Katholieke Universiteit Leuven

PROCEEDINGS OF THE THIRD DUTCH-BELGIAN INFORMATION RETRIEVAL WORKSHOP (DIR-2002)

Katholieke Universiteit Leuven
December 6, 2002

PROCEEDINGS OF THE THIRD DUTCH BELGIAN INFORMATION RETRIEVAL WORKSHOP

Date
Friday, December 6, 2002

Venue
Auditorium LAND 00.215,
Main Building of the Institute for Agriculture
Kasteelpark Arenberg 20
3001 Heverlee
Belgium

Organisation

Interdisciplinary Centre for Law and IT
Research Group Legal Informatics & Information Retrieval

In cooperation with:

IWT Vlaanderen

Dutch Research School for Information and Knowledge Systems (SIKS)

Proceedings editors
Marie-Francine Moens
Rik De Busser
Djoerd Hiemstra
Wessel Kraaij

Reviewers
Regina Barzilay
Paul Clough
Bruce Croft
Norbert Fuhr
Fredric Gey
Djoerd Hiemstra
Graeme Hirst
Wessel Kraaij
Frederique Lisacek
James Mayfield
Marie-Francine Moens
Chris Paice
Kerry Rodden
Mark Sanderson
Martijn Spitters
Padmini Srinivasan
Tomek Strzalkowski

SCHEDULE

9.15	**Registration & Coffee**
10.00	**Keynote Lecture** (Chair: Jos Dumortier – K.U.Leuven) Karen Sparck Jones University of Cambridge, UK **Language and information: old ideas revisited and new ones considered.**
10.50	**Coffee Break**
11.00	**Session I** (Chair: Marie-Francine Moens – K.U.Leuven)
11.00	J. List & A.P. de Vries Center for Mathematics and Computer Science, Amsterdam **XML-IR: Coverage as part of relevance.**
11.20	R. Kosala, M. Bruynooghe, J. Van den Busche & H. Blockeel K.U.Leuven & University of Limburg **Information extraction from web pages based on k-testable tree automaton induction.**
11.40	J. Kamps, M. Marx, C. Monz & M. de Rijke University of Amsterdam **Exploiting structure for information retrieval.**
12.00	L.V. Boldareva, D. Hiemstra & W. Jonker University of Twente **A scalable and efficient content-based multimedia retrieval system.**
12.20	E. van den Broek, L. Vuurpijl, P. Kisters & J.C.M. von Schmid University of Nijmegen **Object-based image retrieval: color-selection exploited.**
12.40	**Lunch**
13.40	**Session II** (Chair: Franciska de Jong – University of Twente)
13.40	J. Kamps, C. Monz & M. de Rijke University of Amsterdam **Combining morphological and n-gram evidence for monolingual document retrieval.**
14.00	K. Van Belleghem, A. Vandecandelaere & L. Dehaspe PharmaDM N.V. **AlphaDMax: an integrated tool for biomedical information retrieval and extraction**.
14.20	R. Angheluta & M.-F. Moens K.U.Leuven **A study of synonym replacement in news corpora.**
14.40	B. van Gils & H. Paijmans **University of Nijmegen & Tilburg University** **Creating document surrogates with lexical cohesion.**
15.00	C.H.A. Koster & M. Seutter University of Nijmegen **Making phrases work.**
15.20	**Coffee Break**
15.40	**Session III** (Chair: Wessel Kraaij – T.N.O. Delft)
15.40	A. Diekema Syracuse University, USA **Spurious matches in Dutch cross-language information retrieval: lexical ambiguity, vocabulary mismatch, and other causes of translation error.**
16.00	D. Vervenne et al. Hogeschool Gent & Ghent University **URUK, a platform for causal text retrieval.**
16.20	M. Volk & P. Buitelaar Eurospider, Switzerland & DFKI, Germany **A systematic evaluation of concept-based cross-language information retrieval in the medical domain.**
16.40	L. Braun, F. Wiesman & I. Sprinkhuizen-Kuyper University of Maastricht **Information retrieval from historical corpora.**
17.00	**End**

CONTENT

Language and information: old ideas revisited and new ones considered

Karen Sparck Jones
University of Cambridge
United Kingdom
Karen.Sparck-Jones@cl.cam.ac.uk

Abstract

Statistically-based approaches to language and information processing (LIP) were novelties in the 1960s. They were hard to implement in practice, through lack of data and machine power, and hard for many to accept in theory. But they were successfully applied to document retrieval, showing how LIP tasks could be viewed in a productive new way. In the last decade there have been striking further developments, extending statistically-based LIP theory and practice in novel and challenging directions. The talk will illustrate some of the possibilities and problems involved.

References

M. Banko, V. Mittal, and M. Witbrock, `Headline generation based on statistical translation',

{\em ACL 2000: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics}, 2000, 318-325.

A. Berger and J. Lafferty, `Information retrieval as statistical translation', SIGIR 99, Proceedings of the 23rd International ACM-SIGIR Conference on Research and Development in Information Retrieval, 1999,222-229.

B. Boguraev and C. Kennedy, `Salience-based content characterisation of text documents', in Mani and Maybury, 1999.

H. Borko, `Research on computer-based classification systems' in Classification research, (Ed. P. Atherton), Copenhagen: Munksgaard, 1965.

P.F. Brown et al. `Class-based n-gram models of natural language', Computational Linguistics, 18(4), 1992, 467-680.

K.W. Church and P. Hanks, `Word association norms, mutual information, and lexicography', Computational Linguistics, 16(1), 1990, 22-30.

W.B. Croft (ed.), Advances in information retrieval, Dordrecht: Kluwer.

S.F. Dennis, `The design and testing of a fully-automatic indexing-searching system for documents consisting of expository text', in Information retrieval - a critical view, (Ed. G. Schechter), Washington, DC: Schecter, 1967.

L.B. Doyle, `Semantic road maps for literature searchers', Journal of the ACM, 8(3), 1961, 404-417.

S.T. Dumais, `Latent semantic indexing (LSI): TREC-3 report', TREC-3, Proceedings of the Third Text REtrieval Conference, (Ed. D.K.Harman), Special Publication 500-225, Gaithersburg, MD: National Institute of Standards and Technology, 1995, 219-230.

U. Germann et al. `Fast decoding and optimal decoding for machine translation", ACL 2001, Proceedings of the 39th Annual Conference of the Association for Computational Linguistics, 2001, 228-235.

K.E. Harper, `Measurement of similarity between nouns', RM-4532-PR, Rand Corporation, Santa Monica, 1965.

K. Knight and D. Marcu, `Statistics-based summarisation - Step 1: sentence compression', AAAI-2000, Proceedings of the 17th National Conference on Artificial Intelligence, 2000, 703-710.

Knight - see Germann et al, 2001.

Lee, L. and F. Pereira, `Distributional similarity models: clustering versus nearest neighbour', ACL 99, Proceedings of the 37th Annual Conference of the Association for Computational Linguistics, 1999, 33-40.

H.P. Luhn, 1957 quotation, p 96 in Schultz, 1968; papers, see Schultz.

I. Mani and M.T. Maybury (eds), Advances in automatic text summarisation', Cambridge, MA: MIT Press, 1999.

Marcu - see Knight and Marcu, 2000.

M. Masterman, R.M. Needham and K. Sparck Jones, `The analogy between mechanical translation and information retrieval', Proceedings of the International Conference on Scientific information, (1958), Washington, DC: National Academy of Sciences - National Research Council, 1959, 917-935.

R. Mihalcea and D. Moldovan, `A method of word sense disambiguation of unrestricted text', ACL 99, Proceedings of the 37th Annual Conference of the Association for Computational Linguistics, 1999, 152-158.

Pereira - see Lee and Pereira, 1999.

D. Prescher, S. Riezler and M. Rooth, `Using a probabilistic class-based lexicon for lexical ambiguity resolution', COLING 2000, Proceedings of the 18th International Conference on Computational Linguistics, 2000, 649-655.

Robertson - see Sparck Jones et al, 2000.

Rooth - see Prescher et al, 2000.

N. Sager, `Natural language information formatting: the automatic conversion of texts to a structured database', Advances in Computers 17, (Ed. M. Yovits), New York: Academic Press, 1978, 89-162.

G. Salton (ed), The SMART retrieval system, Englewood Cliffs, NJ: Prentice-Hall.

H. Schuetze, `Automatic word sense discrimination', Computational Linguistics, 24(1), 1998, 97-123.

C.K. Schultz (ed), H.P. Luhn : Pioneer of information science, New York: Spartan, 1968.

K. Sparck Jones, Synonymy and semantic classification, (1964), Edinburgh: Edinburgh University Press, 1986.

K. Sparck Jones, `Notes and references on early classification work', ACM SIGIR Forum, 25(1), 1991, 10-17.

K. Sparck Jones and P. Willett (eds), Readings in information retrieval, San Francisco: Morgan Kaufmann, 1997.

K. Sparck Jones, `Automatic summarising: factors and directions', in Mani and Maybury, 1999.

K. Sparck Jones, G. Gazdar and R. Needham (eds), `Computers, language and speech: formal theories and statistical data', Philosophical Transactions of the Royal Society of London, Series, A,

Vol. 358, Number 1769, 2000, 1225-1431.

K. Sparck Jones, S. Walker and S.E. Robertson, `A probabilistic model of information retrieval: development and comparative experiments. Parts 1 and 2', Information Processing and Management 36 (6), 2000, 779-840.

M.E. Stevens, Automatic indexing: a state-of-the-art report, Monograph 91, Washington, DC: National Bureau of Standards, 1965.

H.E. Stiles, `The association factor in information retrieval', Journal of the ACM, 8, 1961, 271-279.

Witbrock - see Banko et al, 2000.

XML-IR: Coverage as Part of Relevance

Johan List and Arjen P. de Vries

Center for Mathematics and Computer Science (CWI)
Data Mining and Knowledge Discovery (INS1)
P.O.Box 94079, 1090GB Amsterdam, The Netherlands
{j.a.list, a.p.de.vries}@cwi.nl

Abstract

Relevance is a multidimensional concept, not only consisting of linguistic-only properties but also enriched by various other relevance dimensions that are largely orthogonal to the *topicality* (i.e. content-based relevance) of a document. The question is how to capture such dimensions of relevance effectively in a retrieval model.

In this paper we propose a model where we regard additional relevance dimensions independent (given a document instantiation). The independence assumption is made because it is very difficult to predict influence of relevance dimensions a-priori. The model also reflects our belief that modeling of additional knowledge with prior probabilities (in a probabilistic setting) is a counter-intuitive approach because of 1) the orthogonality of additional relevance dimensions and 2) the difficulty to reliably (re-)estimate dimension models, due to possible 'noise' introduced by non-dimension related priors.

Also, relevance feedback needs to be able to handle multiple dimensions of relevance effectively. Feedback in the model is done with dimension-specific feedback sets.

We can only report informally on the results of our model; based on the experimental scenarios performed, the model is appearing to perform very well, although quantitative assessments using an assessed collection are necessary to confirm this and draw further conclusions.

1 Introduction

We believe a clear distinction should be made between topicality and 'relevance', where topicality (i.e. content-based or linguistic similarity) is an approximation of relevance and can be seen as only a single dimension of relevance. An information need can include a variety of extra dimensions, not necessarily all linguistic in nature. Mizarro (Mizarro, 1998) gives examples of such dimensions, including comprehensibility (style or difficulty of the text) and quantity (how much information does the user want; this is measured in a.o. the size of documents and the number of documents returned to the user). Our aim is to capture such dimensions of relevance effectively in our retrieval model.

A closely related issue is the notion of 'coverage', as e.g. used in the INEX XML Retrieval initiative. Coverage is defined as how much of the document component is relevant to the topic of request. Estimating the right amount of coverage for a search request plays a significant role in the case of structured document retrieval where the desirable retrieval unit is not known a-priori. Effective determination of the retrieval unit is a key issue which distinguishes structured document retrieval from traditional retrieval (where the retrieval unit is fixed a-priori).

To further illustrate the retrieval unit problem, consider a short motivating example. Let us assume we have a document consisting of a section with three subsections, and each subsection containing five paragraphs. Now, the system that estimates topicality identifies three relevant paragraphs in a subsection. The open question is then whether to return the three separate paragraphs, or the single subsection containing these as well as the remaining two (possibly irrelevant) paragraphs. The additional context provided by the full subsection may be more desirable for a user than the individual three paragraphs in isolation.

