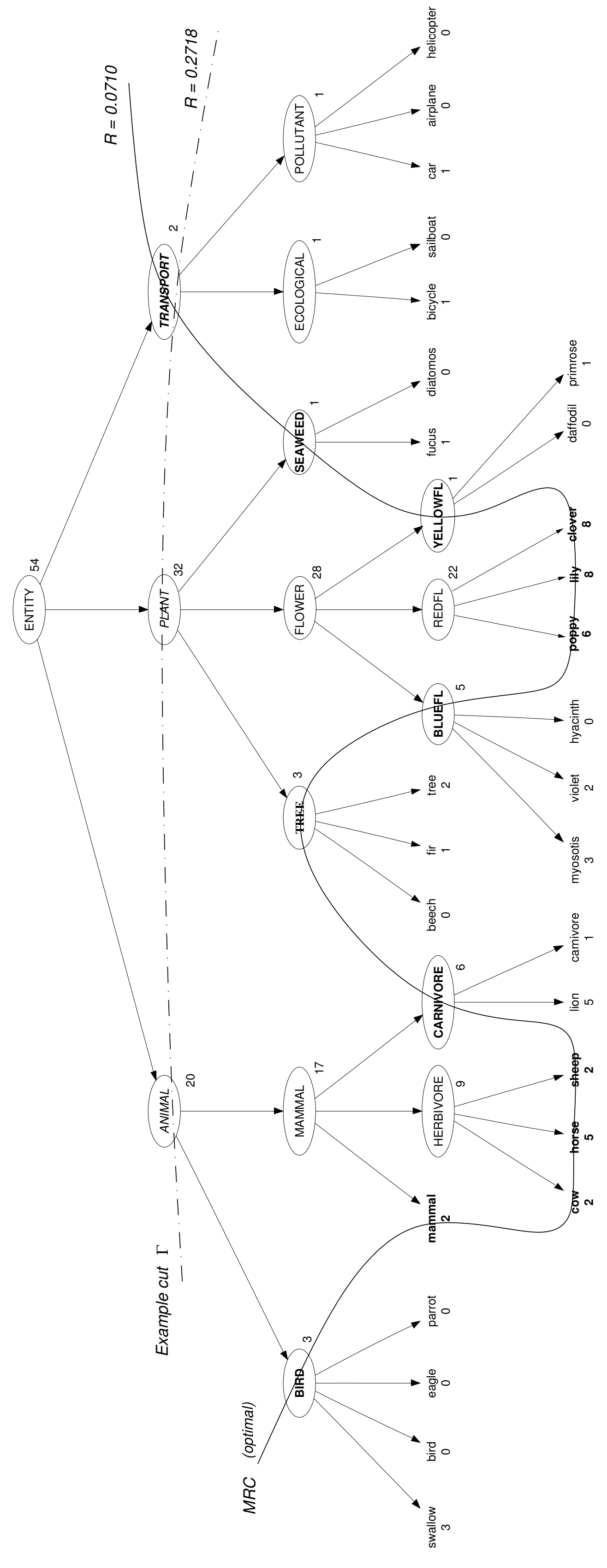


**Figure 4: Examples of cuts and their redundancy value in a toy ontology. Frequencies $f(n)$ of words and concepts (from the dataset) are indicated. Items of the *MRC* cut are in bold; items of the example cut $\Gamma = [\textsc{Animal}, \textsc{Plant}, \textsc{Transport}]$ are in italic. The redundancy is $0.071$ for the *MRC*, $0.272$ for $\Gamma$ and $0.093$ for the set of leaves.**

## 3. EXPERIMENTS

We carried out several experiments with standard document collections of the SMART system[6] [15], and an ontology generated from the english part of *EDR Electronic Dictionary*

*EDR* is organized in five different more or less independent dictionaries. Among these dictionaries, the following two are used to construct the ontology:

**Word Dictionary,** which gather morphological and syntactic information about approximatively 420,000 "words" (words, compounds and idiomatic expressions), related to $\approx 240,000$ lexical strings, and allowing to bind the lexical string with information from the concepts dictionary.

**Concept Dictionary,** describes approximately 490,000 concepts, with a super/sub relation referring to the inclusion relation between concepts. Several additional (binary) semantic relations are also available, such as 'agent', 'implement' and 'place', but they are not used in the current work. Notice that a significant number of concepts are not directly associated to words; they are defined only on the basis of their relationships to the other concepts.

For the evaluation, we use the vector-space SMART information retrieval system, and an external lemmatizer (which also acts as a tokenizer)[7]. A filtering based on the POS tag is carried out (but no stoplist, nor frequency filtering). The new indexing sets are produced while preprocessing the data as follow:

1. First of all, the textual information (title and contents) are aggregated for each document and query, all other information (authors, sources, etc) are removed; then, documents and queries are tokenized and lemmatized by the third-party tool.
2. We look then for the correspondences between the tokens in a document and the entries (leaves) in the ontology, with the lexical string first and the lemmatized form then, if necessary. Tokens without correspondence in the ontology are indexed in the standard way. The coverage rate of the collections by the ontology[8] is 90% in average.
3. Then the hierarchy of concepts related to the tokens found in the ontology is expanded by:
   (a) in a first set of experiments, selecting all possible senses (relying on the mutual reinforcement induced by collocations to have a sort of disambiguation);
   (b) in a second run of experiments, selecting only the most frequent sense[9].
4. An *MRC* cut is then computed with the algorithm previously presented (in practice we limit the cut only to the nodes covering words contained in the documents, but neither the whole ontology, nor the words in the queries).
5. Finally, the tokens of both documents and topics are substituted by the identifiers of the concepts which subordinate them in the cut determined at the preceding step. Tokens of the topics which are not subordinated by a concept of the cut are ignored.

Table 1 (hereafter) gather the 11-pt precision and the 30-doc recall for the experiments carried out.

[6] Available online at *ftp://ftp.cs.cornell.edu/pub/smart/*.

[7] Sylex 1.7, © 1993-98 DECAN INGENIA.

[8] The coverage rate is the number of different words in the collection that are in the ontology, divided by the total number of different words in the collection.

[9] For each words, this information is given by EDR, independently of the context.

## 4. DISCUSSION AND CONCLUSION

Three main conclusions can be drawn out of these experiments:

1. Using adapted additional semantic information can enhance the indexing of documents, and thus the performance of a IR system. The results of semantic (ontology-based) indexing (columns (c) and (d)) are indeed better than the baseline system for four of the collections, but slightly worse on the MED collection.

   This can be explained by the specificity of the vocabulary of these bases, and their adequacy with the semantic resource. ADI and CACM have an important technical vocabulary, but well covered by the *EDR* ontology[10]. CACM documents are extremely small, often restricted to a simple title of few words. Conversely, the vocabulary of TIME is very general and the length of each document is large. The CISI collection present documents of average size, but with a significant number of dates, proper names, etc., for which the POS filtering seems to have annoying consequences (the very low performance clearly indicate an initial loss of information). Finally, the MED collection has an extremely specific vocabulary, for which the *EDR* ontology is not adapted.
2. Semantic disambiguation, even rudimentary, appears to be necessary. Indeed, the simple heuristics consisting in choosing, for a given word, its most (absolute) frequent sense (provided by *EDR* dictionnary) already allows a significant increase in the performance (compare "most frequent concept" v.s. "all concept").

   The expected mutual reinforcement of collocations as a kind of "natural" disambiguation does actually not occur. The reason is probably that the hypernyms relations used do not constitute thematic links (for example, the thematically related terms "doctor", "drug", "hospital", "nurse", etc. are not linked together with the used ontology).

   Anyway, this result enfavours the use of a proper WSD procedure for further improving the results.
3. The overal performance of *MRC* indexing (column (d)) is slightly worse than indexing with direct hypernyms

[10] A complete branch of the *EDR* ontology covers terms and concepts of technical nature, from information sciences and related fields like computer sciences or electronics.

|  |  | (a) | (b) | (c) | (d) |
|---|---|---|---|---|---|
| ADI collection (82 documents, 35 topics) [Documents from Information Science] |  |  |  |  |  |
| all concepts, tf | *index size* | 1800 | 14748 | 10099 | 5496 |
|  | *precision* | 0.2497 | 0.1219 | 0.2550 | 0.2528 |
|  | *recall* | 0.5996 | 0.3452 | 0.6708 | 0.6781 |
| all concepts, tf.idf | *precision* | 0.3578 | 0.3134 | 0.3356 | 0.3138 |
|  | *recall* | 0.6984 | 0.7126 | **0.7406** | 0.7178 |
| most frequent concept, tf | *index size* | 1800 | 5255 | 2888 | 1740 |
|  | *precision* | 0.2497 | 0.1376 | 0.2939 | 0.2924 |
|  | *recall* | 0.5996 | 0.3727 | 0.7141 | 0.6901 |
| most frequent concept, tf.idf | *precision* | 0.3578 | 0.4060 | **0.4274** | 0.4266 |
|  | *recall* | 0.6984 | 0.7306 | 0.7217 | 0.7081 |
| TIME collection (423 documents, 83 topics) [General world news articles from the magazine Time (1963)] |  |  |  |  |  |
| all concepts, tf | *index size* | 21815 | 93707 | 70091 | 18565 |
|  | *precision* | 0.3288 | 0.0337 | 0.2353 | 0.3334 |
|  | *recall* | 0.7755 | 0.1021 | 0.5709 | 0.7253 |
| all concepts, tf.idf | *precision* | 0.5496 | 0.4231 | 0.4536 | 0.5512 |
|  | *recall* | 0.8901 | 0.7642 | 0.8036 | 0.8904 |
| most frequent concept, tf | *index size* | 21815 | 53140 | 31612 | 18465 |
|  | *precision* | 0.3288 | 0.0346 | 0.3692 | 0.3964 |
|  | *recall* | 0.7755 | 0.1201 | 0.7590 | 0.8124 |
| most frequent concept, tf.idf | *precision* | 0.5496 | 0.5143 | 0.5565 | **0.5602** |
|  | *recall* | 0.8901 | 0.8760 | **0.9053** | 0.8968 |
| MED collection (1033 documents, 30 topics) [Collection of abstract from a medical journal] |  |  |  |  |  |
| all concepts, tf | *index size* | 11893 | 51712 | 38524 | 19339 |
|  | *precision* | 0.3623 | 0.0105 | 0.1905 | 0.1426 |
|  | *recall* | 0.4574 | 0.0246 | 0.2749 | 0.2326 |
| all concepts, tf.idf | *precision* | **0.4607** | 0.3029 | 0.2996 | 0.2645 |
|  | *recall* | **0.5547** | 0.3903 | 0.3794 | 0.3554 |
| most frequent concept, tf | *index size* | 11893 | 30284 | 18109 | 10354 |
|  | *precision* | 0.3623 | 0.0105 | 0.3229 | 0.3251 |
|  | *recall* | 0.4574 | 0.0313 | 0.4230 | 0.4214 |
| most frequent concept, tf.idf | *precision* | **0.4607** | 0.4266 | 0.4518 | 0.4394 |
|  | *recall* | **0.5547** | 0.5169 | 0.5404 | 0.5344 |
| CISI collection (1460 documents, 112 topics) [Articles from Information science (Library science)] |  |  |  |  |  |
| all concepts, tf | *index size* | 10019 | 53453 | 39544 | 15344 |
|  | *precision* | 0.0687 | 0.0232 | 0.0569 | 0.0448 |
|  | *recall* | 0.1239 | 0.0376 | 0.0963 | 0.0903 |
| all concepts, tf.idf | *precision* | 0.1733 | 0.1043 | 0.1139 | 0.0878 |
|  | *recall* | **0.2318** | 0.1627 | 0.1675 | 0.1507 |
| most frequent concept, tf | *index size* | 10019 | 26246 | 14993 | 8156 |
|  | *precision* | 0.0687 | 0.0201 | 0.0805 | 0.0817 |
|  | *recall* | 0.1239 | 0.0403 | 0.1300 | 0.1243 |
| most frequent concept, tf.idf | *precision* | 0.1733 | 0.1590 | **0.1825** | 0.1785 |
|  | *recall* | **0.2318** | 0.2131 | **0.2313** | 0.2243 |
| CACM collection (3204 documents, 64 topics) [Collection of titles and abstracts from a Computer science journal] |  |  |  |  |  |
| all concepts, tf | *index size* | 10053 | 51712 | 38524 | 15641 |
|  | *precision* | 0.1555 | 0.0133 | 0.1447 | 0.1341 |
|  | *recall* | 0.3082 | 0.0306 | 0.2549 | 0.2317 |
| all concepts, tf.idf | *precision* | 0.2865 | 0.1293 | 0.1935 | 0.1700 |
|  | *recall* | 0.4534 | 0.2579 | 0.3617 | 0.3190 |
| most frequent concept, tf | *index size* | 10053 | 25207 | 14681 | 8339 |
|  | *precision* | 0.1555 | 0.0230 | 0.1472 | 0.1525 |
|  | *recall* | 0.3082 | 0.0302 | 0.2926 | 0.2996 |
| most frequent concept, tf.idf | *precision* | 0.2865 | 0.2358 | 0.2804 | **0.2964** |
|  | *recall* | 0.4534 | 0.3834 | **0.4567** | 0.4514 |

**Table 1: Evaluation of several indexing scheme (see figure 1) on several collections: (a): words only; (b): words + concepts; (c): direct hypernyms; (d) : hypernyms from the *MRC*.**

(column (c)): the two methods exhibit a good precision (more in favour of *MRC* on TIME and CACM, and more in favour of direct hypernyms on ADI an CISI), but the recall of the latter is systematically better.

However, *MRC* is often close, with a noticeably smaller indexing set; it furthermore performs better on large document (TIME) or specific vocabulary in adequation with the ontology (CACM), which is a good omen for the future of the method.

As futur work, it is forseen to confront the results presented here with similar experiments using *WordNet*, which has a quite different structure than *EDR*. It would be also interesting to generalize this technique with ontologies modelling thematic relationships between terms, in addition to a hierarchical structure of hypernyms.

Finally, we want to emphasize that the technique presented here does not limit to IR but could also be apply to other NLP domains, such as document clustering and text summarization.

## 5. REFERENCES

[1] R. Besançon, J.-C. Chappelier, M. Rajman, and A. Rozenknop. Improving text representations through probabilistic integration of synonymy relations. In *Proceedings of the Xth International Symposium on Applied Stochastic Models and Data Analysis (ASMDA'2001)*, volume 1, pages 200–205, 2001.

[2] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[3] J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarran. Indexing with WordNet synsets can improve text retrieval. In *Proc. of the COLING/ACL 1998 Workshop on Usage of WordNet for Natural Language Processing*, pages 38–44, 1998.

[4] J. Gonzalo, F. Verdejo, C. Peters, and N. Calzolari. Applying EuroWordNet to multilingual text retrieval. *Journal of Computers and the Humanities*, 32(2-3):185–207, 1998.

[5] D. Harman. Towards interactive query expansion. In *Proc. of the 11th Annual Int. ACM-SIGIR Conference on Research and development in information retrieval*, pages 321–331, 1988.

[6] T. Hofmann. Probabilistic latent semantic indexing. In *proc. of the 22th International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 50–57, 1999.

[7] N. Ide and J. Véronis. Word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

[8] A. K. Kiryakov and K. I. Simov. Ontologically supported semantic matching. In *in Proceedings of NODALIDA'99: Nordic Conference on Computational Linguistics, Trondheim*, December 1999.

[9] H. Li. A probabilistic approach to lexical semantic knowledge acquisition and structural disambiguation. Master's thesis, Graduate School of Science, University of Tokyo, 1998.

[10] R. Mihalcea and D. Moldovan. Semantic indexing using WordNet senses. In *Proc. of ACL Workshop on IR & NLP*, 2000.

[11] H. Miyoshi, K. S. amd M. Kobayashi, and T. Ogino. An overview of the EDR electronic dictionary and the current status of its utilization. In *Proc. of COLING*, pages 1090–1093, 1996.

[12] D. I. Moldovan and R. Mihalcea. Using wordnet and lexical operators to improve internet searches. *IEEE Internet Computing*, 4(1):34–43, 2000.

[13] R. Richardson and A. F. Smeaton. Using WordNet in a knowledge-based approach to information retrieval. Technical Report CA-0395, Dublin City University, Glasnevin, Dublin 9, Ireland, 1995.

[14] G. Salton. *Automatic Information Organization and Retrieval.* McGraw-Hill, 1968.

[15] G. Salton. *The SMART Retrieval System – Experiments in Automatic Document Processing.* Prentice Hall, 1971.

[16] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, July 1948.

[17] A. F. Smeaton and I. Quigley. Experiments on using semantic distances between words in image caption retrieval. In *Proc. of 19th Int. Conf. on Research and Development in Information Retrieval*, pages 174–180, 1996.

[18] E. M. Voorhees. Using WordNet to disambiguate word senses for text retrieval. In *Proc. of 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 171–80, 1993.

[19] E. M. Voorhees. Query expansion using lexical-semantic relations. In *Proc. 17th Annual Int. ACM-SIGIR Conf. on Research and Development in Information Retrieval*, pages 61–69, 1994.

[20] E. M. Voorhees. Using WordNet for text retrieval. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 12, pages 285–303. MIT Press, 1998.

[21] J. M. Whaley. An application of word sense disambiguation to information retrieval. Technical Report PCS-TR99-352, Dartmouth College, Computer Science, Hanover, NH, 1999.

[22] Y. Wilks and M. Stevenson. Word sense disambiguation using optimised combinations of knowledge sources. In *Proc. of the 17th Int. Conf. on Computational Linguistics*, pages 1398–1402, 1998.

## SHORT PAPERS

# An evaluation of Bi- and Trigram Enriched Latent Semantic Vector Models

Leif Grönqvist
School of Mathematics and Systems Engineering
Växjö University, Sweden
The National Graduate School of Language Technology (GSLT)
Göteborg, Sweden
leif.gronqvist@msi.vxu.se

## ABSTRACT

The main reason for this work is to find an appropriate way to include multi-word units in a latent semantic vector model. This would be of great use since these models normally are defined in terms of words, which makes it impossible to search for many types of multi-word units when the model is used in information retrieval tasks. The paper presents a Swedish evaluation set based on synonym tests and an evaluation of vector models trained with different corpora and parameter settings, including a rather naive way to add bi- and trigrams to the models. The best results in the evaluation is actually obtained with both bi- and trigrams added. Our hope is that the results in a forthcoming evaluation in the document retrieval context, which is an important application for these models, still will be at least as good with the bi- and trigrams are added, as without.

## Categories and Subject Descriptors

H.3 [**Information Systems**]: Information Storage and Retrieval; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

## General Terms

Algorithms, Experimentation, Performance, Languages

## Keywords

Evaluation, latent semantic indexing, semantic vector models, n-grams, multi-word units

## 1. INTRODUCTION

Latent Semantic Indexing (LSI) [1] has been around for a while since it was introduced to the world of information retrieval in the late 80's [6, 4]. A well functioning Latent Semantic Vector Model (LSVM) has been proven to improve recall but also precision for document retrieval. There are two straight forward ways to use an LSVM in document retrieval:

- Use the model to expand the query with relevant terms before the query is sent to the retrieval system [14, 15]
- Use the model to calculate a vector corresponding to the query, and return documents having vectors (obtained by the model) close to the query vector [16]

### 1.1 The vector space

A semantic vector space may be calculated automatically using some different methods. If we look at Random Indexing (RI) [15] and Singular Value Decomposition (SVD), [2, 11] they differ in one important way: RI often uses a small sliding window as context, and SVD normally uses a document as context. However, they do have important things in common: they both use a large set of text for training, and the extracted models give the possibility to calculate the distance between words and/or documents since these are represented as vectors in the vector space. The new vector space is a projection of the original co-occurrence based vector space but still has hundreds of dimensions. The methodology to calculate similarities (or distances) in a projected vector space is often referred to as Latent Semantic Indexing (LSI). [4, 6, 10] In latent semantic indexing each word, or term, [2] is represented as a vector in an N-dimensional space. Normal Euclidean distance, or other distance measures, give us the possibility to easily calculate the distance between words.