Assume a user is trying to solve the retrieval unit question and decides to use coverage as an additional relevance dimension. For modeling coverage, the user decides to regard coverage as a function of both topicality of document components and the size of document components (size being an aspect of the quantity dimension). The user reasons that:

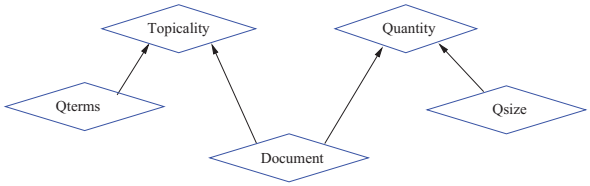


Figure 1: Encoding of additional relevance dimensions. Note that *Qterms* and *Qsize* denote information given by the query (query terms and preferred component size).

- the shorter the document component is, the more likely it will not contain enough information to fulfill the information need;
- the longer the document component is, the more likely it is that distilling the topically relevant information will take substantial more reader effort.

Now, when a user is ranking a document collection with regard to coverage, a ranking is performed against a combination of both topicality relevance and quantity relevance (where the user uses document component size as a representation of quantity). In probabilistic terms we are calculating the probability of complete relevance of a document component, given topicality relevance and quantity relevance.

More generally, we propose a probabilistic model where, given a document instantiation, we regard the dimensions of relevance independent based on the assumption that without user interaction, we cannot say anything about the influence of each dimension on user satisfaction. Traditional information retrieval uses only a single dimension of this model, namely topicality. The model is visualized in Figure 1.

2 Retrieval Model

2.1 Modeling Additional Relevance Dimensions

Firstly, for modeling additional relevance dimensions, we need a probabilistic description. The model in Figure 1 leads to the following. When $P(R_t|D_d)$ is the probability of topical relevance given document d and $P(R_q|D_d)$ is the probability of quantity relevance given document d, then we can calculate a joint probability of 'complete' relevance or user satisfaction as:

$$P(D_d,R_t,R_q,Q_{terms},Q_{size}) = P(R_t|D_d,Q_{terms})P(R_q|D_d,Q_{size})P(D_d)$$

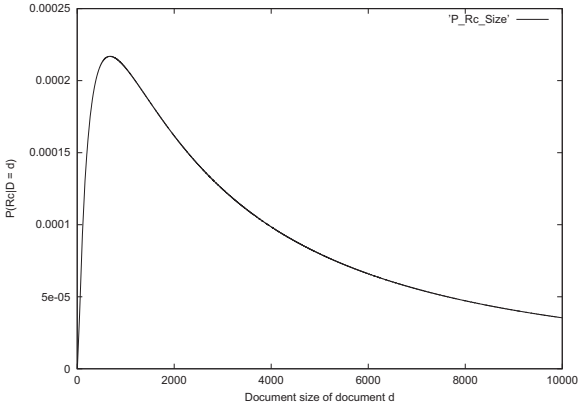


Figure 2: The log-normal distribution used for modeling the quantity dimension

Looking at the motivating example in Section 1 and especially the user reasoning for modeling the quantity dimension, we decided to use a log-normal distribution. It is a distribution characterized by both a steep slope at the start and a long tail (as can be seen from Figure 2). The steep slope at the start reflects the 'punishing' behavior we want to model for (extremely) short document components. The long tail reflects that we do want to punish long document components, but not as harshly as extremely short ones (since these still might be useful, even while taking more reader effort to distill the relevant information).

Secondly, we need the modeling parameter for the distribution itself. We have chosen for component size, but other possibilities include:

- the depth of the document component in the tree structure, where we want to penalize components present deep in the trees (generally small components and too specific) or components present high in the trees (generally large components and too broad);
- the number of children of a document component. A short document component containing a large amount of children highly likely contains a diversified mix of information and a could be less desirable for a user than a more homogeneous component.

Lastly, we need to integrate additional dimensions of relevance into our retrieval model. A general approach for modeling additional knowledge is using the prior probabilities (in a probabilistic setting). For ex-

ample, Westerveld et al. (Westerveld et al., 2001) used this strategy successfully to increase the likelihood of finding entry pages in a Web retrieval task. Also, a prior on document length improved retrieval performance at TREC-style experiments (Hiemstra, 2000), based on the assumption that longer documents have a higher probability of containing relevant information.

Since we regard additional dimensions of relevance independent given an document instantiation, the model in Figure 1 also reflects our belief that modeling other dimensions of relevance through prior probabilities is a somewhat counter-intuitive approach. For the rest of the discussion, note that we have used a language modeling approach for modeling topicality (see subsection 2.2).

Firstly, non-linguistic dimensions of relevance are (largely) orthogonal to the topicality, estimated by the language model. The orthogonality assumption is fueled by research performed in the user modeling and relevance areas (see a.o. (Belkin et al., 1982a), (Belkin et al., 1982b), (Bruce, 1994), (Barry, 1994). Again, the common thread in this work is the fact that relevance is a multidimensional concept, of which topicality is only a single one. Mizarro (Mizarro, 1998) names other, possible non-topical dimensions *abstract characteristics of documents*, constructed independently from the particulars of the database or collection at hand. In other words: other, non-topical dimensions are constructed independently from the language models present in the documents of a collection.

Secondly, encoding additional knowledge in prior probabilities makes it more difficult to reliably estimate dimension models, due to the possible noise non-dimension related prior probabilities introduce.

2.2 Topicality Modeling

The model used for describing topicality of documents is a probabilistic model, the statistical language model described by Hiemstra (Hiemstra, 2000). The main idea of this model is to extract and to compare document and query models and determine the probability that the document generated the query. In other words, the statistical language model extracts linguistic information and is suited for modeling of the topicality dimension of the information need.

In deriving document models for all of the documents in the collection, we regarded every subtree present in the collection as a separate document. The probability of topical relevance $P(R_t|D_d, Q_{terms})$ where Q_{terms} consists of the set of query terms $\{T_1, \cdots, T_n\}$ is calculated with:

$$P(R_t|D_d, Q_{terms}) = P(R_t|D_d, T_1, \cdots, T_n) = P(D_d) \prod_{i=1}^{n} P(I_i) P(T_i|I_i, D_d)$$

where $P(I_i)$ is the probability that a term is important (the event I has a sample space of $\{0, 1\}$).

We follow the reasoning of Hiemstra (Hiemstra, 2000) to relate the model to a weighting scheme (tf.idf-based). After some manipulation of the model we get:

$$P(D_d, T_1, \cdots, T_n) \propto P(D_d) \prod_{i=1}^{n} (1 + \frac{\lambda P(T_i|D_d)}{(1-\lambda)P(T_i)})$$

As estimators for $P(D_d$, $P(T_i|D_d)$ and $P(T_i)$ we used:

$$P(D_d) = \frac{1}{n} \tag{2.1}$$

$$P(T_i|D_d) = \frac{tf_{i,d}}{\sum_i tf_{i,d}} \tag{2.2}$$

where n is the number of documents, $tf_{i,d}$ is the term frequency of term i in document d and $\sum_i tf(i, d)$ is the length of document d.

For $P(T_i)$ we used:

$$P(T_i) = \frac{df_i}{\sum_i df_i} \tag{2.3}$$

where df_i is the document frequency of term i.

Filling in the likelihood estimators gives us the following model for topicality (with a constant λ for all terms):

$$P(R_t|D_d, Q_{terms}) = P(R_t|D_d, T_1, \cdots, T_n) \propto \sum_{i=1}^{n} \log(1 + \frac{\lambda}{1-\lambda} \frac{tf_{i,d}}{\sum_i tf_{i,d}} \frac{\sum df_i}{df_i})$$

We used a very simple query model resulting in query term weights represented with $tf_{i,q}$, the term frequency of term i in query q.

3 Relevance Dimensions and Relevance Feedback

Explicit relevance feedback is the main entry point for learning additional relevance dimensions, since we cannot assume a system has knowledge of other relevance dimensions at the initial query stage.

We make a distinction between relevance feedback as a *directed* process, in the case of a user identifying relevant documents and feeding that information back to the system or relevance feedback as an *undirected* process, in the case of taking top-ranked documents and using that collection for query term expansion.

For the purpose of relevance feedback, let us assume we have a user examining the result set after an initial search and this user is judging the results set on topicality and quantity. We can distinguish three possible decisions by this user when judging a result:

- The user sees the result as correct regarding topicality and not quantity;
- The user sees the result as correct regarding quantity and not topicality;
- The user sees the result as contributing to both relevance dimensions.

If the first situation applies and a user is giving feedback on topicality alone and not on quantity, we simply can re-estimate the language model parameters and disregard quantity influence altogether. If the second situation applies and a user is giving feedback on quantity alone and not topicality, we simply can re-estimate the model parameters of the quantity model and we can leave the language model parameters as they were. The situation becomes more difficult in the third case, when a user is giving feedback based both on topicality and quantity. Feedback in this situation can be visualized with the adapted model of Figure 3. In relevance feedback, the user can be regarded as specifying the probability distributions of topicality and quantity, given that the document *is* 'completely' relevant.

We have only experimented with undirected feedback without further specification of probability distributions for relevance feedback. Then undirected feedback will only increase performance when giving feedback per dimension. To explain this further, consider the user from the motivating example having performed a search. The ranking has been performed on quantity, the combination of topicality and component size. The question if the quantity-ranked set

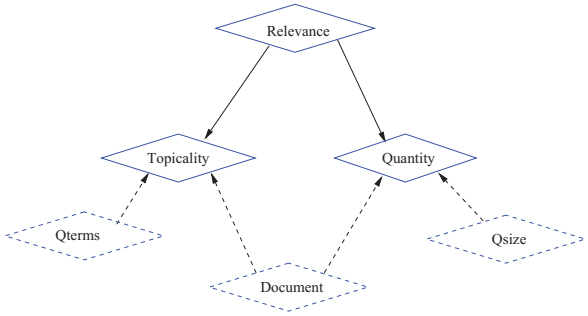


Figure 3: Relevance feedback

of document components can be used effectively for blind feedback.

Compare the document set ranked on quantity with a document set ranked on topicality only. Since it is possible that documents with a lower topicality-only score get a higher rank in the quantity ranking (because of a better size), using the quantity-ranked document set for e.g. topicality feedback will worsen the quality of the (estimated) topicality model parameters.

To update our model given relevance feedback, we perform re-estimation of the separate dimension models. We see the document feedback set as a collection of *content sources*. Each content source is characterized by a collection of properties which map to relevance dimensions. For example, when considering topicality and quantity, we characterize each content source with two properties R_t (topicality) and R_q (quantity). Recall that we consider additional relevance dimensions independent given a document instantiation. We can characterize a document D_r in the feedback set, being characterized by topicality and quantity as:

$$P(D_d, R_t, R_q, Q_{terms}, Q_{size}) = \\ P(R_t|D_d, Q_{terms})P(R_q|D_d, Q_{size})P(D_d)$$

We assigned a uniform distribution to $P(D_d)$ so we can safely leave this out of the model without affecting the ranking. Using the language model for topicality (including a λ_i, varying per term) and the log-normal for quantity gives us for $P(D_d, R_t, R_q, Q_{terms}, Q_{size})$:

$$\left[P(D_d)\prod_{i=1}^{n}\lambda_i P(T_i|D_d) + (1-\lambda_i)P(T_i)\right] \\ P(R_q|D_d, Q_{size})$$

We now want to find the set of model parameters which maximize the likelihood (with r feedback documents):

$$\prod_{e=1}^{r}\Big[P(D_e)\prod_{i=1}^{n}\lambda_i P(T_i|D_e)+(1-\lambda_i)P(T_i)\Big] P(R_q|D_e,Q_{size})$$

When we work the model out further for the topicality and quantity dimensions only (where quantity is modeled by a log-normal distribution) and leave out $P(D)$ since it is uniform, we want to maximize the likelihood L (with r feedback documents):

$$\prod_{e=1}^{r}\Big[\prod_{i=1}^{n}\lambda_i P(T_i|D_e)+(1-\lambda_i)P(T_i)\Big]P(R_q|D_e,Q_{size})$$

or the log-likelihood Λ:

$$\sum_{e=1}^{r}\sum_{i=1}^{n}\log(\lambda_i P(T_i|D_e)+(1-\lambda_i)P(T_i))+ \sum_{e=1}^{r}\log P(R_q|D_e,Q_{size})$$

Due to the independence assumption, we can divide the estimation problem into two subproblems and update each dimension separately (with r feedback documents):

$$\lambda_i^* = \arg\max_{\lambda_i}\sum_{e=1}^{r}\log(\lambda_i P(T_i|D_e)+(1-\lambda_i)P(T_i)) \tag{3.1}$$

$$\{\mu^*,\sigma^*\} = \arg\max_{\{\mu,\sigma\}}\sum_{e=1}^{r}\log P(R_q|D_e,Q_{size}) \tag{3.2}$$

For the first estimation problem in equation 3.1 we can use EM (Hiemstra, 2000). For iteration p we use as E-step (with r feedback documents):

$$k_i = \sum_{e=1}^{r}\frac{\lambda_i^{(p)}P(T_i|D_e)}{(1-\lambda_i^{(p)})P(T_i)+\lambda_i^{(p)}P(T_i|D_e)}$$

and as M-step:

$$\lambda_i^{(p+1)} = \frac{k_i}{r}$$

For the second estimation problem in equation 3.2 we can perform a maximum likelihood estimation procedure for μ and σ as follows. The usual approach to estimation of a log-normal distribution $LN(\mu,\sigma^2)$ is to consider a new data sample Y_i, where $Y_i = \log X_i, i = 1\cdots n$. The estimation then becomes the estimation of a normal distribution $N(\mu,\sigma)$ for which we can easily derive the maximum likelihood estimators.