### 1.2 A major weakness with the current use of LSI

The problem arises when it comes to documents, and in this model, a phrase is modelled the same way as a short

[1]Note that there is an appendix describing used non standard terms

[2]Many researchers would prefer to have terms here rather than words but the training data contains words, so if we do not want to extract the terms automatically as well, we will have to use the words instead. As many linguists point out, this is far from optimal since words are ambiguous and many words may have more or less the same meaning. In a vector space containing vectors corresponding to terms, there would be no two different terms with the same meaning.

document – as a bag of words, which is the easiest possible solution. Both when the model is used in information retrieval and also during the calculation: just add the vectors for all words in the document, using vector addition. The same method is used for Multi-Word Units (MWUs). [3] Obviously this is not optimal, since the meaning of the n-word unit $w_1..w_n$ is not just the sum of the meanings of the words. Vector addition is transitive and commutative, which would result in the same meaning for the units $w_1w_2w_3$, $w_1w_3w_2$, and $w_3w_2w_1$. When putting two words together to a unit there is something more added to the meaning but just the meaning of the individual words. In some cases the meaning of the unit is totally distinct from the meaning of the single words, so this approach can never be perfect. An investigation by Anette Hulth shows that less than 14% of all manually selected keywords contain just one token, [8] so this problem should not be neglected if we want better precision and recall in document retrieval. One goal of this paper is to propose and evaluate some alternative ways to calculate the vectors corresponding to MWUs.

Note that a search for *Bengt Johansson* [4] in an LSI based search system will be a search for Bengt + Johansson, which would match articles related to any Bengt and any Johansson if there are any. In Swedish newspaper articles from the 90's we can read a lot about Bengt Johansson, but even more common are articles about the politicians Bengt Westerberg and Olof Johansson. This makes it almost impossible to find articles about Bengt Johansson since the vectors for Bengt and Johansson are much closer to the politics field than to handball. To solve this, we need to do something about the fact that names are highly ambiguous. We think that it is very difficult to add this information to a word based LSVM, and believe much more in the approach to keep it during the training phase.

### 1.3 Evaluation

Used in an information retrieval context, LSI helps a lot despite the MWU problem. For some specific search queries there will be a problem but the IR testbeds also contain many queries where MWUs are not important, so even a vector model not able to handle MWUs does reasonably well. In this paper we present a new evaluation method which, used in combination with an IR testbed, will show how well a model handles both single word keywords and MWUs, especially if it is able to find synonyms among a set of terms. We think that the two evaluation methods, a document retrieval task, which we want to look further into, and a synonym test, will complement each other in a very nice way.

## 2. THE EVALUATION METHOD

We are using a revised version [5] of *the Infomap NLP System: An Open-Source Package for Natural Language Processing*, developed by the Computational Semantics Lab at Stanford University's Center for the Study of Language and Information. It is an implementation of singular value decomposition (SVD) and interface software to calculate a word by word matrix from a corpus, running the SVD, and functions to calculate and compare vectors in the new projection. The interface is easy to use and also seems to work very well.

To be able to test the various combinations of corpora and parameters, we have written AWK-scripts for:

- Building corpus files in the Infomap format
- Combine the different corpora and settings to build the vector models
- Run the synonym test for each model and store the result

Similar experiments has been done using LSI and SVD on "The Test Of English as a Foreign Language" (TOEFL) with a reported correctness of 64%. [3] Slightly better results have been reported using random indexing instead of SVD. [9] Much better results for the TOEFL test have been reported by Turney [17] using "Pointwise Mutual Information", but he used a huge training set - namely the Web indexed by Alta Vista. Note also that our goal is not to compete in solving synonym tests, but using them for evaluating comparable LSVMs.

## 3. PARAMETER SETTINGS

In this section we will describe the parameters and the selected set of values to use.

### 3.1 Corpus choice

We believe a lot in the old corpus linguist's Mantra "there's no data like more data" since LSI rely a lot on redundancy. Another important thing to keep in mind is that our evaluation set contains a lot of very rare words in newspaper text. On the other hand, this is just an evaluation and not a goal in itself. We will now describe the corpora used. The term "types" is used for distinct words and "tokens" for running words.

#### *3.1.1 Newspapers*

We have a collection of around one million newspaper articles, in total 500 million tokens. From this collection, parts of various size are created. In this kind of collection we can observe the fact that the number of types can be very high in a Swedish corpus compared to an English one, where compounds are written with space between the parts. The number of word types in this corpus is more than 3 million types. The corpus named *Clef* is a part of the collection containing 30 million tokens, which has been used as evaluation data by the Cross-Language Evaluation Forum (CLEF), and *Clef10M* is a 10 million tokens portion of *Clef*.

#### *3.1.2 Bring thesaurus*

Bring is a Swedish thesaurus from 1930. It is inspired by Roget's thesaurus [13] and just like Roget, it has around 1000 main categories containing a main word and groups of related nouns, verbs, and adjectives. We have not used these structures at all so each category is counted as one document. In total, Bring contains 60 000 types and 140 000 tokens.

[3] In this article multi-word unit means just a sequence of words. For more information, see appendix

[4] Bengt Johansson was a successful coach for the Swedish handball team in 1988-2004

[5] Some parts of the program is rewritten to remove limitations due to RAM and Hard-disk usage. Our version can handle up to 100 times bigger matrices than the original one.

#### 3.1.3 *Lexin*

Lexin is a dictionary for language learners of Swedish. It contains 19 000 main words with short descriptions. Each main word and its description is counted as a document. In total 200 000 tokens.

#### 3.1.4 *The Bible*

This is just a machine readable version of the Swedish Bible. It contains 800 000 tokens and 50 000 types, divided into 1200 documents – one for each chapter.

#### 3.1.5 *Swedish Parole*

The Parole corpora is a result of a project financed by the European Union. This Swedish part contains newspaper text and novels, in total 20 million tokens and 600 000 types.

#### 3.1.6 *Combining corpora*

We have been using the corpora one by one for training but also combined them. One problem is that when using large newspaper corpora, the number of types becomes bigger than our software can handle, so if newspaper texts is combined with Lexin and/or Bring, which contain many of the important words for synonym tests that are missing in newspaper texts, many important words will be filtered away due to low frequency. More about this problem can be read in section 3.2.

### 3.2 Input matrix

Our revised version of the Infomap software uses a hash-table instead of the complete matrix, [6] but large vocabularies combined with a large context window still consume a lot of RAM memory, so it is in many cases impossible to use the complete vocabulary on the second dimension of the input matrix. In the result section, we use *Vocab* for vocabulary size and *CoVoc* for the number of cells in the other dimension of the matrix.

### 3.3 Context size

The context size decides how close two words have to occur to be seen as a co-occurrence. If the context size is large, all words in the current document will be seen as context words to each other. We have chosen to look at the values: 1000, 300, 100, 30, 10, and 3 words. The abbreviation *Con* is used in the result section.

### 3.4 Number of dimensions after projection

This is the number of dimensions in the new vector space obtained by the SVD. The number of dimensions is much lower than the vectors in the original co-occurrence matrix, but it is easy to calculate the vector in this space for any known word or combination of words, using the information from the SVD. It is reasonable to think that a bigger training corpus would need a higher dimensionality for all topics to fit into the vector space and earlier articles about LSI proposes a dimensionality from 50 to 500. We have chosen to look at: 30, 100, 300, 1000 and 3000. The abbreviation *Dim* is used in the result section.

### 3.5 Combining the parameters

A new computer performance problem arises when combining all the different parameter settings. Apart from the different training corpora the different settings result in a lot of combinations, and with some compositions of training corpora, the number of LSVMs to train easily reaches thousands. Each training is rather computation intensive since it contains calculations on a huge matrix and also requires hundreds of megabytes of disk space to store the model. We did not have the computer power for all these calculations but exploring the parameter space works quite well even without all data points.

[6]In the hash-table we do not have to store all empty cells as in the complete matrix

## 4. NEW WAYS TO CALCULATE THE SUM OF MEANINGS

A better way to look up MWUs in a semantic vector space would include a better way to calculate the true meaning of an MWU but just adding the meaning of each word in the phrase. Unfortunately the vector spaces we get from standard SVD or RI contains just single words if we have not made a better tokenization of the training data before calculating the vector space. Therefore, without preparing the data in a different way before running SVD or RI, the only obvious way to get a vector from an MWU is the sum of the vectors corresponding to each word.

### 4.1 Preparation of data before SVD

The baseline tokenization of the training data is to just create one token for each word. Delimiters between words are spaces and other non-letters. This process is fast and since this work is a pilot study for future work with large corpora containing billions of tokens, even the more complicated tokenization models has to be fast. We are aiming on a system architecture with a total processing time in the magnitude of 24 hours on a decent workstation.

#### 4.1.1 *The baseline tokenizer*

Even the baseline tokenizer (BT) contains some difficulties. We need to recognize acronyms which may include '.' inside tokens. Other non-trivial tokens are dates, intervals, number (including spaces and/or a decimal point), etc. But, the baseline tokenizer will not try to find things like proper names, compounds [7] or fixed phrases.

#### 4.1.2 *The tuple extended tokenizer*

This tokenizer (TUP) will take one more step from BT. The resulting string of BT-tokens will go through a tuple expander, adding all n-grams up to a specific length. If we for example use the max length four we get: *Former president Bill Clinton* $\rightarrow$ {Former_president_Bill_Clinton, president_Bill_Clinton, Former_president_Bill, Former_-president, president_Bill, Bill_Clinton, Former, president, Bill, Clinton} The result will contain separate vectors for all these new tokens consisting of up to four words each. There is a risk of overtraining since the same BT-token will be a part of many TUP-tokens, but every original word is a part of the same number of tuples so we do not think this will be a problem. Compared to a noun phrase chunker or parser, TUP has the strength that it is language independent and very fast. A chunker or parser may also lead to overtraining if we keep both the chunk and its components. Just keeping the chunks is not a good idea if we want to be able to use either the search term *Former president Bill Clinton* or just *Bill Clinton.*

[7]Since we primarily work with Swedish, compounds are not such a big problem. They are written without spaces between the parts, like in German.

#### 4.1.3 *Other preparations*

Another preprocessing issue is whether we should keep upper and lower case character or just convert everything to lower case. Some other preprocessing steps should be tested, but they are not a part of this paper. To keep the possibility to work on large corpora, they have to scale up well for huge text collections. Some preprocessings to try could be: stemming, lemmatization and compound splitting.

## 5. PRESENTATION OF THE HP440 EVALUATION SET

The evaluation set is based on a Swedish synonym test (called ORD) from "Högskoleprovet" (an entrance test for university studies). There is a new test twice a year and our collection contains the exercises from 11 tests from the years 1998-2004, in total 440 queries with five alternative answers each.

### 5.1 Phrases in the queries

This test is not just a synonym test containing single words. Some of the query terms, and a bigger proportion of the answers contain phrases. The query terms contain in average 1.17 tokens with a maximum of four tokens, but only 10.9% of the query terms contain more than one token. For the answers the average number of tokens is 1.61, maximum is 10, and 35.5% contain more than one token.

### 5.2 About the words

One should note that ORD in "Högskoleprovet" is rather difficult even for university students. The average result for these tests was 62% among people who are trying to qualify for university studies. Many of the words are rare and/or old fashioned, and some are idioms like for example *hugget som stucket* (means *whatever* or word by word: *cut as stung*). Especially the multi-word idioms should confuse an ordinary LSVM.

## 6. EVALUATION RESULTS

Since it has been practically impossible to calculate all combinations of LSVMs we had to explore the parameter space with partial information. On the other hand we had the possibility to calculate any specific LSVM we wanted. Here we will go through the parameter space, dimension by dimension, but still keep the other dimensions in mind since they may interfere with each other. Note also that the baseline, which chooses randomly among the five answers, has an expected result 20% (88 correct answers out of the 440). Note that that the second result column in the tables shows the absolute number of correct answers.

### 6.1 Upper and lower case

The context size and the number of dimensions is here set to 100 and *CoVoc* is 10 000 (10k), so let us take a look at the results for a few corpora in table 1.

There is no significant difference between *keep* and *lower*, but the fact that the vocabulary gets smaller if we use lower case, and some initial experiments in the document retrieval domain, we choose to use lower case from now.

### 6.2 Verify that bigger Vocab is better

We will now go on with *Case* set to lower, *Dim* and *Con* to 100, and *CoVoc* to 10k, and try some different values for *Vocab* in table 2.

**Table 1: Results for the two case options**

| Corpus | Vocab | Case | Result | |
|---|---|---|---|---|
| Bring+Lexin | 94k * | keep | 48.41% | 213 |
| Bring+Lexin | 94k * | lower | 48.64% | 214 |
| Bring+Clef10M | 327k * | keep | 27.05% | 119 |
| Bring+Clef10M | 315k * | lower | 26.59% | 117 |
| Clef | 617k * | keep | 28.64% | 126 |
| Clef | 590k * | lower | 31.14% | 137 |

- The first value in the result-field is the percentage of correct answers, and the second, the absolute number of correct answers
- The '*' marks that *Vocab* is the complete vocabulary for this corpus

**Table 2: Results for *Vocab***

| Corpus | Vocab | Result | |
|---|---|---|---|
| Bring+Lexin | 10k | 28.41% | 125 |
| Bring+Lexin | 30k | 45.23% | 199 |
| Bring+Lexin | 94k * | 48.64% | 214 |
| Bring+Clef10M | 10k | 21.59% | 95 |
| Bring+Clef10M | 30k | 22.27% | 98 |
| Bring+Clef10M | 100k | 27.27% | 120 |
| Bring+Clef10M | 300k | 26.14% | 115 |
| Bring+Clef10M | 327k * | 26.59% | 117 |

There is a clearly significant difference between 94k and 10k vocabulary for Bring+Lexin, and for Bring+Clef10M we get a 3.0 $\chi$2-value which is almost significant on the 95% level. Therefore we conclude that a bigger vocabulary is better.

### 6.3 Find an optimal CoVoc value

In table 3 *Case* is set to lower, *Dim* and *Con* to 100. Now we will test som values for *CoVoc*.

For Bring+Clef10M the differences are not significant, [8] and for Bring+Lexin it is not very clear either. However, the difference between 190 (*CoVoc*=1k) and 214 (*CoVoc*=10k) correct answers is very close to significant at the 90% level according to the $\chi$2-value 2.64. When we raises *CoVoc* above 10k, the results get slightly worse.

[8]The bigger CoVoc settings did not work with the Bring+Clef10M corpus due to the enormous matrix size obtained.

**Table 3: Results for *CoVoc***

| Corpus | Vocab | CoVoc | Result | |
|---|---|---|---|---|
| Bring+Lexin | 94k * | 1k | 43.18% | 190 |
| Bring+Lexin | 94k * | 3k | 47.05% | 207 |
| Bring+Lexin | 94k * | 10k | 48.64% | 214 |
| Bring+Lexin | 94k * | 30k | 46.82% | 206 |
| Bring+Lexin | 94k * | 100k | 45.23% | 199 |
| Bring+Clef10M | 327k * | 1k | 25.68% | 113 |
| Bring+Clef10M | 327k * | 3k | 25.68% | 113 |
| Bring+Clef10M | 327k * | 10k | 26.59% | 117 |

Table 4: Results for *Dim*

| Corpus | Vocab | Dim | Result | |
|---|---|---|---|---|
| Bring+Lexin | 94k * | 30 | 36.82% | 162 |
| Bring+Lexin | 94k * | 100 | 48.64% | 214 |
| Bring+Lexin | 94k * | 300 | 57.27% | 252 |
| Bring+Lexin | 94k * | 900 | 59.77% | 263 |
| Bring+Lexin | 94k * | 1000 | 59.55% | 262 |
| Bring+Lexin | 94k * | 3000 | 54.09% | 238 |
| Bring+Clef10M | 327k * | 30 | 24.77% | 109 |
| Bring+Clef10M | 327k * | 100 | 26.59% | 117 |
| Bring+Clef10M | 327k * | 300 | 26.82% | 118 |

Table 5: Results for *Con*

| Corpus | Vocab | Con | Result | |
|---|---|---|---|---|
| Bring+Lexin | 94k * | 3 | 38.64% | 170 |
| Bring+Lexin | 94k * | 10 | 40.91% | 180 |
| Bring+Lexin | 94k * | 30 | 41.82% | 184 |
| Bring+Lexin | 94k * | 100 | 48.64% | 214 |
| Bring+Lexin | 94k * | 300 | 48.18% | 212 |
| Bring+Lexin | 94k * | 1000 | 47.95% | 211 |
| Bring+Clef10M | 327k * | 3 | 27.05% | 119 |
| Bring+Clef10M | 327k * | 10 | 28.18% | 124 |
| Bring+Clef10M | 327k * | 30 | 28.86% | 127 |
| Bring+Clef10M | 327k * | 100 | 26.59% | 117 |
| Bring+Clef10M | 327k * | 300 | 26.82% | 118 |
| Bring+Clef10M | 327k * | 1000 | 26.14% | 115 |

## 6.4 Find an optimal value for Dim

*Case* is set to lower, *Con* to 100, and *CoVoc* to 10k. In table 4 we have the results for some different values for *Dim*.

As usual, the Bring+Clef10M does not give a significant difference, [9] but Bring+Lexin is clearly better with *Dim*=-1000 than *Dim*=100 and also for *Dim*=100 than for *Dim*=30. We also get the $\chi$2-value 2.67 between *Dim*=1000 and *Dim*=-3000, which is very close to significant at the 90% level. So for Bring+Lexin it seems to be a level for *Dim* where we get a maximum, and the the result gets worse for higher *Dim*. This is in line with the articles about LSI. [5, 7, 10]

## 6.5 Find an optimal value for Con

*Case* is set to lower, *Dim* to 100, and *CoVoc* to 10k. In table 5 we have the results for some different values for *Con*.

Between *Con*=3 and *Con*=100, we get a significant difference at the 99% level from the $\chi$2-value, but for Bring+-Clef10M, there are no significant differences. One should also note that the context may never stretch over document borders, which makes *Con*=300 and *Con*=1000 similar if the documents are not very long.

## 6.6 Find the optimal corpus composition

There are a lot of possible combinations with the available corpora. We will try the ones in table 6 with *Vocab* set to fit all the word types. *Case* is set to lower, *Dim* and *Con* to 100, and *CoVoc* to 10k.

The best results seem to come from LSVMs trained with

[9]We did not manage to run on Bring+Clef10M with more than 300 dimensions, and we did not bother to take the time to find the problem. 300 dimensions gave no improvement over 100, so 1000 and 3000 did not feel important to test.

Table 6: Results for different corpora

| Corpus | Vocab | Result | |
|---|---|---|---|
| Bring | 52k * | 46.82% | 206 |
| Bible | 29k * | 22.50% | 99 |
| Clef | 590k * | 31.14% | 137 |
| Lexin | 57k * | 38.18% | 168 |
| Bring+Lexin | 93k * | 48.41% | 213 |
| Bring+Lexin+Bible | 108k * | 32.50% | 143 |

Table 7: Results after preprocessing

| Preprocessing | Vocab | Dim | Result | |
|---|---|---|---|---|
| Only words | 93k * | 100 | 48.41% | 213 |
| Only words | 93k * | 300 | 57.27% | 252 |
| Only words | 93k * | 1000 | 59.55% | 262 |
| Only words | 93k * | 3000 | 54.09% | 238 |
| Words+bigrams | 323k * | 100 | 43.18% | 190 |
| Words+bigrams | 323k * | 300 | 60.00% | 264 |
| Words+bigrams | 323k * | 1000 | 61.59% | 271 |
| Words+bigr+trigr | 551k * | 300 | 56.82% | 250 |
| Words+bigr+trigr | 551k * | 1000 | 62.73% | 276 |

Bring and Lexin put together. Bring alone gives a little bit lower results (not significant), followed by Lexin alone (significantly worse). The Bible is almost as bad as the baseline.

## 6.7 Adding bi- and trigrams

In table 7 we have tried to add first bigrams, and then also trigrams. The corpus used is Bring+Lexin, *Case* is set to lower, *Con* to 100, and *CoVoc* to 10k.

The differences between *Only words* and the models with ngrams added is not significant, but it seems like *Words + pairs + triples* actually is the best. We can also note that the optimal number of dimensions is much higher for the models with

## 6.8 Overall results

We have seen that with the right combination of parameters, the model with bi- and trigrams added seems to outperform the ordinary model trained just using single words. The best result, 62.73% (70.1% for known words) is better than the average student, taking the word synonym, which is quite fun, even if it is not very meaningful to compare these figures with the ones for humans.

In total, all query and answer terms contain fully 4000 tokens from which 39% are unknown to the best model trained with Bring and Lexin, and the most difficult ones are of course when the query term is unknown.

## 7. FUTURE WORK

Future work will include some parallel activities. The most important ones for us are described here.