The probability density function for a normal distribution Y with mean μ and standard deviation σ is described by

$$f(Y) = \frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{1}{2}(\frac{Y_i-\mu}{\sigma})^2)$$

The likelihood function is given by:

$$\begin{aligned} L &= \prod_{i=1}^{n}\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{1}{2}(\frac{Y_i-\mu}{\sigma})^2) \\ &= (\frac{1}{\sigma\sqrt{2\pi}})^n\exp(-\frac{1}{2}\sum_{i=1}^{n}(\frac{Y_i-\mu}{\sigma})^2) \end{aligned}$$

The log-likelihood is given by:

$$\begin{aligned} \Lambda &= \log(\frac{1}{\sigma\sqrt{2\pi}})^n\exp(-\frac{1}{2}\sum_{i=1}^{n}(\frac{Y_i-\mu}{\sigma})^2) \\ &= \log(\frac{1}{\sigma\sqrt{2\pi}})^n + \log(\exp(-\frac{1}{2}\sum_{i=1}^{n}(\frac{Y_i-\mu}{\sigma})^2)) \end{aligned}$$

Working this out further gives us:

$$\Lambda = -n\log\sigma - \frac{n}{2}\log 2\pi - \frac{1}{2}\sum_{i=1}^{n}(\frac{Y_i-\mu}{\sigma})^2$$

Taking the partial derivatives with respect to μ and σ gives us:

$$\frac{\partial(\Lambda)}{\partial(\mu)} = \frac{1}{\sigma^2}\sum_{i=1}^{n}(Y_i-\mu)^2$$

$$\frac{\partial(\Lambda)}{\partial(\sigma)} = -\frac{n}{\sigma} + \frac{1}{\sigma^3}\sum_{i=1}^{n}(Y_i-\mu)^2$$

We can set the partial derivatives to 0 and solve for μ and σ (we know that the original normal function is

Table 1: Experimentation scenarios.

Scenario	Retr. Unit	Dimension(s)
V_1	$\{tr(article)\}$	R_t
V_2	$\{tr(*)\}$	R_t
V_3	$\{tr(*)\}$	R_t, R_q

positive for all values in the range, and we know there is a single (non-local) maximum). This gives us:

$$\mu^* = \frac{1}{n}\sum_{i=1}^{n} Y_i$$

$$\sigma^* = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \mu)^2}$$

4 Experimental Work

We participated in INEX[1] and implemented an XML retrieval system based on Monet, a main-memory database kernel.

With our system, we performed three experimentation scenarios. The first scenario mimicked 'flat-document' retrieval of articles, i.e. retrieval of documents which possess no structure whatsoever. The second scenario regarded all subtrees or transitive closures in the collection as separate documents.

For the third scenario we re-used the result sets of the second run and used the log-normal distribution to model the quantity dimension. To penalize the retrieval of extremely long document components (this in contrast with the language model that assigns a higher probability to longer documents), as well as extremely short document components, we set the mean at 500 (representing a user with a preference for components of 500 words).

In all three scenarios we used the statistical language model of subsection 2.2 to model topicality.

Table 1 summarizes our experimentation scenarios. Note that $tr(c)$ denotes the transitive closure of a document component with root c and $tr(*)$ denotes the transitive closures of all subtrees present in the original XML syntax trees.

[1]XML Retrieval Initiative, see `http://qmir.dcs.qmw.ac.uk/inex/index.html`

5 Conclusions and Future Work

From an informal look into our results, modeling coverage by using a combination of topicality and quantity (in terms of component size), using a subjective probability function for the latter, seems to work pretty well. To be able to make this conclusion more firmly, we need to perform further experiments on coverage estimation, as well as other dimensions of relevance.

For quantitatively backing up our model, we need evaluation results of the runs as well (sadly not available at the time of finishing this paper). We plan to report on the retrieval performance in our INEX workshop paper (List and de Vries, 2003).

In future work, we intend to perform experimentation with relevance feedback and extend the model further for other dimensions and ultimately, for the mapping of user context to retrieval model parameters.

References

C.L. Barry. 1994. User-defined Relevance Criteria: An Exploratory Study. *Journal of the American Society for Information Science*, 45(3):149–159.

N.J. Belkin, R.N. Oddy, and H.M. Brooks. 1982a. ASK for Information Retrieval: Part 1. Background and Theory. *Journal of Documentation*, 38(2):61–71.

N.J. Belkin, R.N. Oddy, and H.M. Brooks. 1982b. ASK for Information Retrieval: Part 2. Results of a Design Study. *Journal of Documentation*, 38(3):145–164.

H.W. Bruce. 1994. A Cognitive View of the Situational Dynamism of User-centered Relevance Estimation. *Journal of the American Society for Information Science*, 45(3):142–148.

D. Hiemstra. 2000. *Using Language Models for Information Retrieval*. Ph.D. thesis, University of Twente, Twente, The Netherlands.

J.A. List and A.P. de Vries. 2003. CWI at INEX 2002 (to appear, January 2003).

S. Mizarro. 1998. How Many Relevances in Information Retrieval? *Interacting With Computers*, 10(3):305–322.

T. Westerveld, W. Kraaij, and D. Hiemstra. 2001. Retrieving Web Pages using Content, Links, URLs and Anchors. In *NIST Special Publication 500-250, The 10th TREC Retrieval Conference (TREC 2001)*.

Information Extraction from Web Pages Based on k-testable Tree Automaton Induction (extended abstract)

Raymond Kosala[1], **Maurice Bruynooghe**[1], **Jan Van den Bussche**[2], **Hendrik Blockeel**[1]
[1] Department of Computer Science, Katholieke Universiteit Leuven
Celestijnenlaan 200A, B-3001 Leuven
{raymond,maurice,hendrik}@cs.kuleuven.ac.be
[2] WNI, University of Limburg, Universitaire Campus, B-3590 Diepenbeek
jan.vandenbussche@luc.ac.be

Abstract

Information extraction refers to the process of extracting specific pieces of information from a document; for instance, extracting from a text the names of the authors. Much of the work on information extraction from HTML or XML documents uses methods for processing strings, such as finite automata. However, as these documents have a tree structure, it seems natural to exploit this structure by using techniques that parse trees, not strings. Tree automata are a suitable device for this. In this paper we explore methods for automatically learning tree automata that can be used for extracting specific information in tree-structured documents. We present an overview of several algorithms we have experimented with, as well as experimental results.

1 Introduction

Information extraction is the process of extracting specific information from documents. For example, one task might be to extract, from a text on some event, the date and location of the event. A wrapper is a procedure that takes as input a document and, using information extraction techniques, yields a more structured description of the document, for instance, a form with pre-defined fields that have an unambiguous meaning and contain the essential information in the document.

Information extraction interacts with information retrieval in a number of ways. First, it can be a preprocessing step for document retrieval. As argued by Lin and Ho (2002), many websites contain redundant information that can mislead search engines. An information extraction system can be used to filter many irrelevant or redundant features from a document and thus improve the indexing process. One could also consider the use of wrappers to create structured summaries of documents that can be used by document retrieval systems.

As a post-processor, an information extraction system can be used to re-score the relevance ranking of the documents stored by information retrieval systems. For instance, following the ideas in Bear et al. (1997), one could use an information extraction system to extract the most relevant nodes of a HTML document, count how many nodes are extracted, and re-score the relevance of the document based on that. Obviously, information extraction is also useful for retrieving information on a more fine-grained level than complete documents.

Information extraction procedures or wrappers are typically constructed for a specific kind of documents, because for information extraction to work well, documents must be sufficiently homogeneous with respect to semantics and/or format. Obviously, it is cumbersome to manually write wrappers for many different types of documents. Hence, there is a large interest in automatic construction of wrappers using machine learning techniques; this is referred to as wrapper induction. To automatically learn wrappers, one needs a formalism that is sufficiently powerful to represent a broad class of algorithms, but still limited enough to be able to learn them automatically.

A lot of work in the field of information extraction focuses on extracting information from unstructured text. In that case it is natural to consider the use of finite automata as wrappers: these handle strings, they have a reasonably large expressiveness and there exist algorithms for learning them automatically (Murphy, 1996). On the Web, however, many documents are tree-structured; this is specifically the case for HTML and XML documents. Although they can be handled by processing the document as a string, it seems natural to try to exploit the tree structure of these documents. Tree automata are then an obvious choice: they are the counterpart of finite automata for tree structured data.

Gottlob and Koch (2002) have recently related induction of tree automata to wrapper induction approaches, and found that all existing wrapper induction methods are special cases of induction of tree automata. This provides additional motivation for the use of tree automata for information extraction from web documents.

Kosala et al. (2002) investigate a number of approaches to induction of tree automata from HTML documents. This extended abstract is based on that

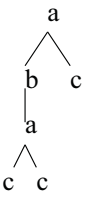


Figure 1: The tree represented in textual notation as $a(b(a(c,c)),c)$.

paper; we here focus less on technical details of the algorithms but describe them at a more intuitive level.

The remainder of this text is structured as follows. In Section 2 we briefly introduce tree automata and explain how they can be used for information extraction. In Section 3 we discuss the preprocessing of HTML documents that we found necessary in order to make our approach practically feasible. In Section 4 we turn our attention to the induction of tree automata, again in the context of information extraction. We introduce several induction algorithms with which we have experimented. These experiments are presented in detail in Section 5. In Section 6 we conclude.

2 Tree Automata

A ranked label f/n is a symbol f with a natural number n (called its rank or arity) associated with it. Let V denote a set of ranked labels, and V_k its subset of labels of rank k. $V = \cup_k V_k$. A tree over the set V is defined as follows: a label of rank 0, say $f/0$, is a tree (denoted f); if f/n is a label of rank f/n and $t_1, \ldots, t_n$ are trees, then $f(t_1, \ldots, t_n)$ is a tree; nothing else is a tree.

For example, with $V = \{a/2, b/1, c/0\}$, $a(b(a(c,c)),c)$ is a tree over V. It is depicted graphically in Figure 1.

A deterministic tree automaton (DTA) M is a quadruple (V, Q, Δ, F) where V is a set of ranked labels, Q is a finite set of states, $F \subseteq Q$ is a set of final or accepting states, and $\Delta : \cup_k (V_k \times Q^k) \to Q$ is the so-called transition function of M. Thus, if q_1, q_2 are states, then $\Delta(a, q_1, q_2) = q_2$ defines a single transition in Δ. We will also use the notation $a(q_1, q_2) \to q_2$.

A DTA processes trees bottom up. It starts with assigning states to leaves, according to the transition function. Whenever all children of a node have had a state assigned to them, the node itself is also assigned a state (again in accordance with the transition function). This process goes on until as many nodes as possible are assigned a state (nodes for which no transition applies may exist). We say that a tree is accepted if the root is assigned an accepting state (a state $q \in F$). The set of all trees accepted by a tree automaton M is called the language defined by M.

For example, for $V = \{a/2, b/1, c/0\}$, $Q = \{q_1, q_2\}$, $F = \{q_2\}$, and $\Delta = \{a(q_1, q_1) \to q_1, a(q_1, q_2) \to q_2, a(q_2, q_1) \to q_2, a(q_2, q_2) \to q_2, b(q_1) \to q_2, b(q_2) \to q_2, c \to q_1\}$, the tree automaton (V, Q, F, Δ) accepts those trees that have a b in them, and only those.

Given that a tree automaton accepts or rejects a whole tree, how can we use it for information extraction, where the aim is really to return a specific node in the tree? We follow the procedure proposed by Freitag (1997): we use a special symbol x to denote the field of interest. The automaton should accept a tree if and only if x occurs in the tree in the position of a node that is to be extracted. By consecutively substituting x for each node in a tree and running the tree automaton to see whether the resulting tree is accepted, we can identify all fields of interest.

We have now introduced the concept of tree automata and how they can be used for extracting a single node from a tree. In the next section we discuss how this approach can be followed in the context of information extraction from web pages.

3 Preprocessing of HTML Documents

The task we face is the following. We wish to learn a tree automaton that recognizes when the marker x is in the place of the information to be extracted. The input of the learning process is a set of examples, more precisely HTML documents with the fields of interest indicated. These would typically be provided by a human user. We assume that there are only positive examples, that is, the user does not need to provide examples of fields that are not to be extracted.

From the set of positive examples, our learning algorithm should construct a tree automaton that accepts all these examples, and generalizes over them in the sense that it should also accept previously unseen documents where the marker is in the right location.

It is not feasible to learn a tree automaton from the original HTML documents, for two reasons. First, a typical HTML document contains a limited set of markup tags, but at the lowest level of the tree an unlimited set of symbols (any text) can occur. To limit the complexity of the tree automaton to be learnt, we need to limit the size of the set V by preprocessing the documents.

A step that drastically reduces the size of V, is to replace all plain text by the symbol CDATA (except the text to be extracted, which is replaced by the marker x). In this case the information extraction procedure has only the exact structure of the document and the markup tags as guidance. This seems

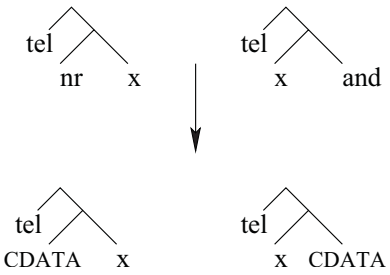


Figure 2: An example of the substitution of CDATA for text. "tel" is the closest (to x) invariant label and is kept as a distinguishing context, other text is changed into CDATA.

a very difficult task, and indeed experimental results with this approach show that overgeneralization occurs, that is, the induced tree automaton accepts many more trees than it should. This problem is alleviated in the following way.

The field to be extracted in HTML documents is often recognizable by some specific text occurring near it. We call this the distinguishing context. Our approach is to replace all text by CDATA except this distinguishing context, which is kept intact. The distinguishing context is determined automatically, using the following heuristic. The preprocessor looks for the invariant text label that is nearest to the field of interest and occurs at the same distance from the field of interest in all examples. If no such text is found, no context is used and all text is turned into CDATA. Figure 2 shows an example.

A second problem we face when trying to apply automaton induction algorithms in the context of HTML documents, is that HTML trees are unranked: a root with a given label can have an indefinite number of children. We have only ranked tree automaton inference algorithms at our disposal (induction of unranked tree automata is a different and more complex problem).

Fortunately, unranked trees can easily be transformed into ranked trees. Multiple children of a single node are then handled by putting them in a tree structure. More specifically, in the transformed tree, each node of the original tree occurs with as left child its leftmost child in the original tree (if any) and as right child the sibling that occurs immediately to its right (if any). Figure 3 illustrates the process. A formal definition of this transformation is given by Kosala et al. (2002). Note that in the resulting binary tree it is essential to distinguish left and right subtrees of a node; if a node has only one child (that is, the other one is empty) it must still be clear whether it is the left or right child.

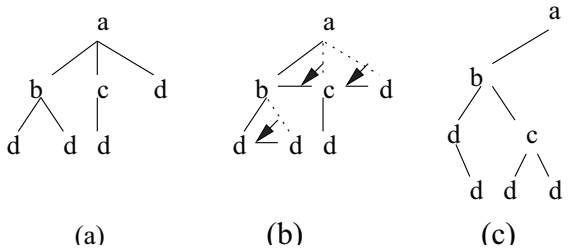


Figure 3: Transformation of an unranked tree into a ranked tree. (a) The original tree. (b) Nodes are linked to their immediately preceding sibling instead of their parent, except for the leftmost child of that parent. (c) The new tree.

4 Inducing Tree Automata

We have seen what tree automata are and how to represent HTML documents as ranked trees. The next topic we focus on is: how can one learn tree automata from example trees?

We have implemented several tree automaton induction algorithms and experimented with them. The basic algorithm that we started from, is the so-called k-testable algorithm developed by Rico-Juan et al. (2000).

A tree automaton (or the language it defines) is k-testable if it can be decided whether a tree belongs to the language or not by looking only at trees of height at most k (we define the height of a tree as the maximum number of nodes on any path from root to leaf).

For instance, if L is a 2-testable language then it is impossible that $t_1 = a(b(c), b(c))$ and $t_2 = a(b(d), b(d))$ are in L but $t_3 = a(b(c), b(d))$ is not. The reason is that $\{a(b, b), b(c), b(d)\}$ are the subtrees of height 2 that occur in t_1 and t_2 and that no other subtrees occur in t_3.

A k-testable automaton never needs more states than there are trees of height $k-1$ or less. Given a number of trees and a fixed k, the most specific k-testable automaton[1] that accepts all these trees is the one that has a different state for each occurring subtree of height at most $k-1$, and allows only the state transitions observed in the examples. The algorithm by Rico-Juan et al. constructs this automaton: it just collects all different subtrees of height at most $k-1$ and creates a different state for each, assigns these states to all nodes of the tree, and subsequently adds all observed state transitions to Δ. It does this as follows.

In the terminology of Rico-Juan et al., "fork" refers to any subtree of a tree, "root" to a subtree occurring at the top of the tree, and "subtree" to a subtree occurring at the bottom of the tree (that

[1]We call an automaton M_1 more specific than M_2 if the language it defines is a subset of the language defined by M_2.

is, for which all leaves are also leaves of the original tree). From here on we use the term "subtree" in the same sense as Rico-Juan et al. For a k-testable language, the algorithm gathers all roots of height $k-1$ ("$k-1$-roots"), subtrees of height $k-1$ or less, and forks of height k. For instance, given the tree $a(b(c),c)$, with $k=3$ the root $a(b,c)$, fork $a(b(c),c)$ and subtrees $b(c)$ and c are gathered.

For a k-testable automaton, all encountered transitions are derivable from these forks, and all states needed are in the root, subsets, and $k-1$-roots of the forks. To distinguish between trees and states, we will denote the state associated with a tree t as $\overline{t}$. Thus, if we choose $k=3$, then for the example tree above the 3-testable automaton with states $\{\overline{a(b,c)},\overline{b(c)},\overline{c}\}$, transition function $\Delta = \{a(\overline{b(c)},\overline{c})\rightarrow\overline{a(b,c)}, b(\overline{c})\rightarrow\overline{b(c)}, c\rightarrow\overline{c}\}$ and final state $\overline{a(b,c)}$ is derived. It accepts exactly this tree. A formal description of the algorithm, and also of the extensions we are about to discuss next, can be found in Kosala et al. (2002).

When applying this algorithm to information extraction, we identified some problems. One is that the resulting tree automaton, when applied to unseen documents, often was unable to parse the whole tree because certain kinds of transitions were needed that were never observed in the training set. This is in part due to the size of V, which even with the CDATA preprocessing step is still large. Note for instance that the automaton cannot reach an accepting state, hence that the extraction task fails, when the document contains a symbol that does not occur in the training set. HTML trees are often large and may contain parts that are entirely irrelevant to the extraction task. Hence any symbol in such irrelevant part that does not occur in the training set causes failure of the extraction task.

This is a clear case of undergeneralization: the resulting automaton is too specific and too strongly biased towards examples very similar to those in the training set. To improve generalization properties of the induction algorithm, we have developed two extensions to the algorithm. Both consist of generalizing the transition function of the original k-testable automaton, by learning transition rules that cover a set of states instead of a single state. This is done by introducing wildcards instead of specific labels in these transition rules.

For example, where the original algorithm would learn a transition $a(\overline{b(c)},\overline{c})\rightarrow\overline{a(b,c)}$ if and only if this transition occurs at least once in the example dataset, the modified algorithms might generalize this transition into $a(\overline{b(*)},\overline{c})\rightarrow\overline{a(b,c)}$, thus also allowing the automaton to parse $a(b(b(c)),c)$ even if no corresponding forks occurred in the examples.

Stronger generalization occurs if we put wildcards at other than the lowest level. The transition $a(*(*),c)\rightarrow a(*,c)$ indicates that also the label immediately below a can be anything, and the resulting state is in this case not determined. Thus when generalizing at any but the lowest level in this way, the resulting automaton becomes nondeterministic.[2]

Our two extensions differ in the way they introduce these wildcards.

The g-testable algorithm works as follows. Next to the parameter k, which has the same meaning as in the k-testable approach, there is a parameter l. The algorithm distinguishes so-called *context forks*, which contain the marker x, and "other forks", which do not contain the marker. Context forks are kept in the set CF and are not generalized with wildcards. Other forks are stored in the set OF and these are generalized by introducing wildcards for all labels below the level l. The idea behind this is to allow stronger generalization in those parts of the tree that do not contain the field to be extracted, than in those that do.

The gl-testable algorithm performs a more complicated kind of generalization. Instead of replacing all labels below a given level with wildcards, it inserts these wildcards more selectively. For instance, while the g-testable algorithm considers the minimal generalization of a state $a(b(c),b(c))$ to be $a(b(*),b(*))$, the gl-testable algorithm can also consider generalizations $a(b(*),b(c))$ and $a(b(c),b(*))$.

This additional flexibility causes the search for good generalizations to be more complicated. While generalizations for the g-testable algorithm are fully ordered, the generalizations considered in gl-testable are partially ordered and thus form a lattice that can be searched. The algorithm searches for generalizations that do not cover any context forks or subtrees.

More specifically, in a first step it tries to perform a relatively strong generalization over a set of forks: given a subset of forks with the same tree structure, it generalizes them towards a single fork by introducing wildcards in each node that does not have the same label in all forks. This generalization may or may not be succesful (it fails if the generalization covers a context fork). If it succeeds, the result is a single fork; if it does not, the result is the original set of forks. In a second step it cautiously further generalizes the fork(s) resulting from the first step by introducing more wildcards one by one, avoiding in each step to cover context forks.

To illustrate these procedures, assume we have sets of context forks $\{a(b,x)\}$ and other forks $\{a(b,c),a(b,b)\}$. The g-testable algoritm can only generalize the other forks to $\{a(*,*)\}$. The gl-testable algorithm, on the other hand, builds its first generalization by introducing wildcards for those

[2] In fact one could impose that the first and third wildcard in the example be the same, since we assume k-testability, but our notation of transitions does not allow this.

parts of the forks that are not common to all forks; in our case this yields the fork $\{a(b,*)\}$. This fork covers the context fork $a(b,x)$ and is therefore not allowed. The original forks are kept and then generalized towards $\{a(*,b), a(*,c)\}$, none of which cover a context fork.

5 Experimental Results

We evaluated our method on some semi-structured data sets commonly used in the IE research (available from http://www.isi.edu/~muslea/RISE/): a collection of web pages containing people's contact addresses (the Internet Address Finder (IAF) database) and a collection of web pages about stock quotes (the Quote Server (QS) database). We also use a subset of the Shakespeare XML dataset (available from http://www.ibiblio.org/bosak/). For full details of the data sets we refer to Kosala et al. (2002).

We use criteria that are commonly used in the information retrieval research for evaluating our method: precision P (number of correctly extracted objects divided by total number of extractions), recall R (number of correct extractions divided by total number of objects present in the answer template), and $F1$, the harmonic mean of P and R: $2PR/(P+R)$.