## 7.1 Handling MWUs

Another strategy than the ngram adding, we want to try is to use an extremely fast dependency parser developed by Joakim Nivre and the MALT [10] group at Växjö University

[10]MALT stands for Models and Algorithms for Language Technology

in Sweden. [12] The parser will give the MWUs to add. Hopefully, this approach will not suffer from the overtraining problems, and it will definitely result in shorter training time. Some kind of collocation algorithms should also be tested to find the important MWUs.

## 7.2 Use the evaluations to find a better model

When we have optimized and fine tuned the LSI package, and the various MWU handling methods are developed, we will try to optimize the settings for an LSVM useful in search engines. The evaluation used will be both the HP440 set and a document retrieval set with documents, queries, and relevance judgements, developed in Borås, Sweden. [1]

## 8. CONCLUSION

Using the almost naive approach to add bi- and trigrams to the vocabulary before training an LSVM, we have observed better performance in our evaluation based on synonym tests. A bigger evaluation set would probably give a significant difference. Even if the difference is not very big (from 59.55% to 62.73%) the result is important, since an LSVM with bi- and trigram makes it possible to search for words and MWUs containing up to three words without making the model work worse for single words.

## 9. REFERENCES

[1] P. Ahlgren. *The effects of indexing strategy-query term combination on retrieval effectiveness in a Swedish full text database.* PhD thesis, University College of Borås and Göteborg University, 2004.

[2] M. W. Berry. Large-scale sparse singular value computations. *The International Journal of Supercomputer Applications*, 6(1):13–49, Spring 1992.

[3] M. W. Berry, S. T. Dumais, and T. A. Letsche. Computational methods for intelligent information access. 1995.

[4] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[5] S. T. Dumais. Using lsi for information filtering: Trec-3 experiments. In *The Third Text REtrieval Conference (TREC3).* National Institute of Standards and Technology, 1995.

[6] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'88*, 1988.

[7] J. I. Hong. An overview of latent semantic indexing. In *SIMS 240*, 2000.

[8] A. Hulth. *Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction.* PhD thesis, Stockholm University, 2004.

[9] J. Karlgren and M. Sahlgren. From words to understanding. *Foundations of Real-world Intelligence*, pages 294–308, 2001.

[10] T. K. Landauer and S. T. Dumais. A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240, 1997.

[11] S. Leach. Singular value decomposition - a primer.

[12] J. Nivre. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, pages 149–160, 2003.

[13] T. E. of the American Heritage Dictionaries, editor. *Roget's II The New Thesaurus.* Houghton Mifflin, 2003.

[14] Y. Qiu and H.-P. Frei. Concept-based query expansion. In *Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval*, pages 160–169, Pittsburgh, US, 1993.

[15] M. Sahlgren, P. Hansen, and J. Karlgren. Sics at clef 2002: Automatic query expansion using random indexing. In *Advances in Cross-Language Information Retrieval, Third Workshop of the Cross-Language Evaluation Forum, CLEF 2002*, Rome, 2002.

[16] Telcordia. Telcordia tm latent semantic indexing software (lsi): Beyond keyword retrieval. Technical report, Telcordia Techologies, 2003.

[17] P. D. Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *EMCL '01: Proceedings of the 12th European Conference on Machine Learning*, pages 491–502, London, UK, 2001. Springer-Verlag.

## APPENDIX

**Definitions of used terms**

- **Corpus**: Linguists have many different definitions of a corpus. In this paper, it is just a collection of text of some sort
- **HP440**: The new evaluation set based on the synonym tests from 11 instances of the Swedish "Högskoleprovet", resulting in 440 queries
- **LSI/LSA**: Latent Semantic Indexing/Analysis is the way to use a projected vector space to find latent similarities between vectors that would be impossible to find in the original spacious vector space because too many vector pairs are orthogonal
- **LSVM**: A Vector Model trained from a corpus using latent semantic indexing and singular value decomposition
- **MWU**: Multi-Word Unit means just a sequence of words. In some cases they are phrases of some kind, and sometimes names like "Bob Dylan", but they could also be just some words somewhere in a text
- **SVD**: Singular Value Decomposition is an algorithm to find the best [11] projection of a high dimensional vector space down to any specific dimensionality
- **Tokens**: Number of running words in a corpus
- **Types**: Number of distinct word types in a corpus

[11] Best in the sense that it keeps distances from the original vector space as good as possible

# Feature Representation for Effective Action-Item Detection

Paul N. Bennett
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
pbennett+@cs.cmu.edu

Jaime Carbonell
Language Technologies Institute.
Carnegie Mellon University
Pittsburgh, PA 15213
jgc+@cs.cmu.edu

## ABSTRACT

E-mail users face an ever-growing challenge in managing their inboxes due to the growing centrality of email in the workplace for task assignment, action requests, and other roles beyond information dissemination. Whereas Information Retrieval and Machine Learning techniques are gaining initial acceptance in spam filtering and automated folder assignment, this paper reports on a new task: automated *action-item detection*, in order to flag emails that require responses, and to highlight the specific passage(s) indicating the request(s) for action. Unlike standard topic-driven text classification, action-item detection requires inferring the sender's intent, and as such responds less well to pure bag-of-words classification. However, using enriched feature sets, such as $n$-grams (up to n=4) with chi-squared feature selection, and contextual cues for action-item location improve performance by up to 10% over unigrams, using in both cases state of the art classifiers such as SVMs with automated model selection via embedded cross-validation.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.6 [**Artificial Intelligence**]: Learning; I.5.4 [**Pattern Recognition**]: Applications

## General Terms

Experimentation

## Keywords

Text classification, speech acts, feature selection, e-mail, $n$-grams, SVMs

## 1. INTRODUCTION

E-mail users are facing an increasingly difficult task of managing their inboxes in the face of mounting challenges that result from rising e-mail usage. This includes prioritizing e-mails over a range of sources from business partners to family members, filtering and reducing junk e-mail, and quickly managing requests that demand the receiver's attention or action. Automated *action-item detection* targets the third of these problems by attempting to detect which e-mails *require* an action or response with information, and within those e-mails, attempting to highlight the sentence (or other passage length) that directly indicates the action request.

Such a detection system can be used as one part of an e-mail agent which would assist a user in processing important e-mails quicker than would have been possible without the agent. We view action-item detection as one necessary component of a successful e-mail agent which would perform spam detection, action-item detection, topic classification and priority ranking, among other functions. The utility of such a detector can manifest as a method of prioritizing e-mails according to task-oriented criteria other than the standard ones of topic and sender or as a means of ensuring that the email user hasn't dropped the proverbial ball by forgetting to address an action request.

Action-item detection differs from standard text classification in two important ways. First, the user is interested both in detecting whether an email contains action items and in locating exactly where these action item requests are contained within the email body. In contrast, standard text categorization merely assigns a topic label to each text, whether that label corresponds to an e-mail folder or a controlled indexing vocabulary [12, 15, 22]. Second, action-item detection attempts to recover the email sender's intent — whether she means to elicit response or action on the part of the receiver; note that for this task, classifiers using only unigrams as

From: Henry Hutchins <hhutchins@innovative.company.com>
To: Sara Smith; Joe Johnson; William Woolings
Subject: meeting with prospective customers
Sent: Fri 12/10/2005 8:08 AM

Hi All,
I'd like to remind all of you that the group from GRTY will be visiting us next Friday at 4:30 p.m. The current schedule looks like this:
+ 9:30 a.m. Informal Breakfast and Discussion in Cafeteria
+ 10:30 a.m. Company Overview
+ 11:00 a.m. Individual Meetings (Continue Over Lunch)
+ 2:00 p.m. Tour of Facilities
+ 3:00 p.m. Sales Pitch
In order to have this go off smoothly, I would like to practice the presentation well in advance. *As a result, I will need each of your parts by Wednesday.*
Keep up the good work!
–Henry

**Figure 1: An E-mail with emphasized Action-Item, an *explicit* request that requires the recipient's attention or action.**

features do not perform optimally, as evidenced in our results below. Instead we find that we need more information-laden features such as higher-order $n$-grams. Text categorization by topic, on the other hand, works very well using just individual words as features [2, 9, 13, 17]. In fact, genre-classification, which one would think may require more than a bag-of-words approach, also works quite well using just unigram features [14]. Topic detection and tracking (TDT), also works well with unigram feature sets [1, 20]. We believe that action-item detection is one of the first clear instances of an IR-related task where we must move beyond bag-of-words to achieve high performance, albeit not too far, as bag-of-$n$-grams seem to suffice.

We first review related work for similar text classification problems such as e-mail priority ranking and speech act identification. Then we more formally define the action-item detection problem, discuss the aspects that distinguish it from more common problems like topic classification, and highlight the challenges in constructing systems that can perform well at the sentence and document level. From there, we move to a discussion of feature representation and selection techniques appropriate for this problem and how standard text classification approaches can be adapted to smoothly move from the sentence-level detection problem to the document-level classification problem. We then conduct an empirical analysis that helps us determine the effectiveness of our feature extraction procedures as well as establish baselines for a number of classification algorithms on this task. Finally, we summarize this paper's contributions and consider interesting directions for future work.

## 2. RELATED WORK

Several other researchers have considered very similar text classification tasks. Cohen et al. [5] describe an ontology of "speech acts", such as "Propose a Meeting", and attempt to predict when an e-mail contains one of these speech acts. We consider action-items to be an important specific type of speech act that falls within their more general classification. While they provide results for several classification methods, their methods only make use of human judgments at the document-level. In contrast, we consider whether accuracy can be increased by using finer-grained human judgments that mark the specific sentences and phrases of interest.

Corston-Oliver et al. [6] consider detecting items in e-mail to "Put on a To-Do List". This classification task is very similar to ours except they do not consider "simple factual questions" to belong to this category. We include questions, but note that not all questions are action-items — some are rhetorical or simply social convention, "How are you?". From a learning perspective, while they make use of judgments at the sentence-level, they do not explicitly compare what if any benefits finer-grained judgments offer. Additionally, they do not study alternative choices or approaches to the classification task. Instead, they simply apply a standard SVM at the sentence-level and focus primarily on a linguistic analysis of how the sentence can be logically reformulated before adding it to the task list. In this study, we examine several alternative classification methods, compare document-level and sentence-level approaches and analyze the machine learning issues implicit in these problems.

Interest in a variety of learning tasks related to e-mail has been rapidly growing in the recent literature. For example, in a forum dedicated to e-mail learning tasks, Culotta et al. [7] presented methods for learning social networks from e-mail. In this work, we do not focus on peer relationships; however, such methods could complement those here since peer relationships often influence word choice when requesting an action.

## 3. PROBLEM DEFINITION & APPROACH

In contrast to previous work, we explicitly focus on the benefits that finer-grained, more costly, sentence-level human judgments offer over coarse-grained document-level judgments. Additionally, we consider multiple standard text classification approaches and analyze both the quantitative and qualitative differences that arise from taking a document-level vs. a sentence-level approach to classification. Finally, we focus on the representation necessary to achieve the most competitive performance.

### 3.1 Problem Definition

In order to provide the most benefit to the user, a system would not only detect the document, but it would also indicate the specific sentences in the e-mail which contain the action-items. Therefore, there are three basic problems:

1. *Document detection*: Classify a document as to whether or not it contains an action-item.
2. *Document ranking*: Rank the documents such that all documents containing action-items occur as high as possible in the ranking.
3. *Sentence detection*: Classify each sentence in a document as to whether or not it is an action-item.

As in most Information Retrieval tasks, the weight the evaluation metric should give to precision and recall depends on the nature of the application. In situations where a user will eventually read all received messages, ranking (*e.g.,* via precision at recall of 1) may be most important since this will help encourage shorter delays in communications between users. In contrast, high-precision detection at low recall will be of increasing importance when the user is under severe time-pressure and therefore will likely not read all mail. This can be the case for crisis managers during disaster management. Finally, sentence detection plays a role in both time-pressure situations and simply to alleviate the user's required time to gist the message.

### 3.2 Approach

As mentioned above, the labeled data can come in one of two forms: a *document-labeling* provides a yes/no label for each document as to whether it contains an action-item; a *phrase-labeling* provides only a yes label for the specific items of interest. We term the human judgments a *phrase-labeling* since the user's view of the action-item may not correspond with actual sentence boundaries or predicted sentence boundaries. Obviously, it is straightforward to generate a document-labeling consistent with a phrase-labeling by labeling a document "yes" if and only if it contains at least one phrase labeled "yes".

To train classifiers for this task, we can take several viewpoints related to both the basic problems we have enumerated and the form of the labeled data. The *document-level* view treats each e-mail as a learning instance with an associated class-label. Then, the document can be converted to a feature-value vector and learning progresses as usual. Applying a document-level classifier to document detection and ranking is straightforward. In order to apply it to sentence detection, one must make additional steps. For example, if the classifier predicts a document contains an action-item, then areas of the document that contain a high-concentration of words which the model weights heavily in favor of action-items can be indicated. The obvious benefit of the document-level approach is that training set collection costs are lower since the user only has to specify whether or not an e-mail contains an action-item and not the specific sentences.

In the *sentence-level* view, each e-mail is automatically segmented into sentences, and each sentence is treated as a learning instance with an associated class-label. Since the phrase-labeling provided by the user may not coincide with the automatic segmentation, we must determine what label to assign a partially overlapping sentence when converting it to a learning instance. Once trained, applying the resulting classifiers to sentence detection is now straightforward, but in order to apply the classifiers to document detection and document ranking, the individual predictions over each sentence must be aggregated in order to make a document-level prediction. This approach has the potential to benefit from more-specific labels that enable the learner to focus attention on the key sentences instead of having to learn based on data that the majority of the words in the e-mail provide no or little information about class membership.

### *3.2.1 Features*

Consider some of the phrases that might constitute part of an action item: "would like to know", "let me know", "as soon as possible", "have you". Each of these phrases consists of common words that occur in many e-mails. However, when they occur in the same sentence, they are far more indicative of an action-item. Additionally, order can be important: consider "have you" versus "you have". Because of this, we posit that $n$-grams play a larger role in this problem than is typical of problems like topic classification. Therefore, we consider all $n$-grams up to size 4.

When using $n$-grams, if we find an $n$-gram of size 4 in a segment of text, we can represent the text as just one occurrence of the $n$-gram or as one occurrence of the $n$-gram and an occurrence of each smaller $n$-gram contained by it. We choose the second of these alternatives since this will allow the algorithm itself to smoothly back-off in terms of recall. Methods such as naïve Bayes may be hurt by such a representation because of double-counting.

Since sentence-ending punctuation can provide information, we retain the terminating punctuation token when it is identifiable. Additionally, we add a *beginning-of-sentence* and *end-of-sentence* token in order to capture patterns that are often indicators at the beginning or end of a sentence. Assuming proper punctuation, these extra tokens are unnecessary, but often e-mail lacks proper punctuation. In addition, for the sentence-level classifiers that use $n$-grams, we additionally code for each sentence a binary encoding of the *position* of the sentence relative to the document. This encoding has eight associated features that represent which octile (the first eighth, second eighth, *etc.*) contains the sentence.

### *3.2.2 Implementation Details*

In order to compare the document-level to the sentence-level approach, we compare predictions at the document-level. We do not address how to use a document-level classifier to make predictions at the sentence-level.

In order to automatically segment the text of the e-mail, we use the RASP statistical parser [4]. Since the automatically segmented sentences may not correspond directly with the phrase-level boundaries, we treat any sentence that contains at least 30% of a marked action-item segment as an action-item. When evaluating sentence-detection for the sentence-level system, we use these class labels as ground truth. Since we are not evaluating multiple segmentation approaches, this does not bias any of the methods. If multiple segmentation systems were under evaluation, one would need to use a metric that matched predicted positive sentences to phrases labeled positive. The metric would need to punish overly long true predictions as well as too short predictions. Our criteria for converting to labeled instances implicitly includes both criteria. Since the segmentation is fixed, an overly long prediction would be predicting "yes" for many "no" instances since presumably the extra length corresponds to additional segmented sentences all of which do not contain 30% of action-item. Likewise, a too short prediction must correspond to a small sentence included in the action-item but not constituting all of the action-item. Therefore, in order to consider the prediction to be too short, there will be an additional preceding/following sentence that is an action-item where we incorrectly predicted "no".

Once a sentence-level classifier has made a prediction for each sentence, we must combine these predictions to make both a document-level prediction and a document-level score. We use the simple policy of predicting positive when any of the sentences is predicted positive. In order to produce a document score for ranking, the confidence that the document contains an action-item is:

$$\psi(d) = \begin{cases} \frac{1}{n(d)} \sum_{s \in d | \pi(s)=1} \psi(s) & \text{if for any } s \in d, \pi(s) = 1 \\ \frac{1}{n(d)} \max_{s \in d} \psi(s) & o.w. \end{cases}$$

where $s$ is a sentence in document $d$, $\pi$ is the classifier's 1/0 prediction, $\psi$ is the score the classifier assigns as its confidence that $\pi(s) = 1$, and $n(d)$ is the greater of 1 and the number of (unigram) tokens in the document. In other words, when any sentence is predicted positive, the document score is the length normalized sum of the sentence scores above threshold. When no sentence is predicted positive, the document score is the maximum sentence score normalized by length. As in other text problems, we are more likely to emit false positives for documents with more words or sentences. Thus we include a length normalization factor.

## 4. EXPERIMENTAL ANALYSIS

## 4.1 The Data

Our corpus consists of e-mails obtained from volunteers at an educational institution and cover subjects such as: organizing a research workshop, arranging for job-candidate interviews, publishing proceedings, and talk announcements. The messages were anonymized by replacing the names of each individual and institution with a pseudonym.[1] After attempting to identify and eliminate duplicate e-mails, the corpus contains 744 e-mail messages.

After identity anonymization, the corpora has three basic versions. *Quoted material* refers to the text of a previous e-mail that an author often leaves in an e-mail message when responding to the e-mail. Quoted material can act as noise when learning since it may include action-items from previous messages that are no longer relevant. To isolate the effects of quoted material, we have three versions of the corpora. The *raw* form contains the basic messages. The *auto-stripped* version contains the messages after quoted material has been automatically removed. The *hand-stripped* version contains the messages after quoted material has been removed by a human. Additionally, the hand-stripped version has had any xml content and e-mail signatures removed — leaving only the essential content of the message. The studies reported here are performed with the hand-stripped version. This allows us to balance the cognitive load in terms of number of tokens that must be read in the user-studies we report — including quoted material would complicate the user studies since some users might skip the material while others read it. Additionally, ensuring all quoted material is removed

[1]We have an even more highly anonymized version of the corpus that can be made available for some outside experimentation. Please contact the authors for more information on obtaining this data.

prevents tainting the cross-validation since otherwise a test item could occur as quoted material in a training document.

### 4.1.1 Data Labeling

Two human annotators labeled each message as to whether or not it contained an action-item. In addition, they identified each segment of the e-mail which contained an action-item. A segment is a contiguous section of text selected by the human annotators and may span several sentences or a complete phrase contained in a sentence. They were instructed that an action item is "an explicit request for information that requires the recipient's attention or a required action" and told to "highlight the phrases or sentences that make up the request".

| | | Annotator 1 | |
|---|---|---|---|
| | | No | Yes |
| Annotator 2 | No | 391 | 26 |
| | Yes | 29 | 298 |

**Table 1: Agreement of Human Annotators at Document Level**

Annotator One labeled 324 messages as containing action items. Annotator Two labeled 327 messages as containing action items. The agreement of the human annotators is shown in Tables 1 and 2. The annotators are said to *agree at the document-level* when both marked the same document as containing no action-items or both marked at least one action-item regardless of whether the text segments were the same. At the document-level, the annotators agreed 93% of the time. The kappa statistic [3, 5] is often used to evaluate inter-annotator agreement:

$$\kappa = \frac{A - R}{1 - R}$$

$A$ is the empirical estimate of the probability of **a**greement. $R$ is the empirical estimate of the probability of **r**andom agreement given the empirical class priors. A value close to $-1$ implies the annotators agree far less often than would be expected randomly, while a value close to 1 means they agree more often than randomly expected.

At the document-level, the kappa statistic for inter-annotator agreement is $0.85$. This value is both strong enough to expect the problem to be learnable and is comparable with results for similar tasks [5, 6].

In order to determine the sentence-level agreement, we use each judgment to create a sentence-corpus with labels as described in Section 3.2.2, then consider the agreement over these sentences. This allows us to compare agreement over "no judgments". We perform this comparison over the hand-stripped corpus since that eliminates spurious "no" judgments that would come from including quoted material, etc. Both annotators were free to label the subject as an action-item, but since neither did, we omit the subject line of the message as well. This only reduces the number of "no" agreements. This leaves 6301 automatically segmented sentences. At the sentence-level, the annotators agreed 98% of the time, and the kappa statistic for inter-annotator agreement is $0.82$.