Table 1 shows the results we obtained as well as those obtained by some current state-of-the-art string-based methods: an algorithm based on Hidden Markov Models (HMMs) (Freitag and McCallum, 1999), the Stalker wrapper induction algorithm (Muslea et al., 2001) and BWI (Freitag and Kushmerick, 2000). We also include the results of the k-testable algorithm in (Kosala et al., 2002b) and the g-testable algorithm in (Kosala et al., 2002a). The results of HMM, Stalker and BWI are adopted from Freitag and Kushmerick (2000). All tests are performed with 10-fold cross-validation following the splits used by them, except for the small Shakespeare dataset, where we used 2-fold cross-validation.

Table 1 shows the best results of gl-testable with a certain k that were obtained by cross-validation. As can be seen, our tree-based methods perform better in most of the test cases than the existing state-of-the-art string-based methods. The only exception is the field date in the QS dataset where BWI performs better. Compared to the results of k-testable, the gl-testable method performs better in the IAF-alt.name, IAF-organization and small Shakespeare data. Compared to the results of g-testable, gl-testable performs better in the IAF-alt.name and IAF-organization data but worse in the small Shakespeare data.

For the QS data, no difference between the k-testable, g-testable, and gl-testable algorithms was measured, which suggests that adding wildcards does not help in this case. For the small Shakespeare data, the gl-testable algorithm performs worse than the g-testable algorithm, which is somewhat surprising given its greater expressiveness. We suspect this is due to the fact that g-testible optimizes two parameters, not only k but also g.

It is also informative to look at the effect of k on the precision and recall of the induced automata. In Figure 4 these measures are shown for gl-testable. As expected, automata tend to become more specific with increasing k. In several cases an optimal value for k can clearly be identified.

The average training time of the tree-based algorithms varied from 0.5s to 7s in our experiments. Their theoretical training time is $O(km \log m)$ with m the total number of nodes in all example trees, so the algorithms scale well. The average extraction time per document varied from 4 to 14 seconds. The theoretical time complexity of the extraction procedure is $O(n^2)$ with n the number of nodes in the document, so the approach does not scale very well to very large documents (in terms of number of nodes). It is feasible for the type of documents considered here, though. We should also note that our implementation is a Prolog program and was not optimized for performance; we expect it is possible to reduce the extraction time with a considerable constant factor, for instance by indexing the states of the automaton.

6 Conclusions

We have argued for the use of tree automata, as opposed to finite automata, for information extraction from tree-structured documents such as web pages in HTML format. We have described a number of algorithms for learning such tree automata from examples (documents where the fields to be extracted are indicated) and presented experimental results indicating the performance of these algorithms on a few benchmark tasks. The results confirm our expectations that exploiting the tree structure of documents is beneficial to the performance of the information extractor.

Note that although we have discussed tree automata for information extraction, which is indirectly related to information retrieval, tree automata can in principle just as well be used for the document retrieval task itself, since a tree automaton just accepts or rejects trees (documents) as a whole. Such an approach would be useful in cases where the criterion for retrieving documents relies to some extent on their structural properties. We are unaware of any work investigating the use of tree automata for this purpose.

	IAF - alt.name			*IAF - organization*			*QS - date*			*QS - volume*		
	Prec	*Rec*	*F1*	*Prec*	*Rec*	*F1*	*Prec*	*Rec*	*F1*	*Prec*	*Rec*	*F1*
HMM	1.7	90	3.4	16.8	89.7	28.4	36.3	100	53.3	18.4	96.2	30.9
Stalker	100	-	-	48.0	-	-	0	-	-	0	-	-
BWI	90.9	43.5	58.8	77.5	45.9	57.7	100	100	100	100	61.9	76.5
k-testable	100	73.9	85	100	57.9	73.3	100	60.5	75.4	100	73.6	84.8
g-testable	100	73.9	85	100	82.6	90.5	100	60.5	75.4	100	73.6	84.8
gl-testable	100	84.8	91.8	100	84.6	91.7	100	60.5	75.4	100	73.6	84.8

	Small Shakespeare		
	Prec	*Rec*	*F1*
k-testable	56.2	90	69.2
g-testable	69.2	90	78.2
gl-testable	66.7	80	72.7

Table 1: Comparison of the results

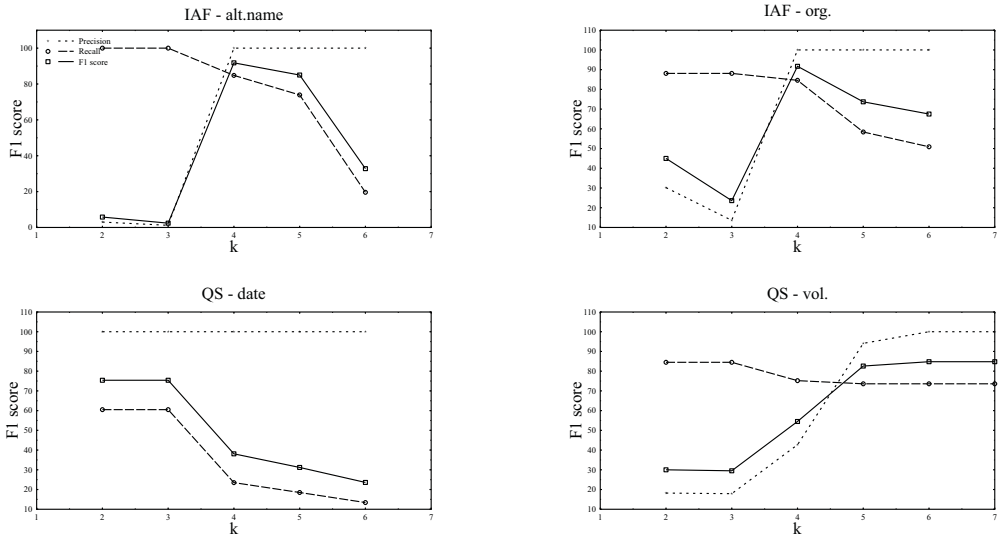


Figure 4: The graphs of F1 score vs k

Acknowledgements

We thank Nicholas Kushmerick for providing us with the datasets used for the BWI experiments. H.B is a postdoctoral fellow of the Fund for Scientific Research of Flanders (FWO-Vlaanderen), which has also supported this work through the project "Query languages for data mining".

References

J. Bear, D. Israel, J. Petit, and D. Martin (1997). Using information extraction to improve document retrieval. In *Proceedings of the Sixth Text Retrieval Conference*, pages 367-378.

D. Freitag and N. Kushmerick (2000). Boosted wrapper induction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Innovative Applications of AI Conference*, pages 577–583. AAAI Press.

D. Freitag and A. McCallum (1999). Information extraction with HMMs and shrinkage. In *AAAI-99 Workshop on Machine Learning for Information Extraction.*

D. Freitag (1997). Using grammatical inference to improve precision in information extraction. In *ICML-97 Workshop on Automata Induction, Grammatical Inference, and Language Acquisition.*

G. Gottlob and K. Koch (2002). Monadic datalog over trees and the expressive power of languages for web information extraction. In *21st ACM Symposium on Principles of Database Systems*, pages 17–28.

R. Kosala, M. Bruynooghe, H. Blockeel, and

J. Van den Bussche (2002a). Information extraction by means of a generalized k-testable tree automata inference algorithm. In *Proceedings of the Fourth International Conference on Information Integration and Web-based Applications & Services (IIWAS)*. To appear.

R. Kosala, J. Van den Bussche, M. Bruynooghe, and H. Blockeel (2002b). Information extraction in structured documents using tree automata induction. In *Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 299–310. Springer.

R. Kosala, M. Bruynooghe, J. Van den Bussche and H. Blockeel (2002). Information extraction from web pages based on k-testable tree automaton inference. Submitted.

S. Lin and J. Ho (2002). Discovering informative content blocks from web documents. In *Proceedings of the Eight ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

K. Murphy (1996). Learning finite automata. Technical Report 96-04-017, Santa Fe Institute.

I. Muslea, S. Minton, and C. Knoblock (2001). Hierarchical wrapper induction for semistructured information sources. *Journal of Autonomous Agents and Multi-Agent Systems*, 4:93–114.

J.R. Rico-Juan, J. Calera-Rubio, and R.C. Carrasco (2000). Probabilistic k-testable tree-languages. In A.L. Oliveira, editor, *Proceedings of 5th International Colloquium, ICGI 2000, Lisbon (Portugal)*, volume 1891 of *Lecture Notes in Computer Science*, pages 221–228. Springer.

Exploiting Structure for Information Retrieval

Jaap Kamps and **Maarten Marx** and **Christof Monz** and **Maarten de Rijke**
Language & Inference Technology Group
University of Amsterdam
Nieuwe Achtergracht 166, 1018 WV Amsterdam
E-mail: {kamps,marx,christof,mdr}@science.uva.nl
http://www.illc.uva.nl/LIT/

Abstract

Structured elements are pervasive in digital libraries, product catalogs, scientific data collections and on the Internet. One of our research aims is to investigate the ways in which the additional structure of a collection can be brought to bear on retrieval effectiveness. This paper reports on our experiments on the use of manually assigned keywords in domain specific collections; on the use of URL and link structure on the Internet; and on the use of XML-structure in annotated scientific collections.

1 Introduction

In 2002, the LIT Group at the University of Amsterdam participated in a number of tasks that contain different types of structural information:

- the usage of (manually assigned) keywords in the scientific collections GIRT and Amaryllis used at CLEF (CLEF, 2002);
- the mark-up, URL and link structure in the .GOV collection used at TREC's Web Track (Web Track, 2002); and
- the XML-structure in the IEEE Computer Society collection used at INEX (INEX, 2002).

These three evaluation exercises are loosely related in that they all go beyond the traditional plain-text collection. In all cases, there is some additional structure available that may help to improve the effectiveness of information retrieval, be it that the type of structure differs greatly between tasks.

The outline of this paper is as follows. First, we'll briefly discuss our experimental set up. Then, in three separate sections, we give a brief overview of our experiences during each of the evaluation campaigns. Finally, we discuss our results and draw some tentative conclusions.For further information on our experiments, we refer to (Monz et al., 2002, 2003; Marx et al., 2002).

2 Experimental Set-up

All experiments were carried out with the FlexIR system developed at the University of Amsterdam (Monz and de Rijke, 2002), using the Lnu.ltc weighting scheme.

CLEF Scientific Collections. The domain-specific collections at CLEF are the GIRT collection of German social science literature, and the Amaryllis collection containing French scientific literature. We built free-text only indexes for the GIRT and Amaryllis collections. For both French and German we used a lexical-based stemmer (Schmid, 1994). For German we applied a compound splitter. All morphological runs use blind feedback. Additionally, we built keyword-only indexes of the manually assigned keywords. The keywords were indexed as given, indexing the keywords or keyword-phrases as a single token. Blind feedback was switched off for keyword runs. For GIRT's English to German bilingual runs, we used the Ding dictionary (Ding, 2002). For Amaryllis' English to French bilingual task, we used the on-line Systran translator (Systran, 2002).

TREC Web Track. This year's collection, aptly named .GOV, is based on a crawl of the .gov Internet domain in early 2002. We built a free-text index of the collection using the Porter stemmer (Porter, 1980). Additionally, we built three different anchor-text only indexes, assigning the anchor texts to the linked documents. We made runs on the text and anchors indexes, using Lnu.ltc and a weighting scheme based on minimal matching spans (Monz et al., 2003). None of our runs used blind feedback.

INEX. The collection for INEX consists of IEEE Computer Society journals and proceedings. We built three free-text indexes: using plain words; using the Porter stemmer (Porter, 1980); and using an ngram approach. We preserved the XML-structure in the inverted index by indexing each tag as a single

token. We made initial retrieval runs, using Lnu.ltc weighting and blind feedback. For the content-and-structure queries, we used an XML-parser to extract the required XML-elements from the initially retrieved set of documents.