In order to produce one single set of judgments, the human annotators went through each annotation where there was disagreement and came to a consensus opinion. The annotators did not collect statistics during this process but anecdotally reported that the majority of disagreements were either cases of clear annotator oversight or different interpretations of conditional statements. For example, *"If you would like to keep your job, come to tomorrow's meeting"* implies a required action where *"If you would like to join the football betting pool, come to tomorrow's meeting"* does not. The first would be an action-item in most contexts while the second would not. Of course, many conditional statements are not so clearly interpretable. After reconciling the judgments there are 416 e-mails with no action-items and 328 e-mails containing action-items. Of the 328 e-mails containing action-items, 259 messages have one action-item segment; 55 messages have two action-item segments; 11 messages have three action-item segments. Two messages have four action-item segments, and one message has six action-item segments. Computing the sentence-level agreement using the reconciled "gold standard" judgments with each of the annotators' individual judgments gives a kappa of $0.89$ for Annotator One and a kappa of $0.92$ for Annotator Two.

| | | Annotator 1 | |
|---|---|---|---|
| | | No | Yes |
| Annotator 2 | No | 5810 | 65 |
| | Yes | 74 | 352 |

**Table 2: Agreement of Human Annotators at Sentence Level**

In terms of message characteristics, there were on average 132 content tokens in the body after stripping. For action-item messages, there were 115. However, by examining Figure 2 we see the length distributions are nearly identical. As would be expected for e-mail, it is a long-tailed distribution with about half the messages having more than 60 tokens in the body (this paragraph has 65 tokens).

## 4.2 Classifiers

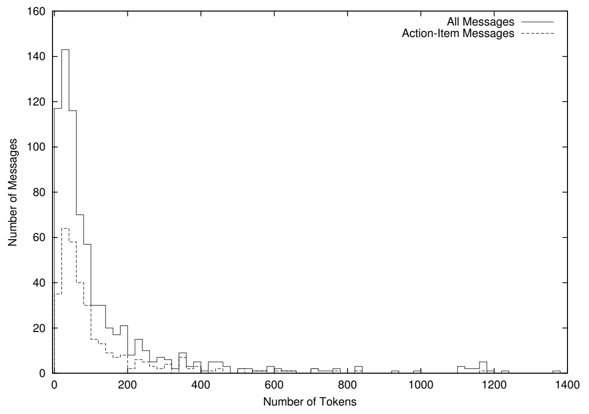
For this experiment, we have selected a variety of standard text classification algorithms. In selecting algorithms, we have chosen algorithms that are not only known to work well but which differ along such lines as discriminative vs. generative and lazy vs. eager. We have done this in order to provide both a competitive and thorough sampling of learning methods for the task at hand. This is important since it is easy to improve a strawman classifier by introducing a new representation. By thoroughly sampling alternative classifier choices we demonstrate that representation improvements over bag-of-words are not due to using the information in the bag-of-words poorly.

### 4.2.1 kNN

We employ a standard variant of the $k$-nearest neighbor algorithm used in text classification, kNN with $s$-cut score thresholding [19]. We use a tfidf-weighting of the terms with a distance-weighted vote of the neighbors to compute the score before thresholding it. In order to choose the value of $s$ for thresholding, we perform leave-one-out cross-validation over the training set. The value of $k$ is set to be $2(\lceil \log_2 N \rceil + 1)$ where $N$ is the number of training points. This rule for choosing $k$ is theoretically motivated by results which show such a rule converges to the optimal classifier as the number of training points increases [8]. In practice, we have also found it to be a computational convenience that frequently leads to comparable results with numerically optimizing $k$ via a cross-validation procedure.

### 4.2.2 Naïve Bayes

We use a standard multinomial naïve Bayes classifier [16]. In using this classifier, we smoothed word and class probabilities using a Bayesian estimate (with the word prior) and a Laplace m-estimate, respectively.



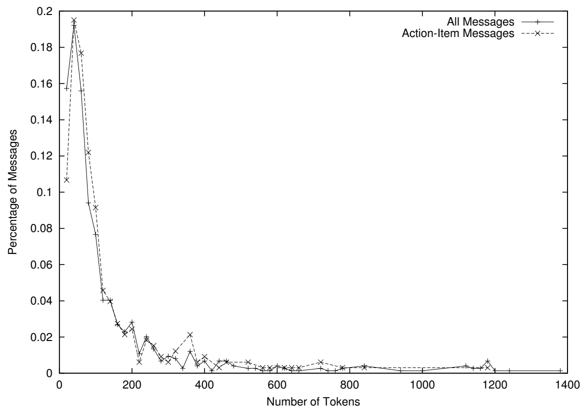


**Figure 2: The Histogram (left) and Distribution (right) of Message Length. A bin size of 20 words was used. Only tokens in the body after hand-stripping were counted. After stripping, the majority of words left are usually actual message content.**

| | Classifiers | Document Unigram | Document Ngram | Sentence Unigram | Sentence Ngram |
|---|---|---|---|---|---|
| F1 | kNN | 0.6670 ± *0.0288* | 0.7108 ± *0.0699* | 0.7615 ± *0.0504* | **0.7790** ± *0.0460* |
| | naïve Bayes | 0.6572 ± *0.0749* | 0.6484 ± *0.0513* | 0.7715 ± *0.0597* | **0.7777** ± *0.0426* |
| | SVM | 0.6904 ± *0.0347* | 0.7428 ± *0.0422* | 0.7282 ± *0.0698* | **0.7682** ± *0.0451* |
| | Voted Perceptron | 0.6288 ± *0.0395* | 0.6774 ± *0.0422* | 0.6511 ± *0.0506* | **0.6798** ± *0.0913* |
| Accuracy | kNN | 0.7029 ± *0.0659* | 0.7486 ± *0.0505* | 0.7972 ± *0.0435* | **0.8092** ± *0.0352* |
| | naïve Bayes | 0.6074 ± *0.0651* | 0.5816 ± *0.1075* | 0.7863 ± *0.0553* | **0.8145** ± *0.0268* |
| | SVM | 0.7595 ± *0.0309* | 0.7904 ± *0.0349* | 0.7958 ± *0.0551* | **0.8173** ± *0.0258* |
| | Voted Perceptron | 0.6531 ± *0.0390* | **0.7164** ± *0.0376* | 0.6413 ± *0.0833* | 0.7082 ± *0.1032* |

**Table 3: Average Document-Detection Performance during Cross-Validation for Each Method and the Sample Standard Deviation ($S_{n-1}$) in italics. The best performance for each classifier is shown in bold.**

#### *4.2.3 SVM*

We have used a linear SVM with a tfidf feature representation and L2-norm as implemented in the SVM$^{light}$ package v6.01 [11]. All default settings were used.

#### *4.2.4 Voted Perceptron*

Like the SVM, the Voted Perceptron is a kernel-based learning method. We use the same feature representation and kernel as we have for the SVM, a linear kernel with tfidf-weighting and an L2-norm. The voted perceptron is an online-learning method that keeps a history of past perceptrons used, as well as a weight signifying how often that perceptron was correct. With each new training example, a correct classification increases the weight on the current perceptron and an incorrect classification updates the perceptron. The output of the classifier uses the weights on the perceptra to make a final “voted” classification. When used in an offline-manner, multiple passes can be made through the training data. Both the voted perceptron and the SVM give a solution from the same hypothesis space — in this case, a linear classifier. Furthermore, it is well-known that the Voted Perceptron increases the margin of the solution after each pass through the training data [10]. Since Cohen et al. [5] obtain worse results using an SVM than a Voted Perceptron with one training iteration, they conclude that the best solution for detecting speech acts may not lie in an area with a large margin. Because their tasks are highly similar to ours, we employ both classifiers to ensure we are not overlooking a competitive alternative classifier to the SVM for the basic bag-of-words representation.

## 4.3 Performance Measures

To compare the performance of the classification methods, we look at two standard performance measures, F1 and accuracy. The F1 measure [18, 21] is the harmonic mean of precision and recall where $Precision = \frac{Correct\ Positives}{Predicted\ Positives}$ and $Recall = \frac{Correct\ Positives}{Actual\ Positives}$.

## 4.4 Experimental Methodology

We perform standard 10-fold cross-validation on the set of documents. For the sentence-level approach, all sentences in a document are either entirely in the training set or entirely in the test set for each fold. For significance tests, we use a two-tailed t-test [21] to compare the values obtained during each cross-validation fold with a p-value of 0.05.

Feature selection was performed using the chi-squared statistic. Different levels of feature selection were considered for each classifier. Each of the following number of features was tried: $10, 25, 50, 100, 250, 750, 1000, 2000, 4000$. There are approximately 4700 unigram tokens without feature selection. In order to choose the number of features to use for each classifier, we perform nested cross-validation and choose the settings that yield the optimal document-level F1 for that classifier. For this study, only the body of each e-mail message was used. Feature selection is always applied to all candidate features. That is, for the $n$-gram representation, the $n$-grams and position features are also subject to removal by the feature selection method.

## 4.5 Results

The results for document-level classification are given in Table 3. The primary hypothesis we are concerned with is that $n$-grams are critical for this task; if this is true, we expect to see a significant gap in performance between the document-level classifiers that use $n$-grams (denoted *Document Ngram*) and those using only unigram features (denoted *Document Unigram*). Examining Table 3, we observe that this is indeed the case for every classifier except naïve Bayes. This difference in performance produced by the $n$-gram representation is statistically significant for each classifier except for naïve Bayes and the accuracy metric for kNN (see Table 4). Naïve Bayes poor performance with the $n$-gram representation is not surprising since the bag-of-$n$-grams causes excessive double-counting as mentioned in Section 3.2.1; however, naïve Bayes is not hurt at the sentence-level because the sparse examples provide few chances for agglomerative effects of double counting. In either case, when a language-modeling approach is desired, modeling the $n$-grams directly would be preferable to naïve Bayes. More importantly for the $n$-gram hypothesis, the $n$-grams lead to the best document-level classifier performance as well.

As would be expected, the difference between the *sentence-level* $n$-gram representation and unigram representation is small. This is because the window of text is so small that the unigram representation, when done at the sentence-level, implicitly picks up on the power of the $n$-grams. Further improvement would signify that the order of the words matter even when only considering a small sentence-size window. Therefore, the finer-grained sentence-level judgments allows a unigram representation to succeed but only when performed in a small window — behaving as an $n$-gram representation for all practical purposes.

| | Document Winner | Sentence Winner |
|---|---|---|
| kNN | **Ngram** | Ngram |
| naïve Bayes | Unigram | Ngram |
| SVM | **Ngram**$^{\dagger}$ | Ngram |
| Voted Perceptron | **Ngram**$^{\dagger}$ | Ngram |

**Table 4: Significance results for $n$-grams versus unigrams for document detection using document-level and sentence-level classifiers. When the F1 result is statistically significant, it is shown in bold. When the accuracy result is significant, it is shown with a $^{\dagger}$.**

| | F1 Winner | Accuracy Winner |
|---|---|---|
| kNN | **Sentence** | **Sentence** |
| naïve Bayes | **Sentence** | **Sentence** |
| SVM | Sentence | **Sentence** |
| Voted Perceptron | Sentence | Document |

**Table 5: Significance results for sentence-level classifiers vs. document-level classifiers for the document detection problem. When the result is statistically significant, it is shown in bold.**

Further highlighting the improvement from finer-grained judgments and $n$-grams, Figure 3 graphically depicts the edge the SVM sentence-level classifier has over the standard bag-of-words approach with a precision-recall curve. In the high precision area of the graph, the consistent edge of the sentence-level classifier is rather impressive — continuing at precision 1 out to 0.1 recall. This would mean that a tenth of the user's action-items would be placed at the top of their action-item sorted inbox. Additionally, the large separation at the top right of the curves corresponds to the area where the optimal F1 occurs for each classifier, agreeing with the large improvement from 0.6904 to 0.7682 in F1 score. Considering the relative unexplored nature of classification at the sentence-level, this gives great hope for further increases in performance.

| | Accuracy | | F1 | |
|---|---|---|---|---|
| | Unigram | Ngram | Unigram | Ngram |
| kNN | 0.9519 | 0.9536 | 0.6540 | 0.6686 |
| naïve Bayes | 0.9419 | 0.9550 | 0.6176 | 0.6676 |
| SVM | 0.9559 | 0.9579 | 0.6271 | 0.6672 |
| Voted Perceptron | 0.8895 | 0.9247 | 0.3744 | 0.5164 |

**Table 6: Performance of the Sentence-Level Classifiers at Sentence Detection**

Although Cohen et al. [5] observed that the Voted Perceptron with a single training iteration outperformed SVM in a set of similar tasks, we see no such behavior here. This further strengthens the evidence that an alternate classifier with the bag-of-words representation could not reach the same level of performance. The Voted Perceptron classifier does improve when the number of training iterations are increased, but it is still lower than the SVM classifier.

Sentence detection results are presented in Table 6. With regard to the sentence detection problem, we note that the $F1$ measure gives a better feel for the remaining room for improvement in this difficult problem. That is, unlike document detection where action-item documents are fairly common, action-item sentences are very rare. Thus, as in other text problems, the accuracy numbers are deceptively high sheerly because of the default accuracy attainable by always predicting "no". Although, the results here are significantly above-random, it is unclear what level of performance is necessary for sentence detection to be useful in and of itself and not simply as a means to document ranking and classification.

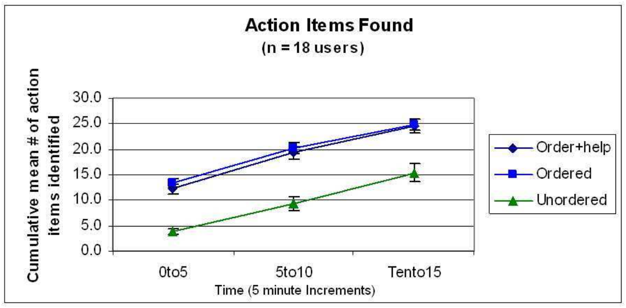


**Figure 4: Users find action-items quicker when assisted by a classification system.**

Finally, when considering a new type of classification task, one of the most basic questions is whether an accurate classifier built for the task can have an impact on the end-user. In order to demonstrate the impact this task can have on e-mail users, we conducted a user study using an earlier less-accurate version of the sentence classifier — where instead of using just a single sentence, a three-sentence *windowed-approach* was used. There were three distinct sets of e-mail in which users had to find action-items. These sets were either presented in a random order (*Unordered*), ordered by the classifier (*Ordered*), or ordered by the classifier and with the

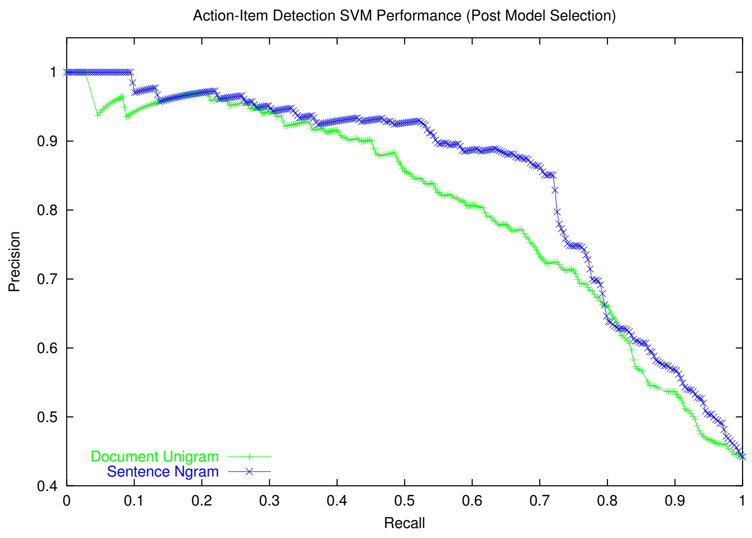


**Figure 3: Both $n$-grams and a small prediction window lead to consistent improvements over the standard approach.**

center sentence in the highest confidence window highlighted (*Order+help*). In order to perform fair comparisons between conditions, the overall number of tokens in each message set should be approximately equal; that is, the cognitive reading load should be approximately the same before the classifier's reordering. Additionally, users typically show "practice effects" by improving at the overall task and thus performing better at later message sets. This is typically handled by varying the ordering of the sets across users so that the means are comparable. While omitting further detail, we note the sets were balanced for the total number of tokens and a latin square design was used to balance practice effects.

Figure 4 shows that at intervals of 5, 10, and 15 minutes, users consistently found significantly more action-items when assisted by the classifier, but were most critically aided in the first five minutes. Although, the classifier consistently aids the users, we did not gain an additional end-user impact by highlighting. As mentioned above, this might be a result of the large room for improvement that still exists for sentence detection, but anecdotal evidence suggests this might also be a result of how the information is presented to the user rather than the accuracy of sentence detection. For example, highlighting the wrong sentence near an actual action-item hurts the user's trust, but if a vague indicator (*e.g.,* an arrow) points to the approximate area the user is not aware of the near-miss. Since the user studies used a three sentence window, we believe this played a role as well as sentence detection accuracy.

## 4.6 Discussion

In contrast to problems where $n$-grams have yielded little difference, we believe their power here stems from the fact that many of the meaningful $n$-grams for action-items consist of common words, *e.g.*, "let me know". Therefore, the document-level unigram approach cannot gain much leverage, even when modeling their joint probability correctly, since these words will often co-occur in the document but not necessarily in a phrase. Additionally, action-item detection is distinct from many text classification tasks in that a single sentence can change the class label of the document. As a result, good classifiers cannot rely on aggregating evidence from a large number of weak indicators across the entire document.

Even though we discarded the header information, examining the top-ranked features at the document-level reveals that many of the features are names or parts of e-mail addresses that occurred in the body and are highly associated with e-mails that tend to contain many or no action-items. A few examples are terms such as "org", "bob", and "gov". We note that these features will be sensitive to the particular distribution (senders/receivers) and thus the document-level approach may produce classifiers that transfer less readily to alternate contexts and users at different institutions. This points out that part of the problem of going beyond bag-of-words may be the methodology, and investigating such properties as learning curves and how well a model transfers may highlight differences in models which appear to have similar performance when tested on the distributions they were trained on. We are currently investigating whether the sentence-level classifiers do perform better over different test corpora without retraining.

## 5. FUTURE WORK

While applying text classifiers at the document-level is fairly well-understood, there exists the potential for significantly increasing the performance of the sentence-level classifiers. Such methods include alternate ways of combining the predictions over each sentence, weightings other than tfidf, which may not be appropriate since sentences are small, better sentence segmentation, and other types of phrasal analysis. Additionally, named entity tagging, time expressions, *etc.*, seem likely candidates for features that can further improve this task. We are currently pursuing some of these avenues to see what additional gains these offer.

Finally, it would be interesting to investigate the best methods for combining the document-level and sentence-level classifiers. Since the simple bag-of-words representation at the document-level leads to a learned model that behaves somewhat like a context-specific prior dependent on the sender/receiver and general topic, a first choice would be to treat it as such when combining probability estimates with the sentence-level classifier. Such a model might serve as a general example for other problems where bag-of-words can establish a baseline model but richer approaches are needed to achieve performance beyond that baseline.

## 6. SUMMARY AND CONCLUSIONS

The effectiveness of sentence-level detection argues that labeling at the sentence-level provides significant value. Further experiments are needed to see how this interacts with the amount of training data available. Sentence detection that is then agglomerated to document-level detection works surprisingly better given low recall than would be expected with sentence-level items. This, in turn, indicates that improved sentence segmentation methods could yield further improvements in classification.

In this work, we examined how action-items can be effectively detected in e-mails. Our empirical analysis has demonstrated that $n$-grams are of key importance to making the most of document-level judgments. When finer-grained judgments are available, then a standard bag-of-words approach using a small (sentence) window size and automatic segmentation techniques can produce results almost as good as the $n$-gram based approaches.

## Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA), or the Department of Interior-National Business Center (DOI-NBC).

We would like to extend our sincerest thanks to Jill Lehman whose efforts in data collection were essential in constructing the corpus, and both Jill and Aaron Steinfeld for their direction of the HCI experiments. We would also like to thank Django Wexler for constructing and supporting the corpus labeling tools and Curtis Huttenhower's support of the text preprocessing package. Finally, we gratefully acknowledge Scott Fahlman for his encouragement and useful discussions on this topic.

## 7. REFERENCES

[1] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, Washington, D.C., 1998.

[2] C. Apte, F. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, July 1994.

[3] J. Carletta. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

[4] J. Carroll. High precision extraction of grammatical relations. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 134–140, 2002.

[5] W. W. Cohen, V. R. Carvalho, and T. M. Mitchell. Learning to classify email into "speech acts". In *EMNLP-2004 (Conference on Empirical Methods in Natural Language Processing)*, pages 309–316, 2004.

[6] S. Corston-Oliver, E. Ringger, M. Gamon, and R. Campbell. Task-focused summarization of email. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 43–50, 2004.

[7] A. Culotta, R. Bekkerman, and A. McCallum. Extracting social networks and contact information from email and the web. In *CEAS-2004 (Conference on Email and Anti-Spam)*, Mountain View, CA, July 2004.

[8] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York, NY, 1996.

[9] S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98, Proceedings of the 7th ACM Conference on Information and Knowledge Management*, pages 148–155, 1998.

[10] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[11] T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. J. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 41–56. MIT Press, 1999.

[12] L. S. Larkey. A patent search and classification system. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, pages 179 – 187, 1999.

[13] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *SIGIR '92, Proceedings of the 15th Annual International ACM Conference on Research and Development in Information Retrieval*, pages 37–50, 1992.

[14] Y. Liu, J. Carbonell, and R. Jin. A pairwise ensemble approach for accurate genre classification. In *Proceedings of the European Conference on Machine Learning (ECML)*, 2003.

[15] Y. Liu, R. Yan, R. Jin, and J. Carbonell. A comparison study of kernels for multi-label text classification using category association. In *The Twenty-first International Conference on Machine Learning (ICML)*, 2004.

[16] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *Working Notes of AAAI '98 (The 15th National Conference on Artificial Intelligence), Workshop on Learning for Text Categorization*, pages 41–48, 1998. TR WS-98-05.

[17] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, March 2002.

[18] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[19] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):67–88, 1999.

[20] Y. Yang, J. Carbonell, R. Brown, T. Pierce, B. T. Archibald, and X. Liu. Learning approaches to topic detection and tracking. *IEEE EXPERT, Special Issue on Applications of Intelligent Information Retrieval*, 1999.

[21] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR '99, Proceedings of the 22nd Annual International ACM Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.

[22] Y. Yang, J. Zhang, J. Carbonell, and C. Jin. Topic-conditioned novelty detection. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2002.

# Predicting Extraction Performance using Context Language Models

Eugene Agichtein Silviu Cucerzan
Microsoft Research, Redmond, WA, USA
{eugeneag, silviu}@microsoft.com

## ABSTRACT

Exploiting lexical and semantic relationships in text can dramatically improve information retrieval accuracy. Most notably, named entities and relations between entities are crucial for effective question answering and other information retrieval tasks. Unfortunately, the success in extracting these relationships can vary for different domains and document collections. Predicting extraction performance is an important step towards integration of information extraction technology for high accuracy information retrieval. In this paper, we present a general language modeling method for quantifying the difficulty of information extraction tasks. We demonstrate the viability of our approach by predicting extraction performance of two real world tasks, Named Entity Recognition and Relation Extraction.

## Categories and Subject Descriptors

**H.3.1 [INFORMATION STORAGE AND RETRIEVAL]: Information Search and Retrieval**

**General Terms** Algorithms, Experimentation.

**Keywords** Language modeling, information extraction, named entity extraction, relation extraction, context language modeling.

## 1. OVERVIEW

The vast majority of text available online is still primarily accessible via keyword matching at the document level. Unfortunately, this approach largely ignores the underlying lexical and semantic relationships between terms, which can be exploited to answer questions, and, more generally, to better satisfy the informational needs of users. The retrieval relevance could be improved if we detect meaningful terms (e.g., named entities such as dates, persons, organizations, and locations), and related entities (e.g., pairs of entities such as "*person*'s birth *date*" and "*person* who invented a *device*") and use them to answer questions directly.

However, real collections can exhibit properties that make them difficult for information extraction tasks. At the same time, tuning an information extraction system for a given collection, or porting an information extraction system to a new language, can require significant human and computational effort. Hence, predicting if an extraction task will be successful (i.e., the required information can be extracted with high accuracy) is extremely important for adapting, deploying, and maintaining information extraction systems, and ultimately, for accurate information retrieval.

The goal of this paper is to develop a lightweight method for *predicting* the accuracy of information extraction for a given task and document collection. This could be an efficient way to estimate the expected success and cost of tuning a full-featured information extraction system *before* running expensive experiments. These predictions can be useful when adapting an IE system to a new task, to a new language, or to a new document collection.

We observe that document collection properties, such as typical text contexts surrounding the entities or relation tuples, can affect difficulty of an extraction task. In turn, we may be able to predict the extraction performance on this task. In this paper, we present a first general approach to use context language models for predicting whether an extraction task will succeed for a given document collection.

More specifically, we will consider two information extraction tasks that are of paramount importance to information retrieval: *Named Entity Recognition*, and *Relation Extraction*.

- Named Entity Recognition (NER) is a task of identifying entities such as "Person", "Organization", and "Location" in text. The ability to identify such entities has been established as an important pre-processing task in several areas including information extraction, machine translation, information retrieval, and question answering. NER often serves as an important step in the Relation Extraction task described next.
- Relation Extraction (RE), is a task of identifying semantic relationships between entities in the text, such as "*person*'s birth *date*", which relates a person name in the text to a date that is the person's birth date. Once the tuples for this relation (e.g., <"Albert Einstein", "14 March 1879">) are identified, they can be used to directly answer questions such as "When was Albert Einstein born?"

*SIGIR'05 Workshop on Methodologies and Evaluation of Lexical Cohesion Techniques in Real-world Applications: Beyond Bag of Words (ELECTRA)*, August 19, 2005, Salvador, Brazil.

Most state of the art NER and RE systems rely on local context to identify entities or determine the relationship between target entities. In NER, contextual patterns such as "*Mr.*" or "*mayor of*" are crucial to hypothesizing occurrences of entities and classifying such identified entities, especially when they are polysemous or of a foreign origin. The local context is also extremely important for the RE task. Intuitively, if the context surrounding the entities of interest for a given relation looks very similar to the general text of the documents (i.e., there are no consistent and obvious "clues" that the entities or relationships of interest are present), then the RE task for that relation will be hard. While NER systems can resort to dictionary lookups in some cases (e.g., for the "Location" entities, dictionaries can be particularly helpful), for others (e.g., people's names or organizations) high accuracy may not be possible. In contrast, if the text context around entities in the collection tends to contain telltale clues, such as "Mr." preceding a person name, the extraction task is expected to be easier, and higher accuracy achievable.

Our approach formalizes and exploits this intuitive observation by building two *language models* for a collection–a task-specific *context language model* for the extraction task, and a *background* model for the collection. We can then compare the two models and compute the divergence of the context model from the background model. If the divergence is high (i.e., the context language model is different from the background model), the extraction task is expected to be easier than if the divergence was low (i.e., the context language model is similar to the background language model).

Interestingly, our task-specific language models may be helpful for other applications, including term weighting for information retrieval, and supporting active learning for interactive information extraction. For example, we could derive improved term weighs for specific retrieval tasks such as birthday finding. We will discuss other promising future directions of this work in Section 5.

The rest of this paper is organized as follows. In the next section we review related work. In Section 3 we present our formal model and algorithms. In Section 4 we present our initial experimental results for NER and RE tasks over large document collections. In Section 5 we present our conclusions, and discuss the implications and future directions of this work.

## 2. RELATED WORK

Our work explores language modeling for information extraction and thus touches on areas of information retrieval, information extraction, and language modeling. In this section we briefly review related work in these areas.

Our approach is largely inspired by the work of Cronen-Townsend, Zhou, and Croft [7] on predicting query performance by measuring the similarity of a language model $LM_Q$ derived from the retrieved documents for a query and a language model for the whole target collection of documents $LM_{Coll}$. Using simple unigram language models, they showed that the relative entropy between the query and collection language models correlates with the average precision in several TREC test collections. In this paper, we apply a similar language modeling technique to the task of predicting information extraction performance.

Language modeling, typically expressed as the problem of predicting the occurrence of a word in text or speech, has been an active area of research in speech recognition, optical character recognition, context-sensitive spelling, and machine translation. An in-depth analysis of this problem in natural language processing is presented in [12], Chapter 6. Language modeling has also been used to improve term weighting in information retrieval (e.g., [13] and others). However, in previous work LM was used as a tool for improving the specific system performance, whereas in our work we attempt to predict performance for general extraction tasks.

Using local context modeling has been previously used for IR tasks [18]. However, our work is different in that we only consider *task-specific* contexts. As our results indicate, using the locality in the overall document collection may not be sufficient, as local context models can become similar to the background model for overall document collection. Our model is similar in spirit to the use of entity language models described in [14] for classifying and retrieving named entities. A related language modeling approach was used for a different problem of predicting the reading difficulty of text for *human* readers [24]. A different approach in [25] uses the co-occurrence graph structure of the examples to predict accuracy of semi-supervised learning for semantic classification of phrases. Our work is complementary, as we present a general approach for modeling the performance of automatic systems on extraction tasks, including both named entity recognition and relation extraction.

For the named entity recognition task, numerous ways of exploiting local context were proposed, from relatively simple character-based models such as [8] and [11] to extremely complex models making use of various lexical, syntactic, morphological, orthographical information, such as [9] and [5]. In this work, we show that we can predict the difficulty of identifying several types of named entities by using relatively simple context language models. This study can be viewed as complementary to Collins' work [6] on the difficulty of identifying named entity boundaries, regardless of entity type.

Relation extraction systems rely on variety of features (e.g., syntactic, semantic, lexical, co-occurrence), but all depend heavily on context. Once the entities are identified, it is the textual context that expresses the relationship between the entities. Partially supervised relation extraction systems (e.g., [1], [10], [16], [20], and others) rely on the text contexts of example facts to derive extraction patterns.

For relation extraction, the task difficulty was previously analyzed by considering the complexity of the target extraction templates ([21] and [22]). Another promising approach described in [23] modeled the task domain variability by considering the different paraphrases used to express the same information in the text. In contrast, our work quantifies the difference between the contexts around the entities and unrelated text contexts. If the contexts of the example facts are similar to the background text an extraction system is expected to have more difficulty deriving extraction patterns and recognizing the relevant entities.

## 3. PREDICTING EXTRACTION DIFFICULTY

In this section we describe the general approach we take for modeling the difficulty of an extraction task, and hence the expected performance of an extraction system on the task (Section 3.1). Then, in Section 3.2, we describe the algorithms for computing the language models to make our predictions.

### 3.1 Model

As we discussed, the textual context, i.e., the local properties of the text surrounding the entities and relations of interest are of crucial importance to extraction accuracy. Intuitively, if the contexts in which the entities occur are very similar to the text contexts where the target entities do not occur, then extraction is expected to be difficult. Otherwise, if there are strong clues in a context, the extraction should be easier and we should expect higher extraction accuracy.

To quantify the notion of context, we use a basic unigram language model, which is essentially a probability distribution over the words in the text's vocabulary. In this study, we derive this probability distribution from the histogram of words occurring in the local context of target entities by using maximum likelihood estimation. Our purpose is to compare the language model associated with an entity type or relationship $LM_C$ with a background language model for the whole target text, denoted by $LM_{BG}$. Therefore, no smoothing of these models is necessary. Intuitively, if the background language model for the collection is very similar to the language model constructed from the context of the valid entities then the task is expected to be hard. Otherwise (if $LM_C$ is very different from $LM_{BG}$), the task is expected to be easier.

A common way to measure the difference between two probability distributions is relative entropy, also known as the Kullback-Leibler divergence:

$$\mathrm{KL}(LM_C \parallel LM_{BG}) = \sum_{w \in V} LM_C(w_i) \cdot \log \frac{LM_C(w)}{LM_{BG}(w)}$$

In Information Theory, KL-divergence represents the average number of bits wasted by encoding messages drawn from the distribution $LM_C$ using as model the distribution $LM_{BG}$.

Alternatively, we can measure how different two models are by using cosine similarity, which represents the cosine of the angle between the two language models seen as vectors in a multidimensional space in which each dimension corresponds to one word in the vocabulary:

$$\mathrm{Cosine}(LM_C, LM_{BG}) = \frac{< LM_C \cdot LM_{BG} >}{\parallel LM_{BG} \parallel_2 \cdot \parallel LM_C \parallel_2}$$

The closer the cosine is to 1, the smaller the angle and thus, the more similar the two models. Hence, to measure the difference of the two models $LM_C$ and $LM_{BG}$ we define *CDist* as:

$$CDist(LM_C \parallel LM_{BG}) = 1 - Cosine(LM_C, LM_{BG})$$

Which maintains the symmetry with the *KL* metric, with bigger values indicating larger difference between models.

### 3.2 Constructing the Language Models

We now describe how to construct a language model for a given extraction task. For clarity, we describe a unigram language model, but our methodology can be extended to higher-order features. For syntax-based extraction systems, we could parse the text and incorporate that information into the model as [4]. However, as we will show experimentally, our initial simple unigram model is sufficient to make useful predictions.

To construct the task-specific context language model $LM_C$ we search the collection for occurrences of valid entities (or relation tuples). While for the NER and RE tasks $LM_C$ is constructed slightly differently (as described below), the overall approach is to consider the text context to be the $K$ words to the right and to the left of the entity in question.

More specifically, the language model for NER is constructed as outlined in Figure 3.1. We scan the document collection D, searching for occurrences of each known entity $E_i$. When an entity is detected, we add to $LM_C$ up to $K$ terms to the right and to the left of the entity.

```
ConstructNERLanguageModel (Entities $E$, Documents $D, K$ )
For each document $d$ in $D$
    For each entity $E_i$ in $E$
        if $E_i$ is present in $d$
            For each instance of $E_i$ spanning from $start$ to $end$
                For each term $w$ in $d$ [$start - K$], …, $d$ [$start$-1]
                    Increment count of $w$ in $LM_C$
                For each term $w$ in $d$ [$end$ + 1], …, $d$ [$end + K$]
                    Increment count of $w$ in $LM_C$
Normalize $LM_C$
return $LM_C$
```

**Algorithm 3.1: NER Context language model construction.**

The algorithm for constructing a task-specific language model for RE is outlined in Figure 3.2. The procedure is similar to the NER algorithm above. We scan the document collection D, searching for occurrences of each known example tuple $T_i$ for the target relation. For this, we search for all attributes of $T_i$ in the text. If all entities are present, we add up to $K$ terms to the right of the leftmost entity, and up to $K$ terms to the left of the rightmost entity to $LM_C$. If the entities in a relation tuple are close together (i.e., there are fewer than $K$ words separating the entities in the text), we include all the terms separating the entities. Clearly, other variations of this algorithm are possible, and could be explored in future work.

```
ConstructRELanguageModel (Tuples $T$, Documents $D, K$ )
For each document $d$ in $D$
    For each tuple $T_i=(t_i^1,t_i^2)$ in $T$
        If $t_i^1$ or $t_i^2$ not present in $d$ continue
        For each pair of adjacent instances of $t_i^1,t_i^2$
        occurring at positions $start$ and $end$
            For each term $w$ in $d$ [$start - K$],…, $d$ [$start$-1]
                Increment count of $w$ in $LM_C$
            For each term $w$ in $d$ [$end$ + 1],…, $d$ [$end + K$]
                Increment count of $w$ in $LM_C$
Normalize $LM_C$
return $LM_C$
```

**Algorithm 3.2: RE Context language model construction.**

Unfortunately, we don't have all the valid entities available (i.e., when predicting whether a task will succeed without going through the complete extraction process). Hence, our model is build based on *sampling* the collection using a small (20-40) sample of the known entities or tuples by providing only these example entities as input to the NER and RE language model construction algorithms above. The sample-based model is expected to be a reasonable approximation of the complete task specific language model.

The background language model, $LM_{BG}$ is derived through maximum likelihood estimation using the word frequencies in each document collection. When we discard stopwords from $LM_C$ we also discard them from $LM_{BG}$.

In order to interpret the divergence of a task specific language model $LM_C$ from the background language model, we build a reference context language model $LM_R$ (also denoted as RANDOM). We construct $LM_R$, by taking random samples of words in the vocabulary (excluding stopwords) of the same size as the entity samples. We then use these words input to Algorithm 3.1. Using $LM_R$ we can then compute the "reference" divergence of a context language model from the background model for a given sample size. For large sample sizes, $LM_R$ is expected to approximate the background model. Indeed, Figure 3.1 reports that for larger random word sample sizes, $LM_R$ becomes more similar to the background model, and the divergence steadily decreases.

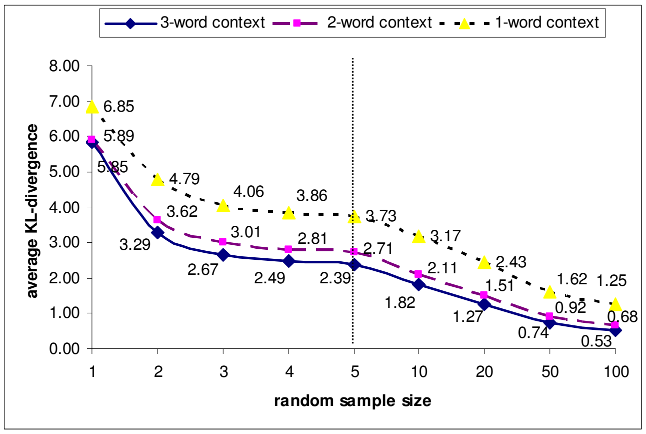


**Figure 3.1: The average KL-divergence between the context language models for random samples of words and the background language model.**

Constructing the context models $LM_C$ and $LM_R$ can be done efficiently by using any off-the-shelf search engine and considering only the documents retrieved by search for the example entities or tuples, and run Algorithms 3.1 and 3.2 only over these reduced document sets.

Having described constructing the language models for extraction tasks, we now turn to experimental evaluation.

## 4. EXPERIMENTS

We evaluated our prediction for two real-world tasks: Named Entity Recognition (NER) and Relation Extraction (RE). We first describe the experimental setup (Section 4.1), including the datasets, entity and relation types, and parameter settings we considered, as well as the methods for comparison. Then we describe our experiments for predicting NER difficulty (Section 4.2), followed by our experiments on predicting RE difficulty (Section 4.3).

## 4.1 Experimental Setup

In order to design a realistic evaluation we focused on two extraction tasks, NER and RE, over large document collections. The overall goal of the experiments is to determine if the language models, constructed from a realistically small sample of the extractions of interest, can make useful predictions about the observed accuracy of the extraction task for that collection.

The document collections used for these experiments are reported in Table 4.1. The Reuters RCV1 documents were drawn from the collection used in the CoNLL 2003 [17] NER shared task evaluation. For the RE experiments, we used a large online encyclopedia document collection.

| *Task* | *Collection* | *Size* |
|---|---|---|
| NER | Reuters RCV1, 1/100 | 3,566,125 words |
| | Reuters RCV1, 1/10 | 35,639,471 words |
| RE | Encyclopedia documents | 64,187,912 words |

**Table 4.1: Document collections used in experiments.**

For all experiments, we start with a small sample (20-40) of entities or relation tuples, drawn at random from a list of known valid entities or tuples. In Table 4.2 we report the size and composition of the samples used for the experiments.

| *Task* | *Sample Extractions (Description)* | *Size* |
|---|---|---|
| NER | Location names (LOC) | 20 |
| | Miscellaneous named entities (MISC) | 20 |
| | Organization names (ORG) | 20 |
| | Person names (PER) | 20 |
| RE | Person's birth dates (BORN) | 35 |
| | Person's death dates (DIED) | 35 |
| | Person's inventions (INVENT) | 35 |
| | Person's writings (WROTE) | 35 |

**Table 4.2: Entity and relations used in experiments.**

To validate our extraction performance predictions for the NER task, we used as reference the top performing systems in the CoNLL shared task competition, which were evaluated over a manually annotated subset of news articles from the same RCV1 corpus as described above. Moreover, we built the samples of named entities by randomly sampling the set of named entities present in the training set provided by the CoNLL competition organizers [17].

To validate our performance predictions for the RE task, we used a simple bootstrapping-based extraction system similar to Snowball [1], which is heavily dependent on both the example entities and the text context in which they appear to derive extraction patterns. The accuracy was computed by sampling the extracted relations. For comparison, we also report RANDOM, the divergence of the random keyword sample-based language model, $LM_R$.