3 Exploiting Keyword Structure

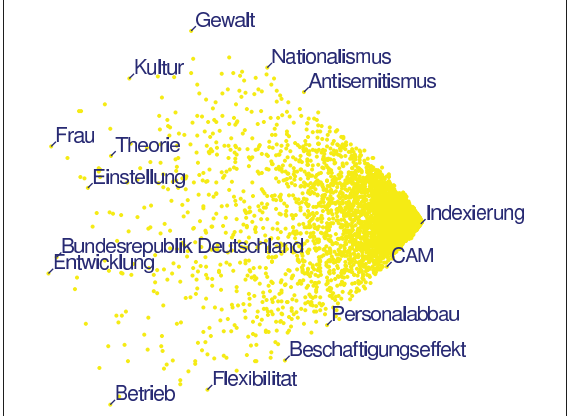
Many domain-specific collections, such as the scientific collections of GIRT and Amaryllis, contain meta-information such as keywords. Special dictionaries or thesauri for the meta-information are not always available. Our strategy for CLEF 2002 was to compute the similarity of keywords based on their occurrence in the collection, and explore whether the resulting keyword space can be used to improve retrieval effectiveness.

The GIRT collection contains $76,128$ documents from German social science literature published between 1978 and 1996 (Kluck and Gey, 2001). The documents are also classified by keywords assigned by human indexers. The average number of keywords in a document is 9.91. A total of $6,745$ different keywords are used in the collection. The Amaryllis collection contains $148,688$ documents in French from various scientific fields. The average number of manually assigned keywords in a document is 10.75. A total of $125,360$ different keywords are used in the collection. We decided to focus on the $10,274$ keywords that occur ≥ 25 times in the collection. We determined the number of occurrences of keywords and pairs of keywords, and used these to define a distance metric (Gower and Legendre, 1986). We reduced the matrix to 10 dimensions using metric multi-dimensional scaling techniques (Cox and Cox, 1994). For all calculations, we used the best approximation of the keyword distance matrix on 10 dimensions (the plots in Figure 1 show the 2 principal dimensions).

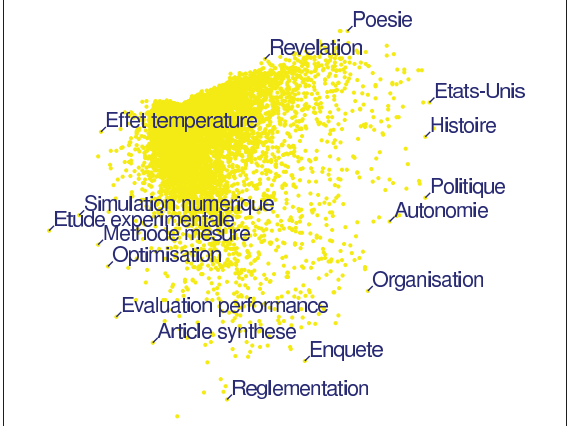
We experimented with the use of the resulting keyword spaces for keyword recovery and document reranking. The resulting keyword space has a 10-dimensional vector for each of the keywords. Vectors for documents and topics are based on the initially retrieved documents from a morphological base run (not using the keywords). For each of these documents, we collect the keywords, and determine a document vector by taking the mean of the keyword vectors. Next, we determine a topic vector by taking the weighted mean of the document vectors for the top 10 documents. We can recover keywords for a topic by selecting, from the keywords used in the top 10 documents, the ten closest to the topic vector. In the monolingual Amaryllis task, the topic authors have assigned keywords for the topics in the narrative field. Table 2 compares these manually assigned keywords to our recovered keywords.

Table 1: Manual Keyword Space.

GIRT 6745 keywords



Amaryllis 10274 keywords



We used the keyword space for recovering keywords, and for document reranking. The recovered keywords are used in keyword-only runs. We created combined runs of the morphological base runs and the keyword-only runs. For document reranking, we simply reranked the documents retrieved in the base run by the distance between the document and topic vectors. The morphological base runs and rerank runs are also used in combined runs. The runs were combined in the following manner. Following Lee (1995), the scores are normalized using $RSV_i' = \frac{RSV_i - min_i}{max_i - min_i}$. We as-

Table 2: Provided versus Recovered Keywords.

Amaryllis topic 001
⟨FR-title⟩ *Impact sur l'environnement des moteurs diesel* ⟨FR-desc⟩ *Pollution de l'air par des gaz d'échappement des moteurs diesel et méthodes de lutte antipollution. Emissions polluantes (NOX, SO2, CO, CO2, imbrûlés, ...) et méthodes de lutte antipollution* ⟨EN-title⟩ The impact of diesel engine on environment ⟨EN-desc⟩ Air pollution by the exhaust of gas from diesel engines and methods of controlling air pollution. Pollutant emissions (NOX, SO2, CO, CO2, unburned product, ...) and air pollution control
Provided keywords
Concentration et toxicité des polluants *Mécanisme de formation des polluants* *Réduction de la pollution* *Choix du carburant* *Réglage de la combustion* *Traitement des gaz d'échappement* *Législation et réglementation*
Recovered keywords
Moteur diesel *Qualité air* *Azote oxyde* *Norme ISO* *Produit pétrolier* *Lutte antipollution air* *Véhicule à moteur* *Gas oil* *Consommation carburant* *Carburant*

signed new weights to the documents using the summation function used by Vogt and Cottrell (1998): $RSV_{new} = \lambda \cdot RSV_1 + (1-\lambda) \cdot RSV_2$. All combination of the morphological base run with a rerank run use interpolation factor 0.6, all combination with a keyword-only run use factor 0.7. These factors were obtained from pre-submission experiments on the GIRT collection.

Results. Table 3 lists our non-interpolated average precision scores for the morphological base runs, and for the best combined runs. The figure in brackets indicates the improvement over the best underlying run. Results for the GIRT monolingual morphological run are disappointing (German monolingual 0.4476, GIRT01 0.3083, GIRT00 0.3145). The GIRT bilingual runs score even worse; the morphological base run has only 1.4 relevant documents in the top 10. This explains the decrease in performance for the run combined with a keyword-only run.

Table 3: NI average precision scores.

GIRT monolingual	
Morphological	0.1639
Morph/Keyword	0.1687 $(+2.9\%)$
Morph/Rerank	0.1906 $(+16.3\%)$
GIRT bilingual	
Morphological	0.0666
Morph/Keyword	0.0620 (-6.9%)
Morph/Rerank	0.0704 $(+5.7\%)$
Amaryllis monolingual	
Morphological	0.2681
Morph/Keyword provided	0.3401 $(+26.7\%)$
Morph/Keyword recovered	0.2923 $(+9.0\%)$
Morph/Rerank	0.2796 $(+4.3\%)$
Amaryllis bilingual	
Morphological	0.2325
Morph/Keyword	0.2660 $(+14.4\%)$
Morph/Rerank	0.2537 $(+9.1\%)$

For the monolingual Amaryllis task, the provided keywords score remarkably well (keyword-only run 0.2684), the recovered keywords score 0.1120. In combination, both improve the morphological base run: the combination with recovered keywords scores 0.2923 $(+9.0\%)$; and with provided keywords scores 0.3401 $(+26.7\%)$. The chosen combination factors were generally close to the optimal values for recovered keywords and rerank runs. They proved far from optimal for the provided keywords in monolingual Amaryllis; with 0.4 the combination scores 0.4175 $(+55.6\%)$.

4 Exploiting Link Structure

TREC's Web Track featured two tasks: named-page finding and topic distillation. For the text index, we indexed all of the documents' textual contents, decoding special html-characters into plain ASCII, and replacing diacritics with the unmarked characters. The resulting plain-text index covers 1.25 million documents. Arguably, pages that do not receive links from other sites will rarely be key resources. This motivated experiments with anchor-text only runs on three different indexes:

1. Only extracting complete link descriptions in the collection, which includes all links between pages on different sites. All unique anchor-texts are assigned to the document to which the link points. We remove repeated occurrences

of the same anchor-text. The resulting index covers only 15% of the collection.

2. Here we try to recover as many links as possible, including links within a site. We again remove repeated occurrences of the same anchor-texts. The resulting index covers 54% of the collection.

3. We use the same procedure as for the second anchors index, but now retain repeated occurrences, similar to (Craswell et al., 2001).

For the named-page finding task, we experimented with plain text runs, anchor-text runs, and their combinations. The text and anchor-only runs were combined in the following manner: We only consider the first ten results of both runs. The scores are normalized, and we assign new weights to the documents using the summation function used by Fox and Shaw (1994): $RSV_{new} = RSV_1 + RSV_2$.

We performed extensive experiments with link and URL structure for topic distillation. For topic distillation, only the best documents in the collection will be regarded as relevant. We experimented with the following approach for exploiting the URL information: Since there will rarely be more than one key resource per site, we cluster pages by their base URL, and return the page with the lowest URL depth. Specifically, we assign the top 100 documents to the first 10 different base URLs. Next, we return the page with the lowest URL depth or slash-count per cluster.

We also experimented with the use of the link structure of the documents. There are two established ways of exploiting link structure: page-rank (Brin and Page, 1998) uses the global link structure; Hyperlink Induced Topic Search (HITS) (Kleinberg, 1999) uses the local link structure surrounding an initially retrieved set of documents. We implemented an approach that combines both global and local link structure by comparing how much of the links of a page are present in the local set of initially retrieved documents.

We carried out pre-submission experiments using Kleinberg (1999)'s HITS for the topic distillation task. Table 4 shows the top 10 authorities over the top 100, top 200, and top 500 initially retrieved documents for the test topic '*obesity in the U.S.*'. HITS is successful at isolating key resources, but shows considerable topic drift toward generally good 'authorities.' A loosely-related authority can easily infiltrate due to the correlation between authorities and pages with many inlinks (Kleinberg, 1999; Amento et al., 2000). Our link-based method tries to avoid such topic drift. A general good authority may have many links in the local set, but the proportion of inlinks that is in the local set of documents will remain low. The top 10 results are also shown in Table 4 as 'Realized Indegree Top 100/200/500.' Informal evaluation shows that our combined approach is more robust than HITS: when considering the top 500 initially retrieved documents HITS authorities are unrelated to the topics, whereas the 'realized indegree' method remains on topic.

Results. The official run results are shown in Table 5: column 'MRR' lists the mean reciprocal rank of the first correct answer (the official measure); column 'Top 10' lists the number of topics with at least one correct named page in the top 10; and column 'Unknown' lists the number of topics for which no named page was found in the top 50. The combined

Table 5: Official named page finding run results.

Run	MRR	Top 10	Unknown
Text-only	0.4254	82 (54.7%)	46 (30.7%)
Anchors	0.3279	69 (46.0%)	70 (46.7%)
Combined	**0.4317**	99 (66.0%)	35 (23.3%)

text and anchor run performed the best with a MRR of 0.4317. The anchor-text only run, only indexing half the documents, scores 77.08% of the text only run. The combination of both runs improves the MRR by 1.48% over the text only run, the number of topics in the top 10 is improved by 20.73% over the text only run.

For the topic distillation task, we made runs on the text-only and anchors-only collections. Furthermore, we experimented with approaches to exploiting the URL information and link structure of the documents. The results of our official runs are

Table 6: Official topic distillation run results.

Run	Prec. at 10, 20, and 30		
1. Text-only	**0.1755**	0.1245	0.1020
2. Realized indegree 1	0.0673	0.0582	0.0463
3. Anchors	0.1000	0.0714	0.0558
4. Realized indegree 3	0.0633	0.0469	0.0381
5. base URL clusters 3	0.0653	0.0786	0.0660

shown in Table 6. The official measure is precision at 10, at which the text-only run scores best with 0.1755. The anchor-text only run scores 56.98% of the text only run. A text-only run using Lnu.ltc weighting, not submitted, scored better than the official run with a precision at 10 of 0.2102. The run

Table 4: Test topic "obesity in the U.S."