In our experiments we explored the following parameters:

- Context size *K*: number of words to the left and to the right of entity to include as context.
- Divergence metric, *CDist* or *KL*: The language model similarity metrics defined in Section 3.1.
- Example set size *S*: number of randomly drawn entities (or relation tuples). Fixed sample size for each task between 20 and 40.
- Random sample size *R*: number of randomly drawn terms to estimate the background model. Fixed to match the value of *S* above for each task.
- *Stopwords*: we analyze two cases, when stopwords (common English words such as prepositions, conjunctions, numerals, etc.) are included the vocabulary and when they are excluded. In both cases, we discard punctuation.

## 4.2 Predicting NER Difficulty

In order to evaluate the accuracy of our predictions for the difficulty of extracting different types of named entities, we use as reference the accuracy of the top five systems in the CoNLL 2003 shared task competition [17], which are summarized in Table 4.3. It is also worth noting the performance of a baseline system that only identifies and labels entities that occurred in the training set. This baseline system obtains F-measure scores of 80.5 for LOC, 83.5 for MISC, 66.4 for ORG, and 55.2 for PER.

| | Florian et al. [9] | Chieu et al. [5] | Klein et al. [11] | Zhang et al. [19] | Carreras et al. [3] | Average |
|---|---|---|---|---|---|---|
| LOC | 91.15 | 91.12 | 89.98 | 89.54 | 89.26 | **90.21** |
| MISC | 80.44 | 79.16 | 80.15 | 75.87 | 78.54 | **78.83** |
| ORG | 84.67 | 84.32 | 80.48 | 80.46 | 79.41 | **81.86** |
| PER | 93.85 | 93.44 | 90.72 | 90.44 | 88.93 | **91.47** |
| Overall | 88.76 | 88.31 | 86.31 | 85.50 | 85.00 | 86.77 |

**Table 4.3: F-measures on the Reuters RCV1 collection reported by the top 5 systems participating in the CoNLL 2003 Shared Task competition.**

We report results on predicting NER difficulty in Tables 4.4a, 4.4b, and 4.5. The first two tables present the results obtained on a smaller subset of the Reuters corpus (3.5 million words), while the latter shows the results obtained for a bigger subset of the corpus (35 million words). It is remarkable that language models estimated on the smaller corpus are as good predictors as those estimated on a corpus 10-times bigger.

RCV1 1/100, Left/Right Context Size 1, Counting Stopwords

| | Sample 1 | Sample 2 | Sample 3 | Average |
|---|---|---|---|---|
| LOC | 1.47 | 1.54 | **1.49** | 1.50 |
| MISC | 1.31 | **2.09** | 2.29 | 1.89 |
| ORG | 4.36 | 2.25 | **4.12** | 3.57 |
| PER | 7.40 | 4.08 | **5.28** | 5.58 |
| RANDOM | **1.57** | 1.24 | 1.73 | 1.51 |

RCV1 1/100, Left/Right Context Size 2, Counting Stopwords

| | | | | |
|---|---|---|---|---|
| LOC | **1.12** | 1.15 | 1.11 | 1.12 |
| MISC | 0.94 | **1.46** | 1.62 | 1.34 |
| ORG | **3.60** | 1.85 | 3.85 | 3.10 |
| PER | 5.71 | 3.22 | **4.30** | 4.41 |
| RANDOM | 1.04 | 0.73 | **1.00** | 0.92 |

RCV1 1/100, Left/Right Context Size 3, Counting Stopwords

| | | | | |
|---|---|---|---|---|
| LOC | 0.92 | 0.94 | **0.93** | 0.93 |
| MISC | 0.78 | **1.18** | 1.33 | 1.09 |
| ORG | **2.95** | 1.65 | 3.45 | 2.68 |
| PER | 5.16 | 2.80 | **3.79** | 3.91 |
| RANDOM | 0.87 | 0.63 | **0.83** | 0.78 |

**Table 4.4a: KL-divergence for the context models for random samples of 20 entities/random words and the background language model for a corpus of 3.5 million words, when the language models include stopwords.**

RCV1 1/100, Left/Right Context Size 1, Ignoring Stopwords

| | Sample 1 | Sample 2 | Sample 3 | Average |
|---|---|---|---|---|
| LOC | 2.44 | 2.66 | **2.48** | 2.52 |
| MISC | 2.40 | **3.49** | 3.77 | 3.22 |
| ORG | **4.58** | 4.49 | 6.74 | 5.27 |
| PER | 9.08 | 6.23 | **7.61** | 7.64 |
| RANDOM | **2.35** | 2.11 | 2.84 | 2.43 |

RCV1 1/100, Left/Right Context Size 2, Ignoring Stopwords

| | | | | |
|---|---|---|---|---|
| LOC | 1.73 | 1.83 | **1.80** | 1.78 |
| MISC | 1.67 | **2.51** | 2.72 | 2.30 |
| ORG | **3.87** | 3.45 | 5.90 | 4.40 |
| PER | 8.03 | 4.87 | **5.91** | 6.27 |
| RANDOM | 1.68 | 1.22 | **1.62** | 1.50 |

RCV1 1/100, Left/Right Context Size 3, Ignoring Stopwords

| | | | | |
|---|---|---|---|---|
| LOC | 1.44 | 1.50 | **1.50** | 1.48 |
| MISC | 1.35 | **1.98** | 2.16 | 1.83 |
| ORG | **3.28** | 2.85 | 5.31 | 3.81 |
| PER | 7.19 | 4.32 | **5.36** | 5.62 |
| RANDOM | 1.43 | 1.05 | **1.33** | 1.27 |

**Table 4.4b: KL-divergence for the context models for random samples of 20 entities/random words and the background language model for a corpus of 3.5 million words, when the language models discard stopwords.**

Our ranking identifies ORG and PER entities as "easy" to extract entity types and LOC and MISC as hard to extract. These correlate with the results reported by the participants in the CoNLL 2003 Shared Task competition (Table 4.2), with one exception: the LOC entities. We believe this happens for two reasons: first, the location entities in the test set overlap to a large degree with the locations in the training data, as indicated by the performance of the baseline system; second, all systems shown in Table 4.3. except [11] used extensive lists of gazetteers, which were likely to contain most locations that news articles may talk about and thus, covering most of the locations in the test. A drawback of our model is that it does not take into account how easy to identify entities of a certain type based on intrinsic information (e.g., morphology) or gazetteer lists.

RCV1 1/10, Left/Right Context Size 1, Ignoring Stopwords

| | Sample 1 | Sample 2 | Sample 3 | Average |
|---|---|---|---|---|
| LOC | 1.65 | 1.82 | **1.81** | 1.76 |
| MISC | 1.79 | **2.75** | 2.99 | 2.51 |
| ORG | **4.36** | 2.93 | 5.48 | 4.25 |
| PER | 7.10 | 5.04 | **5.50** | 5.88 |
| RANDOM | **1.98** | 1.60 | 2.43 | 2.00 |

RCV1 1/10, Left/Right Context Size 2, Ignoring Stopwords

| | | | | |
|---|---|---|---|---|
| LOC | 1.12 | **1.22** | 1.26 | 1.20 |
| MISC | 1.16 | **1.84** | 2.02 | 1.67 |
| ORG | **3.57** | 2.18 | 4.33 | 3.36 |
| PER | 5.95 | 3.81 | **4.29** | 4.68 |
| RANDOM | 1.36 | 0.83 | **1.24** | 1.14 |

RCV1 1/10, Left/Right Context Size 3, Ignoring Stopwords

| | | | | |
|---|---|---|---|---|
| LOC | 0.92 | **0.99** | 1.04 | 0.98 |
| MISC | 0.91 | **1.41** | 1.55 | 1.29 |
| ORG | **2.94** | 1.79 | 3.76 | 2.83 |
| PER | 5.28 | 3.28 | **3.75** | 4.10 |
| RANDOM | 1.12 | 0.68 | **0.96** | 0.92 |

**Table 4.5: KL-divergence for the context models for random samples of 20 entities/random words and the background language model for a corpus of 35 million words, when the language models discard stopwords.**

Often, our predictions suggest a clear distinction between the four entity types considered. In most cases, the average KL-divergence value for one type of entities is greater than the maximum KL-divergence and smaller than the minimum KL-divergence obtained for any sample of another type of entities.

Tables 4.4a and 4.4b show the contrast between using stopwords in the language model and discarding the stopwords. As expected, the context language models are more similar to the background model when stopwords are included, but in both cases, the conclusions are essentially

| Relation \ K | *KL* (Section 3.1) 1 | 2 | 3 | 4 | 5 | *CDist* (Section 3.1) 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|
| INVENT | 4.33 | 3.76 | 3.55 | 3.33 | 3.3 | 0.24 | 0.2 | 0.16 | 0.12 | 0.13 |
| BORN | **8.68** | **7.62** | **6.86** | **6.4** | **6.16** | **0.86** | **0.74** | **0.58** | **0.54** | **0.54** |
| DIED | **7.72** | **7.72** | **6.87** | **6.49** | **6.75** | **0.62** | **0.66** | **0.49** | **0.42** | **0.41** |
| WROTE | 4.47 | 4.1 | 3.89 | 3.77 | 3.65 | 0.5 | 0.38 | 0.29 | 0.24 | 0.12 |
| RANDOM | 0.4 | 0.25 | 0.17 | 0.09 | 0.08 | 0.24 | 0.23 | 0.12 | 0.02 | 0.01 |

**Table 4.6: Predicting RE performance for the INVENT, BORN, DIED, and WROTE relations when the language models include stopwords.**

the same. This is encouraging, as it shows that this approach may work even for languages for which no lexical information (such as stopwords) is known *a priori*.

## 4.3 Predicting RE Performance

We now turn to predicting the performance for the relation extraction task (RE). The goal is to predict which relations are "hard" to extract, and which ones are "easy". Table 4.7 reports the actual extraction accuracy on the RE task using a simple bootstrapping-based information extraction system similar to Snowball [1] and KnowItAll [20]. We report the precision of the facts extracted by the system estimated by sampling 100 facts from the extracted relation instances. As we can see, the BORN and DIED relations are "easy" for the extraction system (exhibiting precision of as high as 97%), whereas INVENT and WROTE are relatively "hard" (exhibiting precision as low as 50%).

Table 4.6 reports the *KL* and *CDist* values computed from the models incorporating all words in the contexts. The *KL* divergence values of the BORN and DIED relations are significantly higher than the *KL* values for the INVENT and WROTE relations, predicting that the former should have higher accuracy than the latter. Hence, *KL* correctly identifies "easy" relations vs. "hard" relations to extract. On this task, the *CDist* divergence values for BORN and DIED are also noticeably higher than the corresponding values for INVENT and WROTE.

| *Relation* | *Accuracy* (%) *strict* | *partial* | *Task Difficulty* |
|---|---|---|---|
| INVENT | 0.35 | 0.64 | Hard |
| BORN | 0.73 | 0.96 | **Easy** |
| DIED | 0.34 | 0.97 | **Easy** |
| WROTE | 0.12 | 0.50 | Hard |

**Table 4.7: Precision for the RE task on the Encyclopedia collection for the INVENT, BORN, DIED, and WROTE relations.**

Table 4.8 reports the divergence values using the language models built by discarding common English stopwords. As we can see, these models have higher divergence from the background than the models with stopwords included. This is not surprising, as stopwords tend to appear in both generic and task specific contexts. As before, the context models built without stopwords have higher *KL* divergence from the background model for the "easy" relations than the *KL* divergence of the context models for the "hard" relations. In contrast, the *CDist* values for INVENT and DIED models now become more similar. In fact, the *CDist* values for INVENT are actually higher than the corresponding values for DIED, incorrectly suggesting that the INVENT extraction task is "easy". Hence, it appears that *KL* is a more robust predictor of extraction performance.

| Relation \ K | *KL* (Section 3.1) 1 | 2 | 3 | 4 | 5 | *CDist* (Section 3.1) 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|
| INVENT | 6.68 | 5.95 | 5.69 | 5.4 | 5.27 | 0.74 | 0.72 | **0.73** | **0.72** | **0.73** |
| BORN | **9.4** | **8.79** | **8.49** | **8.1** | **7.61** | **0.89** | **0.86** | **0.86** | **0.86** | **0.86** |
| DIED | **9.16** | **8.88** | **8.11** | **7.79** | **8.1** | **0.75** | **0.79** | 0.66 | 0.61 | 0.6 |
| WROTE | 6.72 | 5.95 | 5.82 | 5.71 | 5.48 | 0.62 | 0.61 | 0.61 | 0.62 | 0.61 |
| RANDOM | 0.36 | 0.22 | 0.18 | 0.13 | 0.12 | 0.4 | 0.24 | 0.2 | 0.14 | 0.13 |

**Table 4.8: Predicting RE performance for INVENT, BORN, DIED, and WROTE relations when the language models discard stopwords.**

## 5. CONCLUSIONS AND FUTURE WORK

We presented a context language modeling approach for predicting extraction performance. We have shown that our approach is effective for predicting extraction accuracy for tasks such as named entity recognition and relation extraction, both crucial for high accuracy and domain-specific information retrieval.

As our experiments indicate, starting with even a small sample of available entities can be sufficient for making a reasonable prediction about extraction accuracy. Our results are particularly encouraging as we consider a relatively simple model that does not require extra information to that typically available to modern NER and RE systems.

Extending our method to use more sophisticated language models (e.g., *n*-grams) can further improve our predictions. For languages where reliable NLP tools are available, one promising direction would be to incorporate syntactic features, and to apply techniques such as co-reference resolution to build richer and more accurate context language models. Additionally, incorporating gazetteer lists similar to those typically used by the NER systems can further improve prediction accuracy.

Furthermore, our techniques could be applied for building interactive information extraction systems that guide the user by requesting more examples for the extraction tasks predicted to be "hard".

As our experiments show, context language models localized around entities and relation instances of interest diverge from the document-level language model. Thus, for tasks such as question answering and information extraction, modeling term proximity near entities of interest is a promising direction for improving information retrieval accuracy.

### ACKNOWLEDGEMENTS
We thank Eric Brill for his encouragement and valuable suggestions, and the anonymous reviews for their insightful comments.

## 6. REFERENCES

[1] E. Agichtein and L. Gravano, Snowball: Extracting Relations from Large Plain-Text Collection, in *ACM DL 2000*

[2] E. Brill, S. Dumais, and M. Banko. An Analysis of the AskMSR Question-Answering System, in *EMNLP 2002*

[3] X. Carreras, L. Marquez, and L. Padro, A Simple Named Entity Extractor using AdaBoost, in *CoNLL 2003*

[4] C. Chelba and F. Jelinek, Exploiting Syntactic Structure for Language Modeling, in *COLING-ACL 1998*

[5] H. L. Chieu and H. T. Ng, Named Entity Recognition with a Maximum Entropy Approach, in *CoNLL 2003*

[6] M. Collins, Ranking Algorithms for Named Entity Extraction: Boosting and the Voted Perceptron, in *ACL 2002*

[7] S. Cronen-Townsend, Y. Zhou, W. B. Croft, Predicting Query Performance, in *SIGIR 2002*

[8] S. Cucerzan and D. Yarowsky, Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence, in *EMNLP-VLC 1999*

[9] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang, Named Entity Recognition through Classifier Combination, in *CoNLL 2003*

[10] R. Jones, A. McCallum, K. Nigam, and E. Riloff, Bootstrapping for Text Learning Tasks, in *IJCAI Workshop on Text Mining: Foundations, Techniques and Applications, 1999*

[11] D. Klein, J. Smarr, H. Nguyen, and C. Manning, Named Entity Recognition with Character-Level Models, in *CoNLL 2003*

[12] C. D. Manning and H. Schűtze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999

[13] J. M. Ponte and W. B. Croft, A Language Modeling Approach to Information, in *SIGIR 1998*

[14] H. Raghavan, J. Allan, and A. McCallum, An exploration of Entity Models, Collective Classification and Relation descriptions, in *LinkKDD* 2004

[15] D. Ravichandran and E. Hovy, Learning Surface Text Patterns for a Question Answering System, in *ACL 2002*

[16] E. Riloff and R. Jones, Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping, in *AAAI 1999*

[17] E.F. Tjong Kim Sang and F. De Meulder, Introduction to the CoNLL-2003 Shared Task, in *CoNLL 2003*

[18] J. Xu and W. B. Croft, Improving the effectiveness of information retrieval with local context analysis, in *ACM Transactions on Information Systems, 2000*

[19] T. Zhang and D. Johnson, A Robust Risk Minimization based Named Entity Recognition System, in *CoNLL 2003*

[20] O. Etzioni, M. Cafarella D. Downey, S. Kok, A. Popescul, T. Shaked, S. Soderland, D. Weld, and A. Yates, Web-scale information extraction in KnowItAll: preliminary results, in *WWW 2004*

[21] A. Bagga, Analyzing the complexity of a domain with respect to an information extraction task, in *MUC-7*, *1998*

[22] S. Huttunen, R. Yangarber, and R. Grishman, Complexity of event structure in IE scenarios, in *COLING 2002*

[23] I. Dagan and O. Glickman, Probabilistic textual entailment: generic applied modeling of language variability, *in Learning Methods for Text Understanding and Mining Workshop, 2004*

[24] K. Collins-Thompson and J. P. Callan, A language modeling approach to predicting reading difficulty, in *HLT, 2004*

[25] R. Jones, Semi-supervised Learning on Small Worlds, in *Link Discovery Workshop at KDD, 2004*

# POSTERS AND DEMOS

# A framework for detecting contextual concepts in texts

Ágnes Sándor
Xerox Research Centre Europe
6. chemin Maupertuis
38240 Meylan, France
+33-4-76-61-50-14
Agnes.Sandor@xrce.xerox.com

## ABSTRACT

Although information extraction systems often focus on assertions about concrete entities and relationships among them, there is growing interest in tools for identifying the contexts – meta-information - in which these assertions are made. Important elements of context include for example the source of the information, the author's attitude towards the information or his evaluation. We call recurring relevant contextual clues contextual concepts. The difficulty of the automatic processing of contextual concepts lies in the fact that usually there are no conventional ways, or set expressions, for conveying them. This yields a great variety of surface forms that traditional information access systems do not detect. We propose a framework to extract contextual concepts as the instantiations of general underlying patterns.

## General Terms

Reliability, Experimentation, Human Factors.

## Keywords

information access, contextual concept, concept matching, pattern matching

## 1. INTRODUCTION

Information extraction tools are usually developed to extract "facts" from free-text documents, as defined in the MUC tasks [2]. A "fact" in this framework is typically a relationship among a few concrete entities such as people, places, or objects. Facts, however, are presented in contexts that give them various statuses, which are relevant for a wide range of users: a researcher processing scientific facts described in the literature might want to know if those facts are old established facts (background knowledge) or new findings in the article; a customer wishing to buy an electronic device might want to inquire if it can be dangerous for health; a manufacturer searching for new products needs to know how a particular product compares with old ones. These kinds of relevant contextual contents are meta-information that the author might convey related to the facts he describes. There has been growing interest in developing automatic tools to detect such kinds of contextual clues to complement fact extraction in order to give access to their interpretation. We call recurring relevant contextual clues contextual concepts. The main difficulty in the extraction of contextual concepts lies in the diversity of the linguistic expressions conveying them.

The greatest effort to extract contextual concepts has been invested in the analysis of scientific writing [10,7,8,12,5] since the settings in which the facts are described follow some recurring patterns in scientific argumentation schemes. We do not know, however, of any attempt at the automatic extraction of contextual concepts like possible danger or a product presenting innovative technology compared to previous products.

The framework we propose has been successfully applied in a text mining project for detecting biological abstracts describing paradigm shifts in research [6]. Here we extend its application for extracting three additional contextual concepts: background knowledge, possible danger and innovation. Our goal with this presentation is to show that this framework is suited to handling new and diverse contextual concepts, which is regarded as a step for generalization. None of the three systems described here has been developed fully in large-scale applications like the paradigm shift system, so we cannot present evaluations. Their foundations, however, have been laid.

## 2. THE FRAMEWORK

Each of the following sentences conveys (through the expression in bold) one of our target concepts:

**a. Background knowledge**

**Recent studies** *indicate that ligands of the peroxisome proliferator-activated receptors-gamma (PPAR-gamma) alter cardiac remodeling during chronic ischemia.*

*AMH promoter sequence variations or the* **previously proposed** *SF3a2-AMH fusion co-transcripts cannot be responsible for aberrant AMH expression leading to Mullerian duct degradation.*

*Vascular endothelial growth factor (VEGF) is* **universally accepted** *as a primary factor in the regulation of vessel patency in vascular networks throughout the body and including the retina.*

**b. Danger**

*If the DNA repair mechanism does not work as well as it should, mutations in cells* **could accumulate with disastrous consequences**.

*Find out what you need to k***now** *to protect yourself from this* **potential cause of** *brain* **tumors**.

*Describes* **possible** *health* **risks posed** *by cell phones, including headaches, DNA damage, memory loss, ear ache, and fatigue.*

**c. Innovation**

*It delivers superb sensitivity and new insights into materials processes that* **cannot be obtained with conventional** *thermal analysis* **methods**.

*Video content in Macromedia Flash allows designers to maintain control of the look and feel of their applications,* **unlike today's existing** *video* **options** *which require external players to be launched and have platform inconsistencies.*

*A.D. Pharma has introduced an* **exciting new product unlike** *anything* **previously available**.

Our aim is to capture in each set the underlying patterns that allow human language processing to recognize the three concepts.

Previous efforts for extracting contextual concepts in scientific writing include that of Teufel [11], who proposes a method based on a combination of statistics and a pre-compiled lexicon. The method missed infrequent and discontinuous expressions due to the diversity of the surface forms. Ferret [3] developed a rule-based system, but the target expressions are collocations. This approach would not be suited to the type of expressions we are looking for as the above examples show. As far as a machine learning solution is concerned, the compilation of annotated training corpora would represent a major difficulty. Whereas the compilation of a corpus of sentences describing background knowledge is conceivable (it is part of the Zone Analysis project [9]), the same task in the case of dangerous devices or innovative products is much more tedious exactly because of the great variety of linguistic expressions and data scarcity: we would need the tool we are to develop in order to constitute a training corpus. In the case of background knowledge, even if we have a large quantity of sentences, it is impossible to make sure - for the same reason - that a sufficiently wide range of expressions is represented.

Existing document processing systems like for example Fastus [4] typically deal with surface variations of common underlying meanings by allowing substitution of similar expressions or expression schemata, where similarity is defined such that if one expression is relevant to a user's query, then similar ones can be assumed to be relevant as well. These similarities are usually defined using three levels of linguistic information: morphological, syntactic and lexical semantic equivalences.

The framework we propose relies on a novel kind of common pattern: shared conceptual features. The expressions conveying the contextual concepts are composed of constituent expressions presenting those features. We introduce the notion of constituent concepts through simple paraphrases of our target concepts as follows:

**a. Background knowledge**

Paraphrase: **past/general achievement:**

*Recent* **[past/general]** *studies* **[achievement]**

*previously* **[past/general]** *proposed* **[achievement]**

*universally* **[past/general]** *accepted* **[achievement]**

**b. Danger**

Paraphrase: **possibility** of **causing/resulting** in **bad thing**

*could* **[possibility]** *accumulate* **[causeff]** *with disastrous* **[bad_thing]** *consequences* **[causeff]**

*potential* **[possibility]** *cause* **[causeff]** *of ... tumors* **[bad_thing]**

*possible* **[possibility]** ... *risks* **[bad_thing]** *posed* **[causeff]**

**c. Innovation**

Paraphrase: **now available interesting product** is in **contrast** with **products available** in the **past**

*cannot* **[contrast]** *be obtained* **[available]** *with conventional* **[past]** … *methods* **[product]**

*unlike* **[contrast]** *today's* **[now]** *existing* **[available]**... *options* **[product]**

*exciting* **[attitude]** *new* **[now]** ***product*** **[product]** *unlike* **[contrast]** *... previously* **[past]** ***available*** **[available]**

Besides being composed of constituent concepts, the above expressions reveal a second property of expressions of contextual concepts . We can also observe that the contextual concepts are conveyed through coherent linguistic structures, i.e. the constituent concepts are in linguistic dependency relationships with each other (DEPENDENCY(argument1,argument2)):

**a. Background knowledge**

*Recent studies*
**MOD(***studies***[achievement],***Recent***[past/general])**

*previously proposed*
**MOD(***proposed***[achievement]**,*previously***[past/general])**

*universally accepted*
**MOD(***accepted***[achievement]**,*universally***[past/general])**

**b. Danger**

*could accumulate with disastrous consequences*
**AUX(***accumulate***[causeff]**,*could***[possibility])**
**MOD(***accumulate***[causeff]**,*consequences***[causeff])**
**MOD(***consequences***[causeff]**,*disastrous***[bad_thing])**

*potential cause of ... tumors*
**MOD(***cause***[causeff]**,*potential***[possibility])**
**MOD(***cause***[causeff]**,*tumors***[bad_thing])**

*possible ... risks posed*
**MOD(***risks***[bad_thing]**,*possible***[possibility])**
**MOD(***risks***[bad_thing]**,*posed***[causeff])**

**c. Innovation**

*cannot be obtained with conventional ... methods*
**AUX(***obtained***[available]**,*cannot***[contrast])**
**MOD(***obtained***[available]**,*methods***[product])**
**MOD(***methods***[product]**,*conventional***[past])**

*unlike today's existing ... options*
**PREP(***options***[product]**,*unlike***[contrast])**
**MOD(***options***[product]**,*existing***[available])**
**MOD(***options***[product]**,*today***[now])**

*exciting new product unlike ... previously available*
**MOD(***product***[product]**,*exciting***[attitude])**
**MOD(***product***[product]**,*new***[now])**

**PREP**(*available*[**available**],*unlike*[**contrast**])
**MOD**(*available*[**available**],*previously*[**past**])

Now we have introduced the two basic ingredients that make up the patterns matching contextual concepts: constituent concepts and syntactic relationships. They are both different from traditional pattern matching features. The keywords in constituent concept classes are neither synonyms, nor of the same type as for example named entities, nor do they necessarily belong to the same part of speech class. The only element that they share is one common feature of their meaning: the very feature that is a component of the meaning of the target concept. As for the syntactic relationships, the only information we use is that two words are in a dependency relationship. The type of relationship (subject, modifier, etc.) is unimportant. This is explained by the fact that the presence of a syntactic relationship between two words reflects semantic coherence between them, thus since our target concepts are semantically coherent, so are they syntactically.

The third ingredient of the concept-matching systems is co-occurrence rules that match certain syntactically related pairs of keywords expressing particular constituent concepts with the target concepts. The more constituent concepts we have in the system the more co-occurrence rules we need. Co-occurrence rules filter out wrong combinations of the constituent concepts, i.e. cases when the expressions of the constituent concepts do not make up the target concept, like in the following sentence:

*No* **[contrast]** *new* **[now]** *products* **[product]** *are available* **[available]** *at the moment.*

The development infrastructure and general linguistic resource of our systems is a robust dependency parser, XIP [1].

## 3. CONCLUSION AND CHALLENGES FOR FUTURE WORK

The most important feature of our framework is its ability to represent varied and discontinuous linguistic expressions of complex concepts as instantiations of the same underlying pattern, which is not realizable with traditionally used patterns. The heterogeneity of the keywords composing the lists of constituent concepts on the one hand and the flexibility of the syntactic constraints on the other hand, lead to the coverage of a broad range of surface expressions. The only system that has been extensively evaluated is the system for detecting paradigm shifts, where the precision is almost 100%. Recall is not easy to estimate.

The main drawback of our framework is that it extensively relies on concept-specific resources, which must be constructed ad hoc and are not obtained automatically at this stage.

The value of a concept matching system lies in the relevance of its target concepts. The paradigm shift concept has proved to be relevant for text mining in biology (see Evaluation of the Method in [6]), so the effort invested in the manual compilation of the resources has been justified. We believe that concepts like background knowledge, innovation and danger are relevant, although we have not had actual applications. Future research is targeted at the definition of relevant contextual concepts as well as the development of a standard process for constructing concept-matching systems in which as many steps as possible are automated.

## 4. ACKNOWLEDGMENTS

I would like to thank Aaron Kaplan whose contribution to this work has been essential

## 5. REFERENCES

[1] Aït-Mokhtar S., Chanod J.-P., Roux, C. Robustness beyond shallowness: incremental dependency parsing. *Natural Language Engineering, 8(2/3) 2002. 121-144.*

[2] DARPA, ed. *Proceedings of 6th Message Understanding Conference (MUC-6), 1995.*

[3] Ferret, O. Segmenter et structurer thématiquement des textes par l'utilisation conjointe de collocations et de la récurrence lexicale. *TALN, Nancy, 24-27 juin 2002..*

[4] Hobbs, J., Appelt, D., Bear, J., Israel, D., Kameyama, M., Stickel, M., and Tyson, M *FASTUS: a cascaded finite-state transducer for extracting information from natural-language text*. in Finite State Devices for Natural Language Processing (E. Roche and Y. Schabes, eds.) MIT Press, Cambridge, USA, 383-406. 1996.

[5] Langer, H., Lungen, H., Bayerl, P.S. *Text Type Structure and Logical Document Structure.* Proceedings of the ACL Workshop on Discourse Annotation. Barcelona, Spain. 49-56. 2004.

[6] Lisacek, F., Chichester, C., Kaplan, A., Sándor, Á. *Discovering Paradigm Shift Patterns in Biomedical Abstracts: Application to Neurodegenerative Diseases.* Proceedings of the First International Symposium on Semantic Mining in Biomedicine (SMBM) 10th -13th April, 2005. European Bioinformatics Institute, Hinxton, Cambridgeshire, UK. 41-50. 2005.

[7] Minel J-L., Desclés J-P., Cartier E., Crispino G., Ben Hazez S., Jackiewicz A. *Résumé automatique par filtrage sémantique d'informations dans des textes.* Présentation de la plate-forme FilText , Revue Technique et Science Informatique, n° 3, 2001.

[8] Mizuta, Y., Collier, N. *Zone Identification in Biology Articles as a Basis for Information Extraction.* Proceedings of the Joint Workshop of Natural Language Processing in Biomedicine and Its Applications (JNLPBA) at the Coling 2004 International Conference, 19-35.

[9] Mizuta, Y., Mullen, T., Collier, N. Annotation of Biomedical Texts for Zone Analysis. NII Technical Report (NII-2004-007E, ISSN 1346-5597). 2004.

[10] Rodríguez, C. 2003, *Applying Information Extraction techniques to metalinguistic discourse.* Topics in Computational Linguistics and Intelligent Text Processing; Lecture Notes in Computer Science. Springer-Verlag. 2003.

[11] Teufel, S. *Meta-discourse markers and problem-structuring in scientific articles.* In M. Stede, L. Wanner, and Eduard Hovy, editors, Proceedings of the Workshop on Discourse Relations and Discourse Markers at the 17th International Conference on Computational Linguistics. 43-49, August 15. 1998.

[12] Teufel,S., Moens, M. *Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status.* Computational Linguistics, 28 (4), 409-445. Dec. 2002.

# Automatic Hierarchical Clustering of Web Pages

Ricardo Campos
Centre of Human Language Technology and Bioinformatics
University of Beira Interior
+351 275 319 891
ricardo.campos@ipt.pt

Gaël Dias
Centre of Human Language Technology and Bioinformatics
University of Beira Interior
+351 275 319 891
ddg@di.ubi.pt

## ABSTRACT

In this paper, we propose a system that clusters web pages and presents them as a hierarchical structure instead of a classical search ordered list retrieved from any search engine. The organization of the results based on this concept makes easier the user's search navigation between the results of the search engine. In particular, we use web content mining techniques to represent texts, based on their most relevant terms, which can be simple words or phrases. A soft clustering algorithm is then applied to group documents into clusters hierarchically linked. Finally, each cluster is labelled with its most relevant term based on a simple heuristics.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *clustering.*

## General Terms

Algorithms, Measurement, Experimentation.

## Keywords

Web page automatic clustering, Hierarchical soft clustering, Web content mining.

## 1. INTRODUCTION

The World Wide Web has become a huge network of information and search engines actually deal with the problem of retrieving relevant documents. One of their main problems is that the induced relevance may not satisfy the user intents. This is manly due to two problems: (1) search engines interpret the content of documents in a basic way and (2) they present the retrieved information in an unstructured way. In fact, systems are not capable of understanding completely what users are looking for due to small queries and on the other side, they keep retrieving a huge set of unstructured information.

To answer these problems, we propose a meta-search engine named WISE that uses web content mining techniques introduced by [1] associated to a soft clustering algorithm called PoBoc [2] to find, analyze, understand, disambiguate and organize the set of documents returned by any search engine for a given query. As a consequence, the user's search is simplified turning his task less time-consuming.

## 2. RELATED WORK

The Information Retrieval community has suggested in scientific published literature different solutions to the problem of organizing web search results. But all these works [3] [4] [5] [6] [8] [9] [10] have in common the fact that they mainly consider the titles and the snippets of each document retrieved from a search query. However, [3] refer that the results are obviously inferior when compared to the use of overall text. For that purpose, [4] enriches[1] the snippets with two existing knowledge bases. All these works also use lists of stop-words and stemming algorithms which make their solution language-dependent.

In order to represent the documents retrieved by the search engines, [4] [5] [6] use the well-known space vector model [7] considering only the snippets and not the overall text. Some other works show another text representation. [2] [8] [9] [10] use the concept of shared n-grams between snippets. But none uses web content mining techniques to represent their documents as we will show in the next section.

Once documents are represented into a given structure, clustering techniques must be applied to produce a structured list of results. The proposed algorithms proposed so far in the literature distinguish themselves by the use of (1) simple words or phrases and (2) by implementing flat clustering or hierarchical clustering. In particular, the work done by [11] which was not tested in web environment and [3] propose flat clustering with simple words, while [8] and [10] do it with phrases. [9] are the first to introduce hierarchical clustering with phrases, followed by [4] and [5]. [6] also propose hierarchical clustering but just considering simple words. In our work, we use a soft clustering algorithm called PoBoc [2] that has shown successful results within the analysis of textual data and allows words to belong to different clusters. It is used in association with phrases that are extracted previously from texts based on the SENTA software [12].

Our solution differs from all previous work as it proposes a deep analysis of text content using a multiword extractor that produces relevant phrases. Compared to existing methodologies that elect frequent strings as phrases, we use a more sophisticated language-independent phrase extractor [12]. Based on the extraction of these phrases, we then apply web content mining techniques to extract as deep knowledge as possible from texts to finally

[1] To our best knowledge, they are the only ones.

produce a set of soft clusters based on a soft clustering algorithm called PoBoc [2].
In this paper, we will first show the architecture of our solution. In a second part, we will show some first results and finally draw some conclusions.

## 3. GLOBAL ARCHITECTURE

Our architecture called WISE is composed of 4 main parts:

(1) The selection of relevant pages from the set of all retrieved documents by the search engine;
(2) The integration of the SENTA software [12] that extracts phrases from raw texts;
(3) The detection of relevant terms that characterize the document, using the WEBSPY software [1] that implements the web content mining techniques;
(4) The presentation of the documents into a hierarchical structure using the PoBoc [2] algorithm.

The overall architecture can be seen in Figure 1.

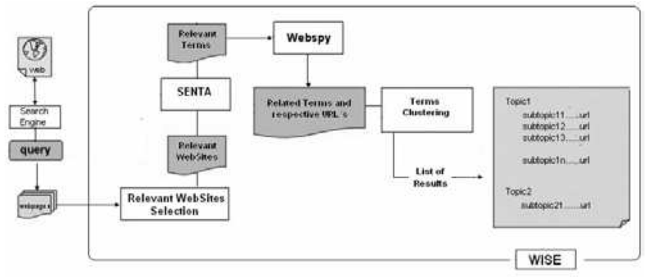


**Figure 1. The architecture of WISE.**

Our algorithm proposes a new method for extracting relevant pages and a new representation of documents, based on relevant terms to apply a clustering algorithm. Our algorithm follows the next 8 steps:

(1) Retrieve the list of results of the search engine for a given query (meta-crawler);
(2) Select the most important results from all the retrieved documents. All the literature specified above treats each document as equals, but they are not. Each one has different relevance to the query, which decreases as more and more documents are retrieved. Producing clusters in many documents of little relevance can reduce the quality of the results, and for that reason we exclude some results, benefiting precision upon recall. For that purpose, we applied a function that chooses from the returned documents, the best ones i.e. the ones that overpass a given threshold calculated by Equation 1.

$$average\_relevance = \frac{\#\,\text{returned URLs}}{\#\,\text{different absolute URLs}} \quad \textbf{(1)}$$

Based on the above equation, we select all the relevant URLs which number of occurrences is greater than the calculated threshold. But, we also take as relevant all the retrieved absolute URLs.
In order to better understand our procedure, the number of occurrences of a URL is the sum of all URLs that share the same absolute URL. For instance, considering Figure 2, the average relevance threshold would be 4/3 = 1.3. As a consequence, we would not consider the webpage which absolute URL is http://geocities.com being its number of occurrences below the threshold.

| Site | N.º of Occurrences |
|---|---|
| http://www.vodafone.com | 2 |
| http://www.vodafone.com/news/01.html | 2 |
| http://geocities/mobiles/test.html | 1 |
| http://www.manUnited.uk | 1 |

**Figure 2. List of results related to mobile phones query.**

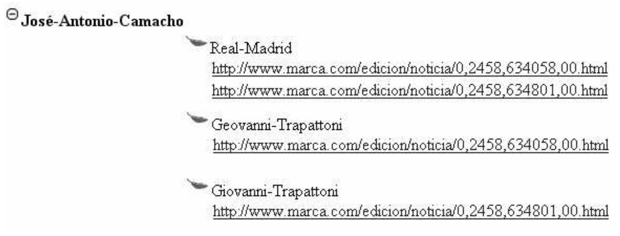
In addition, we extend the number of relevant pages given by the search engine by considering a set of pages not caught by the system, but related with the query. For that purpose, for any absolute retrieved URL, we catch its N best pages re-running the search engine over the absolute URL with the same query.
(3) Identify phrases in the documents in order to increase the knowledge about each document by using the SENTA software [12];
(4) Calculate the set of relevant terms to the query for each document by applying the WEBSPY software [1] that implements a set of decision trees C5.0 based on 12 characteristics between all the words/phrases in the document and the query terms. This step retrieves a set of related terms with a probability of relevance. Our purpose is to use the overall text and all the relations between words/phrases and the query term to represent as best as possible the semantic content of each text;
(5) Calculate the similarity matrix. In order to prepare the clustering step, the WEBSPY software is applied again. For each relevant term retrieved from step (4) we apply WEBSPY based on the pages where relevant term occurs. As a consequence, each relevant word/phrase is also represented by a set of related words with a given relevance probability. In order to build the similarity matrix, we then apply the Cosine measure between all the pairs of relevant word/phrase.
(6) Group into a hierarchical structure the set of all relevant documents retrieved in step (2). By this clustering step, we aim at disambiguating the sense of the query term. Thus, the user is helped in his search for information. This step is done using the PoBoc algorithm [1], a soft clustering algorithm that allows a word/phrase to belong to different cluster. This characteristic is fundamental for text analysis as it is obvious that one word may appear in different contexts with a different semantic content.
(7) Label each cluster with its most relevant word/phrase based on a simple heuristic that chooses the word/phrase that occurs more often in the set of words/phrases representing relevant terms and taking into account the sum of probabilities in case of ties.
(8) Present the final results to the user.

Our solution uses the overall text information, not only the titles and the snippets, turning the solution more robust in terms of semantic ambiguity. Also, we do not use lists of stop-words neither we use stemming algorithms, which makes our solution flexible, and domain/language-independent. To our best knowledge, we are the first using web content mining techniques to understand documents, using afterwards this knowledge as a base to form structured hierarchical soft clusters.

## 4. RESULTS

The results shown below are clusters returned by our system WISE from the system query execution *Benfica* on may 31st, 2005 using Google™ as search engine. The cluster that can be seen in Figure 3 refers to the set of URLs related to *José-António-*

*Camacho*[2], the former football coach of *Real-Madrid*, spoken at that time to be a possible successor of *Giovanni-Trapattoni* as the new coach of *Benfica*. We can notice that the labels show some degree of quality and semantic description of the content of the cluster due to the identification of relevant phrases. One other interesting issue is the capacity of the system to deal with word mistakes (*Giovanni*, not *Geovanni*). This result is due to the fact that no restriction is made over term frequency.