HITS Top 100	Realized Indegree Top 100
www.nih.gov/icd/od/foia/	www.niddk.nih.gov/health/nutrit/pubs/unders.htm
www.nlm.nih.gov/	www.nlm.nih.gov/medlineplus/obesity.html
www.nlm.nih.gov/medlineplus/obesity.html	hin.nhlbi.nih.gov/bmi_palm.htm
www.nlm.nih.gov/accessibility.html	www.ahcpr.gov/research/may00/0500RA6.htm
www.nlm.nih.gov/contacts/	www.nhlbi.nih.gov/guidelines/obesity/bmi_tbl.htm
www.nlm.nih.gov/disclaimer.html	www.nlm.nih.gov/medlineplus/diabetes.html
www.nichd.nih.gov/	www.fitness.gov/Reading_Room/reading_room.html
www.nlm.nih.gov/medlineplus/diabetes.html	www.cdc.gov/nccdphp/dnpa/dnpalink.htm
www.nlm.nih.gov/medlineplus/highbloodpressure.h	response.restoration.noaa.gov/photos/dispers/
www.nlm.nih.gov/medlineplus/sleepdisorders.html	www.fda.gov/bbs/topics/NEWS/NEW00575.html
HITS Top 200	**Realized Indegree Top 200**
www.nih.gov/icd/od/foia/	www.nhlbi.nih.gov/health/public/heart/obesity/
www.nlm.nih.gov/	www.nlm.nih.gov/medlineplus/obesity.html
www.nlm.nih.gov/medlineplus/obesity.html	hin.nhlbi.nih.gov/bmi_palm.htm
www.nichd.nih.gov/	www.niddk.nih.gov/health/nutrit/pubs/unders.htm
www.nlm.nih.gov/disclaimer.html	www.ftc.gov/bcp/conline/pubs/health/setgoals.htm
www.nlm.nih.gov/accessibility.html	www.cdc.gov/nccdphp/dnpa/
www.nlm.nih.gov/contacts/	www.cdc.gov/health/obesity.htm
www.nlm.nih.gov/medlineplus/diabetes.html	whi.nih.gov/health/prof/heart/
www.nlm.nih.gov/medlineplus/highbloodpressure.h	www.ahcpr.gov/research/may00/0500RA6.htm
www.nlm.nih.gov/medlineplus/respiratorydiseases	www.ftc.gov/bcp/conline/pubs/health/setgoals.pdf
HITS Top 500	**Realized Indegree Top 500**
www.disability.gov/	www.nhlbi.nih.gov/health/public/heart/obesity/
www.nhlbi.nih.gov/health/public/heart/obesity/	whi.nih.gov/health/public/heart/
www.business.gov/	www.niddk.nih.gov/health/nutrit/win.htm
www.seniors.gov/	hin.nhlbi.nih.gov/bmi_palm.htm
www.tradenet.gov/	www.niddk.nih.gov/health/nutrit/pubs/binge.htm
www.workers.gov/	www.nlm.nih.gov/medlineplus/obesity.html
www.students.gov/	www.niddk.nih.gov/health/nutrit/pubs/unders.htm
www.seniors.gov/	www.niddk.nih.gov/health/diabetes/pubs/afam/
www.npr.gov/	www.cdc.gov/health/obesity.htm
www.cio.gov/	www.healthfinder.gov/news/

using the base-URL clusters fails to improve the anchor text base run, although it improves precision at 20 and 30. The runs based on link information all perform worse than the underlying base runs.

Table 7: Anchors only run results.

Run	Index	MRR	Prec. at 10
Named page	Anchors 1	0.1391	
Named page	Anchors 2	**0.3279**	
Named page	Anchors 3	0.3098	
Distillation	Anchors 1		0.0673
Distillation	Anchors 2		**0.1000**
Distillation	Anchors 3		0.0837

The post-submission experiments shown in Table 7 show the performance of anchor-text only runs using the three anchor-text indexes as described above. The second anchor-text index, which was used for our official runs, shows the best performance. This index contains unique occurrences of links between and within sites.

5 Exploiting XML Structure

The INEX collection, 21 IEEE Computer Society journals from 1995–2002, consists of 12,135 documents with extensive XML-markup. The INEX initiative for the evaluation for XML retrieval featured two types of topics: traditional content-only topics, and content-and-structure topics. Our aims at INEX were to set up a baseline system on which we plan to build in future editions of this task. Some of our more ambitious plans failed to be realized due to the inconvenience of crashing XML-parsers, or the inability to produce the required Xpath-location. Our baseline system uses a two-stage strategy. In the first stage, we use the content words in the query to retrieve an initial set of documents. In the second stage, we subject this set of potentially relevant documents to greater scrutiny. In particular, for the content-and-structure queries, we used an XML-parser to extract the required XML-elements from the initially retrieved set of documents.

Our official runs experiment with the effectiveness of different types of morphological normalization for structured corpora. Morphological normalization proved successful for plain text collections (Monz and de Rijke, 2002; Monz et al., 2002). The XML retrieval tasks departs from the strict boolean query matching used in traditional database theory,

allowing for various gradations of relevance. In particular, related words like morphological variants should share some of their relevance. In order to study the precise effect of morphological normalization, we created plain-word, stemmed, and ngrammed indexes that preserve the XML-structure of the original documents. This allows for both the content-only and content-and-structure topics to be evaluated against all three indexes. Informal evaluation shows that morphological normalization helps to retrieve relevant documents missed out by the plain text run. At the time of writing, relevance assessment for INEX is still in progress.

6 Discussion and Conclusions

The three sets of experiments described in this paper, are loosely connected in that they all go beyond the traditional plain-text collection. In all our experiments some additional structure is brought to bear on the information retrieval task, be it that the type of structure differs greatly between tasks. Still, it is worth to discuss some points of agreement between the experiments. There are similarities in the used techniques, the HITS approach uses the same multi-dimensional scaling techniques we applied to the keyword space. MDS techniques give the best approximation of a high-dimensional space in a small number of dimensions. HITS authorities and hubs are based on the principal dimension only, whereas we focus on approximations on ten dimensions.

Both the `.GOV` collection used at Web Track and the IEEE Computer Society collection used at INEX have extensive mark-up in HTML and XML, respectively. Assigning different weights of importance to words occurring in specific tags (such as bold-faced words of headings) can be effective for improving retrieval effectiveness (Cutler et al., 1997). We did not apply this technique yet, since estimating the relative weights of different tags requires a set of test topics. These were not yet available, because both the `.GOV` and IEEE Computer Society collections are used for the first time in 2002. Using the sets of test topics of this year's evaluation, we plan to look at this in more detail.

Most of the journals in the IEEE Computer Society collection have keywords assigned to the documents. Thus, the same techniques we used on the GIRT and Amaryllis collection, i.e., keyword recovery and reranking documents, can be directly applied to the XML retrieval task. Since it was unclear whether these techniques were relevant for the particular type of relevance judgments used at INEX, we did not implement this for our official runs. Again, we plan to address this in future research when the evaluation sets for XML retrieval come available.

It is not the case that using some additional structure will always help to improve the retrieval effectiveness over a highly sophisticated plain-text base run. Our experiments with link-based methods for Web Track's topic distillation task show a decrease in precision at 10. This is in line with earlier attempts at exploiting link structure in the ad hoc task (Hawking and Craswell, 2002). A possible explanation could be the topics used for the distillation task. These are more specific than the very general topics used in Kleinberg (1999), such as '*java*,' '*censorship*,' '*search engines*,' and '*Gates*.' Also, after stopping, the test topic '*obesity in the U.S.*' results in the one-word query '*obesity*.' For such general queries, relevant documents will dominate the top 10, top 100, or even top 200 of initially retrieved documents. Under this assumption, link-based approaches, which ignore the content of documents and solely consider the link topology, can be effective. If non-relevant documents dominate the initially retrieved set of documents, one cannot expect link-based methods to deliver. For the named page finding task, a genuine needle-in-a-haystack task, we experimented with text-only and anchor-text only runs, and their combinations. Here, the combined text/anchor-text run slightly improves the mean reciprocal rank, but significantly improves the number of topics with the named page in the top 10.

The experience on CLEF's scientific collections is that recovered keywords and reranking runs score worse than the morphological base runs. The lower performance of the keyword-only runs is no surprise considering the lack of information contained in the documents' textual parts. The lower performance of the reranking runs is probably due to the unsophisticated reranking strategy that, for example, does not take keyword frequency into account. Having said that, the combined runs with keywords and reranking show a significant improvement of retrieval effectiveness. It is interesting to note that for the GIRT task the combined reranking runs outperform the combined keyword runs, whereas for the Amaryllis task, the combined keyword runs outperform the combined reranking runs. This may be due to the difference in the numbers of keywords used to characterize the documents, which is much

more fine-grained in the case of Amaryllis. The fact that the combined runs significantly improve over the best underlying base runs gives us some confidence in the effectiveness of our approach. It shows that extracting the meaning of keywords from their usage in the collection itself can be a viable alternative for manually constructed, domain-dependent dictionaries and thesauri. Additionally, the keyword space can be useful for providing visualizations of keywords, documents, and topics (Hearst, 1999).

Acknowledgments

The paper benefitted greatly from the comments of the anonymous reviewers. We want to thank Willem van Hage and Vera Hollink for their technical support. Jaap Kamps was supported by the Netherlands Organization for Scientific Research (NWO), grant # 400-20-036. Maarten Marx received support from NWO grant 612.000.106. Christof Monz was supported by the Physical Sciences Council with financial support from NWO, project 612-13-001. Maarten de Rijke was supported by grants from NWO, under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 220-80-001, and 612.000.207.

References

Amento, B., L. Terveen, and W. Hill (2000). Does 'authority' mean quality? predicting expert quality ratings of web documents. In E. Yannakoudakis, N. J. Belkin, M.-K. Leong, and P. Ingwersen (Eds.), *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 296–303. ACM Press, New York NY, USA.

Brin, S. and L. Page (1998). The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference*, pp. 107–117. Elsevier Science, New York.

CLEF (2002). Cross language evaluation forum. `http://www.clef-campaign.org/`.

Cox, T. F. and M. A. A. Cox (1994). *Multidimensional Scaling*. Chapman & Hall, London UK.

Craswell, N., D. Hawking, and S. Robertson (2001). Effective site finding using link anchor information. In D. H. Kraft, W. B. Croft, D. J. Harper, and J. Zobel (Eds.), *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 250–257. ACM Press, New York NY, USA.

Cutler, M., Y. Shih, and W. Meng (1997). Using the structure of html documents to improve retrieval. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*.

Ding (2002). Ding: A dictionary lookup program. `http://dict.tu-chemnitz.de/`.

Fox, E. A. and J. A. Shaw (1994). Combination of multiple searches. In D. K. Harman (Ed.), *The Second Text Retrieval Conference (TREC-2)*, pp. 243–252. National Institute for Standards and Technology. NIST Special Publication 500-215.

Gower, J. C. and P. Legendre (1986). Metric and euclidean properties of dissimilarity coefficients. *Journal of Classification 3*, 5–48.

Hawking, D. and N. Craswell (2002). Overview of the TREC-2001 web track. In E. M. Voorhees and D. K. Harman (Eds.), *The Tenth Text Retrieval Conference (TREC 2001)*, pp. 25–31. National Institute for Standards and Technology. NIST Special Publication 500-250.

Hearst, M. A. (1999). User interfaces and visualization. In *Modern Information Retrieval*, Chapter 10, pp. 257–324. ACM Press, New York and Addison Wesley Longman, Harlow.

INEX (2002). Initiative for the evaluation of XML retrieval. `http://qmir.dcs.qmw.ac.uk/INEX/`.

Kleinberg, J. M. (1999). Authoritative structures in a hyperlinked environment. *Journal of the ACM 46*, 604–632.

Kluck, M. and F. C. Gey (2001). The domain-specific task of CLEF - specific evaluation strategies in cross-language information retrieval. In C. Peters (Ed.), *Cross-Language Information Retrieval and Evaluation, CLEF 2000*, Volume 2069 of *Lecture Notes in Computer Science*, pp. 48–56. Springer.

Lee, J. H. (1995). Combining multiple evidence from different properties of weighting schemes. In E. A. Fox, P. Ingwersen, and R. Fidel (Eds.), *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 180–188. ACM Press, New York NY, USA.

Marx, M., J. Kamps, and M. de Rijke (2002). The University of Amsterdam at INEX-2002. In N. Fuhr, N. Gövert, G. Kazai, and M. Lalmas (Eds.), *INEX: Initiative for the Evaluation of XML retrieval*.

Monz, C. and M. de Rijke (2002). Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck (Eds.), *Evaluation of Cross-Language Information Retrieval Systems, CLEF 2001*, Volume 2406 of *Lecture Notes in Computer Science*, pp. 262–277. Springer.

Monz, C., J. Kamps, and M. de Rijke (2002). The University of Amsterdam at CLEF-2002. In C. Peters (Ed.), *Results of the CLEF 2002 Cross-Language System Evaluation Campaign*, pp. 73–84.

Monz, C., J. Kamps, and M. de Rijke (2003). The University of Amsterdam at TREC 2002. In E. M. Voorhees and D. K. Harman (Eds.), *The Eleventh Text Retrieval Conference (TREC 2002)*. National Institute for Standards and Technology.