**José-Antonio-Camacho**

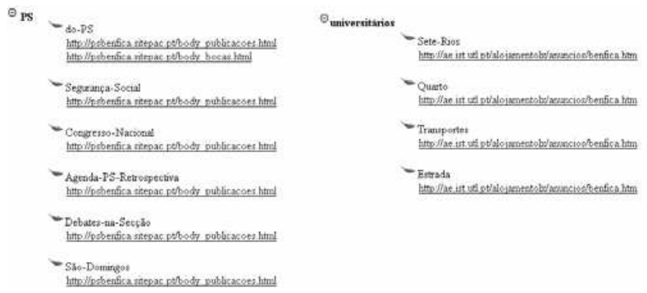
Real-Madrid
http://www.marca.com/edicion/noticia/0,2458,634058,00.html
http://www.marca.com/edicion/noticia/0,2458,634801,00.html

Geovanni-Trapattoni
http://www.marca.com/edicion/noticia/0,2458,634058,00.html

Giovanni-Trapattoni
http://www.marca.com/edicion/noticia/0,2458,634801,00.html

**Figure 3. Monothetic labels and phrases[3].**

In Figure 4, we can see the ability of our system to deal with term disambiguation. As we have already seen, *José-António-Camacho* is related to *Benfica* (i.e., *Benfica* is related to Benfica Football Club), but the system also retrieves a cluster with label *PS* which refers to the politic socialist party located in the *Benfica* neighborhood (i.e. *Benfica* is related to politics through the fact that *Benfica* is also related to a famous neighborhood of Lisbon). Moreover, the label *Universitários* refers to student life like housing, transports and roads (i.e. *Benfica* is also related to a privileged neighborhood for student housing).



**PS**

do-PS
http://psbenfica.sitepac.pt/body_publicacoes.html
http://psbenfica.sitepac.pt/body_bocas.html

Segurança-Social
http://psbenfica.sitepac.pt/body_publicacoes.html

Congresso-Nacional
http://psbenfica.sitepac.pt/body_publicacoes.html

Agenda-PS-Retrospectiva
http://psbenfica.sitepac.pt/body_publicacoes.html

Debates-na-Secção
http://psbenfica.sitepac.pt/body_publicacoes.html

São-Domingos
http://psbenfica.sitepac.pt/body_publicacoes.html

**universitários**

Sete-Rios
http://ae.ist.utl.pt/alojamentobt/anuncios/benfica.htm

Quarto
http://ae.ist.utl.pt/alojamentobt/anuncios/benfica.htm

Transportes
http://ae.ist.utl.pt/alojamentobt/anuncios/benfica.htm

Estrada
http://ae.ist.utl.pt/alojamentobt/anuncios/benfica.htm

**Figure 4. Word sense disambiguation.**

## 5. CONCLUSION

Our paper proposes to organize the flat ranked search lists returned by current search engines to produce a topic hierarchical structure that will help the user in his search for information. Our main contribution to the field is the use of web content mining techniques applied to the overall information within texts which allows deep semantic analysis of web documents, understanding facts that, till now, no search engine understands. Moreover, the identification of phrases to define key concepts in texts allows a greater document content understanding. The architecture and the proposed algorithms are the answer to one of the biggest problems of search engines: returning quality results through an organized and disambiguated structure of concepts. This paper shows an on-going work and further improvements will be made, especially in terms of merging clusters as data sparseness usually produce too many clusters. The WISE software will be soon freely available at http://wise.di.ubi.pt under GPL license.

[2] *José-António-Camacho* stands for the name *José António Camacho* that has been recognized by SENTA as a phrase.

[3] The fact that we only have one absolute URL for the concept *José-António-Camacho* is casual. In fact, our system is capable of retrieving different absolute URLs for the same concept.

## 6. REFERENCES

[1] Veiga, H., Madeira, S. and Dias, G. 2004. *Webspy*. Technical Report no 1/2004. http://webspy.di.ubi.pt

[2] Cleuziou, G., Martin, L. and Vrain, C. 2003. PoBOC: an Overlapping Clustering Algorithm. Application to Rule-Based Classification and Textual Data. In Proceedings of the 16th biennial European Conference on Artificial Intelligence (ECAI'04), Valencia, Spain, August, 440-444.

[3] Jiang, Z., Joshi, A., Krishnapuram, R. and Yi, L. 2002. Retriever: Improving web search engine results using clustering. In Managing Business with Electronic Commerce.

[4] Ferragina, P. and Gulli, A. 2005. A Personalized Search Engine Based on Web-Snippet Hierarchical Clustering. In the Proceedings of the 14th international conference on World Wide Web, Chiba, Japan, ISBN: 1-59593-051-5. 801-810.

[5] Martins, B. and Silva, M. 2003. Web Information Retrieval with Result Set Clustering. In Proceedings of NLTR 2003 - Natural Language and Text Retrieval Workshop associated to EPIA'03, December.

[6] Fung, B., Wang, K. & Ester, M. (2003). *Large hierarchical document clustering using frequent itemsets*. In Proceedings of the SIAM International Conference on Data Mining, Cathedral Hill Hotel, San Francisco, CA, May 1-3.

[7] Salton, G., Yang, C.S., and Yu, C.T. 1975. A theory of term importance in automatic text analysis. American Society of Information Science 26, 1, 33-44.

[8] Zamir, O. and Etzioni, O. 1998. Web Document Clustering: A Feasibility Demonstration. In Proceedings of the 19th International ACM SIGIR Conference on Research and Development of Information Retrieval (SIGIR'98), 46-54.

[9] Zhang, D. and Dong, Y. 2001. Semantic, Hierarchical, Online Clustering of Web Search Results. In Proceedings of the 6th Asia Pacific Web Conference (APWEB), Hangzhou, China, April.

[10] Zeng, H., He, Q., Chen, Z. and Ma, W. 2004. Learning to cluster web search results. In the Proceedings of the 27th annual international conference on Research and development in information retrieval, Sheffield, UK, ISBN: 1-58113-881-4, 210-217.

[11] Hearst, M. and Pedersen, J. 1996. Re-examining the Cluster Hypothesis: Scatter/ Gather on Retrieval Results. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96), Zurich, Switzerland, August, 76-84.

[12] Dias, G. (2002). Extraction Automatique d'Associations Lexicales à partir de Corpora. PhD Thesis. DI/FCT New University of Lisbon (Portugal) and LIFO University of Orléans (France).

# Evaluating Latent Semantic Vector Models with Synonym Tests and Document Retrieval

Leif Grönqvist
School of Mathematics and Systems Engineering
Växjö University, Sweden
The National Graduate School of Language Technology (GSLT)
Göteborg, Sweden
leif.gronqvist@msi.vxu.se

## ABSTRACT

The most interesting result of our paper [2] is the improvement in performance for a latent semantic vector model (LSVM) when we enrich it with bigrams and trigrams. The evaluation is made using a Swedish synonym test. This poster is about the concrete performance differences between a word based model (WLSVM) and the enriched one (T3LSVM). [1] The second part of this poster is about evaluation of an LSVM using a document retrieval testbed with incomplete information, i.e. we do not have relevance judgements for all queries and each document in the set. We will discuss possible evaluation measures.

## Categories and Subject Descriptors

H.3 [**Information Systems**]: Information Storage and Retrieval; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

## General Terms

Algorithms, Experimentation, Performance, Languages

## Keywords

Evaluation, latent semantic indexing, semantic vector models, n-grams, multi-word units, evaluation measure

## 1. EVALUATION WITH SYNONYM TESTS

This section is a deeper look at the evaluations in our paper, [2] to find what is really happening when adding bigrams and trigrams to an LSVM.

### 1.1 Difference types between the models

The two models are trained using Bring+Lexin, with Context=100, Dim=1000, CoVoc=10000, and the complete vocabulary. Basically we have four cases:

[1] T3 stands for tuples up to length 3.

*ELECTRA Workshop - Methodologies and Evaluation of Lexical Cohesion Techniques in Real-world Applications* Salvador, Brazil, August 19. 2005, in association with SIGIR-2005
Copyright 2005 ACM 1-59593-034-5/05/0008 ...$5.00.

**Table 1: A summary for a comparison of two models**

| | =0 | =1 | + | − |
|---|---|---|---|---|
| Total | 148 | 247 | 30 | 15 |
| Percentage | 33.6% | 56.1% | 6.82% | 3.41% |

| | |
|---|---|
| =0 | Both models (WLSVM and T3LSVM) are wrong |
| =1 | Both models are right |
| + | Improvement: T3LSVM is right and WLSVM is wrong |
| − | Change for the worse: WLSVM is right and T3LSVM is wrong |

The + and − columns in table 1 shows the differences between the models, so let us take a look at the specific cases behind these numbers.

### 1.2 Single word questions and answers

Table 2 shows the cases where the models have different result, and both the questions and answers are single words. Note that for single words, we have 10 improvements and 10 changes for the worse, which is actually a positive result since the T3LSVM is mainly supposed to handle MWUs (MultiWord Units) better and it still seems just as good as WLSVM on single words.

### 1.3 Single word questions and MWU answers

In table 3 we have the cases where the question is single word and at least one of the selected answer alternatives are MWUs. We can see an improvement in 10 cases out of 14.

### 1.4 MWU questions

Table 4 shows the results for MWU-questions, and now we get an improvement in 10 out of 11 cases!

## 2. EVALUATION USING A DOCUMENT RETRIEVAL TESTBED

Since we are using an existing testbed, the easiest approach – to look at top lists of different lengths and count the number of relevant documents – is not really fair. The problem is that except for the relevant and non-relevant documents for each topic, we now have a bunch of unknown documents. We do not want to count unknown documents as non-relevant, instead we want to base the evaluation measure on the manually judged documents only.

Table 2: **Single word questions and answers**

| Question | WLSVM-answer | T3LSVM-answer |
|---|---|---|
| paradoxal/ paradoxical | likartad/ similar | *motsägelsefull/ contradictory* |
| barockt/absurd | exklusivt | *orimligt/absurd* |
| avhandla/ deal with | rådfråga/consult | *diskutera/discuss* |
| ograverad/ untouched | *orörd/untouched* | ojämn/rough |
| begeistrad/ enthusiastic | otålig/impatient | *förtjust/delighted* |
| sinnrik/ingenious | *fyndig/inventive* | vettig/reasonable |
| avisera/announce | rådgöra/ consunt with | *underrätta/ inform* |
| absorption | *upptagande* | överföring |
| fromleri | girighet | *skenhelighet* |
| överhalning | *utskällning* | upprepning |
| baxna | *häpna* | blekna |
| indisponerad | *opasslig* | obetydlig |
| raljera | *skämta* | skratta |
| godvilligt | osjälviskt | *självmant* |
| bevekande | *vädjande* | uppmärksam |
| aversion | ilska | *motvilja* |
| beting | försök | *arbetsuppgift* |
| symbios | *samlevnad* | förruttnelse |
| vattendelare | passage | *skiljelinje* |
| anfang | *begynnelsebokstav* | slutord |
| **Correct** | **10 (50%)** | **10 (50%)** |

The correct answer is marked in italics.

Table 3: **Single word questions and MWU answers**

| Question | WLSVM-answer | T3LSVM-answer |
|---|---|---|
| bravad/ exploit | osjälvisk handling/ unselfish act | *utmärkt prestation/ excellent achievem.* |
| ringakta/ despise | skämmas för/ be ashame of | *se ned på/ look down on* |
| insinuation | grav misstanke | *elak antydan* |
| ambulera/ ambulate | *flytta omkring/ move around* | skynda på/ hurry up |
| travestera | envist upprepa | *skämtsamt förvränga* |
| kverulant | *klagande person* | skrytsam person |
| traktera | behandla någon illa | *bjuda på mat och dryck* |
| intuition | känsla av obehag | *förmåga till spontan bedömning* |
| avbräck/ setback | förkortning/ abbreviation | *ekonomisk skada/ economical loss* |
| galghumor | elak och nedlåtande humor | *bister och ironisk humor* |
| falang | *riktning inom politiskt parti eller annan organisation* | grupp av ämnen vid universitet eller högskola |
| nyansera | överbrygga åsiktsskillnader | *åstadkomma finare skiftningar* |
| brådmogen | illa förberedd | *tidigt utvecklad* |
| grannlaga | *som kräver finkänslighet* | som måste göras snabbt |
| **Correct** | **4 (28.6%)** | **10 (71.4%)** |

Table 4: **MWU questions and answers**

| Question | WLSVM | T3LSVM |
|---|---|---|
| ta skruv/ do the trick | ta sats/ take a run | *ha verkan/"work"* |
| utgjuta sig/ dilate | berömma sig själv/pride oneself | *tala vitt och brett/talk a great length* |
| inte skräda orden/not mince matters | vara fåordig/ being silent | *säga sin uppriktiga mening/give one's honest opinion* |
| vinnlägga sig/ strive after | skynda sig/ hurry up | *anstränga sig/ exert oneself* |
| dabba sig/ make a blunder | tappa fattningen /lose one's head | *begå ett misstag/ make a blunder* |
| hålla tand för tunga/be quiet | hålla med/agree | *hålla tyst/ keep silent* |
| black om foten/ impediment | stöd/support | *hinder/obstacle* |
| förtrösta på/ in trust | *sätta sin lit till/ put confidence in* | gå i god för/ guarantee |
| inte ett vitten | inte en aning | *inte ett dugg* |
| hugga i sten/go wide of the mark | anstränga sig/ exert oneself | *missta sig/ be misstaken* |
| frottera sig med /get on with | retas med/ tease | *umgås med/ get on with* |
| **Correct** | **1 (9.1%)** | **10 (91.0%)** |

## 2.1 Buckley & Voorhees: bpref

Bpref [1] is a measure defined to work for testbeds with incomplete information, see equation 1.

$$bpref = \frac{1}{R}\sum_{r=1}^{R} 1 - \frac{Irr_R(r)}{R} \tag{1}$$

| | |
|---|---|
| R | number of known relevant documents |
| $Irr_R(r)$ | number of documents, among the top R known irrelevant ones, ranked higher than r |

One problem with the bpref definition, that the authors note in their paper, is that only the same number of irrelevant documents as the number of relevant (R) are used for the calculation. This is a problem especially when the number of known relevant documents is low. Bpref-10 is an improved variant of bpref, using R+10 irrelevant documents, but it still to some extent has the same weakness. Note also that the bpref measures is only defined for data sets with at least R [2] irrelevant documents.

## 2.2 The RankEff measure

We are currently using what we call ranking efficiency (RankEff) defined in equation 2.

$$RankEff = \frac{\sum_{r=1}^{R} Irr(r)}{R(N-R)} \tag{2}$$

| | |
|---|---|
| N | number of relevance judged documents |
| R | number of known relevant documents |
| Irr(r) | number of irrelevant documents ranked higher than r |

[2]R+10 for Bpref-10

RankEff is similar to bpref but it does not suffer from the same weaknesses as bpref:

- It uses all the relevance judgements
- It can handle data sets with any number of relevant and irrelevant documents, except if R=0 or N=R
- It handles small number of relevant documents better than bpref

To get an idea on what the values says, we can note that if all relevant documents are ahead of all irrelevant, we get 100%. The opposite case leads to 0%, and between these extremes, the measure shows the level for the average rank of the relevant documents. RankEff and bpref are equivalent if exactly 50% of the documents are relevant.

## 2.3 A test evaluation using bpref and RankEff

To get a first impression of the two evaluation measures we have tested to evaluate a WLSVM trained on the Clef corpus [2] with the parameters: Dim=300, Con=300, Vocabulary=300 000 and CoVoc=5000. The testbed contains 50 topics (queries) [3] with manual relevance judgements for 100-500 documents. [4] In table 5 we can see the RankEff and bpref values for each topic. Note that RankEff and bpref is very similar when $n_{rel}$ is close to $n_{irrel}$, i.e. topic 140 and 139. For some topics, bpref becomes 0, but RankEff shows different values, for example topic 126 where we have only three relevant documents. The relevant documents are found at ranks: 26, 34, and 83 out of 390 which gives RankEff=88%. Another topic with bpref=0 is number 121 where the only relevant document has rank 39 out of 98, which gives a muck lower RankEff at 61%. The reason why bpref cannot give meaningful values here is that it looks at very few irrelevant documents, in these cases only one or three documents.

Some users do not care about the lower rank documents, but we think a good evaluation measure should show if a relevant document moves up in the ranking, even if it is in the lower regions. RankEff will do this, but a weakness is that it will give the same value if two systems rank the relevant documents at 1, 2, 51, 52 and 25, 26, 27, 28 respectively. Some users would prefer the first ranking and some the second. A measure that could favour one of them based on user preferences could be useful and not very complicated to construct.

## 2.4 Future work

We have not analyzed RankEff enough yet. Especially we need to think more about significance testing when used on a set of queries.

## 3. REFERENCES

[1] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *SIGIR '04*, pages 25–32, New York, NY, USA, 2004. ACM Press.

[2] L. Grönqvist. An evaluation of bi- and trigram enriched latent semantic vector models. In *ELECTRA Workshop, in association with ACM SIGIR*, 2005.

[3]Topic 110 is not included in table 5 since it had no relevant documents at all

[4]The documents were selected for manual judgements based on top-100 lists from five simple document retrieval systems. Different grades of overlap gave different number of documents.

**Table 5: RankEff and bpref for some example topics**

| Topic | RankEff | bpref | $n_{rel}$ | $n_{irrel}$ |
|---|---|---|---|---|
| 91 | 83.37% | 41.32% | 11 | 141 |
| 92 | 75.79% | 57.72% | 60 | 108 |
| 93 | 79.06% | 28.00% | 5 | 297 |
| 94 | 98.23% | 86.11% | 12 | 292 |
| 95 | 85.38% | 55.52% | 71 | 232 |
| 96 | 87.55% | 16.67% | 6 | 257 |
| 97 | 84.08% | 41.84% | 14 | 192 |
| 98 | 100.00% | 100.00% | 2 | 247 |
| 99 | 87.65% | 52.08% | 12 | 272 |
| 100 | 77.36% | 27.81% | 13 | 124 |
| 101 | 94.37% | 58.33% | 12 | 228 |
| 102 | 71.87% | 46.50% | 60 | 122 |
| 103 | 78.36% | 9.50% | 20 | 211 |
| 104 | 70.07% | 12.47% | 19 | 188 |
| 105 | 73.26% | 6.54% | 32 | 342 |
| 106 | 69.43% | 18.82% | 33 | 229 |
| 107 | 70.04% | 11.00% | 10 | 242 |
| 108 | 58.40% | 2.68% | 28 | 151 |
| 109 | 86.44% | 22.99% | 19 | 323 |
| 111 | 77.60% | 13.22% | 11 | 349 |
| 112 | 80.33% | 56.00% | 5 | 180 |
| 113 | 75.91% | 1.17% | 16 | 342 |
| 114 | 82.93% | 25.33% | 15 | 184 |
| 115 | 51.50% | 4.63% | 18 | 352 |
| 116 | 81.45% | 28.93% | 11 | 298 |
| 117 | 85.01% | 0.00% | 3 | 289 |
| 118 | 87.67% | 19.00% | 10 | 279 |
| 119 | 89.82% | 76.95% | 54 | 127 |
| 120 | 89.51% | 0.00% | 1 | 162 |
| 121 | 60.82% | 0.00% | 1 | 97 |
| 122 | 75.34% | 26.80% | 47 | 241 |
| 123 | 96.38% | 76.00% | 10 | 160 |
| 124 | 74.48% | 53.90% | 76 | 148 |
| 125 | 91.00% | 75.03% | 69 | 195 |
| 126 | 88.20% | 0.00% | 3 | 387 |
| 127 | 94.12% | 71.88% | 8 | 319 |
| 128 | 85.37% | 15.00% | 20 | 307 |
| 129 | 91.17% | 52.78% | 30 | 256 |
| 130 | 98.63% | 63.89% | 6 | 268 |
| 131 | 59.22% | 5.54% | 17 | 280 |
| 132 | 69.44% | 21.03% | 33 | 226 |
| 133 | 88.11% | 51.21% | 17 | 284 |
| 134 | 98.81% | 50.00% | 2 | 210 |
| 135 | 89.23% | 69.95% | 55 | 193 |
| 136 | 98.36% | 65.31% | 7 | 192 |
| 137 | 42.88% | 0.00% | 6 | 358 |
| 138 | 81.48% | 12.50% | 8 | 270 |
| 139 | 84.67% | 80.36% | 106 | 137 |
| 140 | 83.40% | 82.68% | 92 | 96 |
| **Average** | **81.30%** | **36.63%** | **24.4** | **232.3** |

***Thank You!***

**Workshop Chairs**

Rosie Jones (Yahoo! Inc., USA)

Olga Vechtomova (University of Waterloo, Canada)

Gaël Harry Dias (University of Beira Interior, Portugal)