Porter, M. (1980). An algorithm for suffix stripping. *Program 14*(3), 130–137.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*.

Systran (2002). Systran Online Translator. `http://www.systransoft.com/`.

Vogt, C. C. and G. W. Cottrell (1998). Predicting the performance of linearly combined ir systems. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel (Eds.), *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 190–196. ACM Press, New York NY, USA.

Web Track (2002). Web track at TREC. `http://www.ted.cmis.csiro.au/TRECWeb/`.

A scalable and efficient content-based multimedia retrieval system

Lioudmila Boldareva, Djoerd Hiemstra, Willem Jonker
Database Group, University of Twente, The Netherlands,
{*L.V.Boldareva, D.Hiemstra W.Jonker*}@*cs.utwente.nl*

Abstract

In this work the problem of content-based information retrieval is approached from a new perspective. We look at a probabilistic approach in CBIR from the angle of Bayesian networks. Our data structure serves to break two bottlenecks of retrieval performance: (1) high dimensionality of feature vectors and (2) poor mapping of raw features into high-level content that a human understands (the semantic gap). We use the network structure instead of the feature space, and propose updating the higher-level content description by utilising the relevance feedback obtained from the user. Strategies for display update for the next iteration are studied. A new approach for selecting the next display set is tied to our data structure.

1 Introduction

In content-based information retrieval, there is a problem of the gap between human perception of the document, which is often referred to as high-level content and its actual representation at the lowest level, in the data storage.

This problem is addressed by indexing documents in the collection. Text documents are indexed based on the words they contain; for images and video, pictorial features such as colours, shapes, textures, motion detection are used. Careful selection of the feature set allows capturing semantics of the documents, especially in limited domains, where the range of possible values is pre-determined. Examples of such limited domains are texture catalogues, medical image databases, video archives of known context. Often the number of features automatically extracted from raw data is large, with the hope that it helps to capture the semantics better. Modern retrieval systems have rich feature space: in MARS (Rui et al., 1997b) the vector space is of at least several dozens dimensions, in PicHunter (Cox et al., 2000) the number of features is about 128, Viper (Müller et al., 2000b) boasts $O(80\ 000)$ values. The objects are represented as points in metric space, with the dimensionality corresponding to the number of extracted features. The similarity between them is determined by means of an appropriate metric, such as Euclidean distance.

A lot of work is done to determine "the best feature set", i.e. such representation of multimedia data that would match the human perception of it. This is not a trivial task for images and video, since visual data carries a lot of information which is hard to decode automatically. However, both automatic and manual feature selection might still not solve the problem. The user is not always certain what pictorial characteristics are important for his/her target, and the semantics of an object may be ambiguous.

At the same time, due to the large number of dimensions, the pictorial features are subject to "the curse of dimensionality", when the performance of indices drops dramatically as the number of dimensions grows. Effective multidimensional indexing and approximate retrieval based on indexing are active research topics, as well as the problem of (weighted) nearest neighbour search in the indexed space (Faloutsos and Lin, 1995; Wu and Manjunath, 2001; de Vries et al., 2002). The situation turns tricky: on one hand we have large number of features that are hard to index; on the other hand, the importance of a certain feature is unknown.

A significant improvement of the performance of content-based retrieval systems can be achieved by using *relevance feedback*, a technique that allows the user to rate the (intermediary) search results. Further ranking and retrieval of documents in the collection is based on the feedback received from the user. In the domain of image retrieval, where the semantic gap is especially large, relevance feedback is often used not only for ranking the output documents, but also for fine-tuning the whole system, adapting such parameters as similarity function (Rui et al., 1997a; Wu and Manjunath, 2001; Aksoy and Haralick, 2000; Geman and Moquet, 1999), and/or the feature set that is used. In the 2-layer retrieval model in MARS (Rui et al., 1997a), the useful feature subset is determined based on the user response, when features are examined across iterations and most distinguishing ones get more weight in subsequent iterations. In the Qbic image retrieval system

(Flickner et al., 1995) the user can manually emphasize the importance of a certain primitive feature by using "control knobs". At a higher level of representation, dealing with identified colour-texture-shape objects, or "Blobs" in Blobworld (Carson et al., 1999), the user can explicitly point the region of interest, modelled by pre-computed Gaussian mixtures, that incorporate colour and texture information.

In the present paper we approach the problem of content-based image retrieval and indexing from another, new perspective. We look at a *probabilistic approach* to document indexing and retrieval, from the angle of Bayesian networks. Our data structure serves to break two bottlenecks of content-based multimedia retrieval performance: (1) high dimensionality of feature vectors, preventing efficient indexing and (2) ineffective mapping of raw features into higher-level concepts reflecting the actual content. We propose using the network structure for the data instead of multidimensional feature space. The network encodes higher-level context and makes use of relevance feedback. Primitive pictorial features are not addressed at run-time to determine the similarity of objects, but instead, a probabilistic method is used at indexing time to construct the meta-data. We will also study various approaches to retrieval with relevance feedback in the light of our data structure.

The rest of the paper is organised as follows: In section 2 we briefly describe our novel approach, section 3 discusses some issues that arise in the retrieval process, the implementation of the system is described in section 4. Finally, sections 5-6 report preliminary experimental results and future research directions.

2 Bayesian retrieval framework

2.1 Data organization

Consider a collection $\Im$ of objects i among which there is an object that the user is looking for — the target T. In the search session the user retrieves a set of candidate objects on the screen and feeds back to the system his/her opinion about their relevance to the target. Each object might look like the target the user has in mind, and then it is *selected* by the user, or it is *de-selected*, if it doesn't resemble the target. For the selected candidate object i we denote the event as $(\delta_i = 1)$, and for de-selected ones as $(\delta_i = 0)$. The feedback obtained from the user allows the system to make some inference and compile a new display set of n elements, *the display set*, to show in the next iteration. There may be several rounds of feedback during one search session.

To perform comparison of relevant and non-relevant objects, it is necessary to organize the collection by introducing relations between the objects. As mentioned in the introduction, the relations are often defined by a vector space derived from primitive features and a corresponding similarity function(s). In our work we suggest that the user feedback can be used to build up and update the objects relations.

As in (Maron and Kuhns, 1960), we introduce a "measure of closeness" of an object i to an object j as a conditional probability and denote it as $P(\delta_i|T = j)$.

Def. 1 $P(\delta_i|T = j)$ is the probability of an object i being selected by the user given that another object j is the target of the search.

In other words, the user's judgement about the relevance of objects is a necessary component of our system. It is reasonable to assume that $P(\delta_i|T = i) \equiv 1$, i.e. the user always identifies the target as relevant. We also put a constraint that the target exists in the collection and is unique: $P(T = j|T = i) \equiv 0$, $i \neq j$, $\sum_{i\in\Im} P(T = i) = 1$.

As an example take a user looking for an image of a cherry tree. He/she may find an image of an apple tree relevant to his/her query, and the measure of this relevance is denoted as $P(\delta_{apple\,tree}|T = cherry\,tree)$. An image of a cherry fruit may be found relevant too, having another value of $P(\delta_{cherry}|T = cherry\,tree)$.

Each $P(\delta_i|T = j)$ can be seen as a *weighted arc*, or an oriented path of weight P from j to i that is traversed during the search session, utilising the user's feedback. The graph containing these arc values can be seen as a map of oriented paths between elements in the collection. We call its matrix representation *topographic*. The topographic matix need not be symmetric. Imagine a road map, where two cities are connected by roads with different number of lanes going in each direction. It is important to notice, that we deliberately do not construct the complete graph, with all existing connections between the nodes, but choose only the most significant ones.

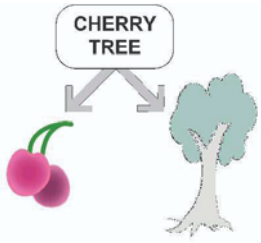


Figure 1: Graphic representation of cherry tree example

Looking at the structure of the graph, each element in the collection can be *described* by a number of other elements pointed by it, which, in

turn, are pointed by third elements. The cherry tree in our small example is described by the set {*apple tree*, *cherry*}, which drives at a tree with a cherry – the cherry tree (see Fig. 1). These associations that come from users judgements and refer to the hidden semantics of objects, serve as meta-data for the collection. The collection describes itself by means of meaningful relations observed in earlier retrieval sessions.

To initiate the system, these relations are calculated as conditional probabilities $P(\delta_i|T=j)$ based on low-level primitive features, which are silently present in the system, but are addressed only once. The subsequent successful searches are used to accumulate the knowledge about objects relations and update the arc weights in the topographic matrix. In this way, instead of chasing "the right feature set" we leave this task to the users, believing that vox populi will give us this best feature set enclosed in a black box.

With the graph representation that refers to a multimedia object as a whole, when a primitive stand-alone feature does not explicitly play an important role, the nodes in the topographic matrix need not be images only. Other types of media, such as video, audio or speech transcripts can be plugged in as separate nodes in the graph. Note however, that integrating other types of media is not trivial. Our data structure relies on multiple feedback iterations. Dynamic media such as video or audio may not stand many feedback loops, because assessing a video clip or a music fragment requires from the user more efforts and time compared to still images. Nevertheless, such nodes may be potential targets or, conversely, the starting points in a search session. Textual nodes are of particular interest for a retrieval system, since querying in the form of text is very convenient for the user.

2.2 Retrieval during the search session

We assume that the user is consistent in his/her judgements, does not forget what the target is, and that the target object is unique and exists in the collection. The assumption of uniqueness is valid with queries like "find me an image of a Golden Retriever puppy". Queries like "find me all pictures of Britney Spears" are not handled by the model directly. However, there is a way to retrieve ranked lists of "most relevant" objects which may be considered targets. In our framework we use the following definition of the target:

Def. 2 *The target* as an object, after retrieval of which the user terminates the search successfully.

The goal of a retrieval system is to help the user find the target object (and possibly all similar objects) after few iterations, with a reasonably small amount of time spent on each round.

Probabilistic methods in information retrieval were initially used for text collections (Maron and Kuhns, 1960; Robertson, 1977; Hiemstra, 2000) and later the ideas were adapted to image retrieval (Vasconcelos and Lippman, 2000). In content-based retrieval systems the user's information need is unknown and should be guessed. In general, retrieval with the use of relevance feedback can be formulated as follows:

> In the current data structure, having observed the user judgements in the search process, what is the object that the user wants to find?

We construct the answer (that is, predict the user's target object) using Bayes' rule. Then the problem is reformulated as estimating the user's action of selecting/deselecting relevant objects, given the target that he/she has in mind:

$$P(T=i,U|\delta^1_{(\cdot)},\ldots,\delta^n_{(\cdot)}) = \frac{P(\delta^1_{(\cdot)},\ldots,\delta^n_{(\cdot)}|T=i)P(T=i|U)P(U)}{P(\delta^1_{(\cdot)},\ldots,\delta^n_{(\cdot)})} \quad (1)$$

where U denotes the current user. Since we assume that the state of the (unknown) user variable does not change during one search session, and U affects $\delta_{(\cdot)}$ through T, we may omit the user notation in further formulae, to keep the notation short (Gelman et al., 1995, Chapter 5). The upper index in $\delta^1_{(\cdot)}\ldots\delta^n_{(\cdot)}$ denotes n displayed objects, either selected by the user ($\delta^i_{(\cdot)}=1$) or not ($\delta^i_{(\cdot)}=0$); $P(T=i)$ is the probability that the object i is the target, and $P(\delta^k|T=i)$ is the probability of a k-the object on the screen to be selected by the user given that i is his/her target.

We distinguish between the objects that have been displayed to the user $\delta^{1\ldots n}_{(\cdot)}$ and the rest of the collection $\{\Im\}$. The index of the element displayed on the screen determines uniquely an element from the collection, and further we omit the subscript, if the upper index is used. Note that equation (1) is regarded as recursive, i.e. the posterior probability of being the target determined at step s as $P(T=i|\delta^1,\ldots,\delta^n)$ serves as the prior $P(T=i)$ at the next iteration $s+1$ (Gelman et al., 1995, Chapter 2).

In this way, in each round the observed user response is used to calculate probability $P(T=i)$. In the beginning, before any information from the user is received, each object has a certain prior probability to be the target[1]. The possible output of in-

[1]Often equal prior probabilities are assigned to all elements of the collection. The importance of selecting the "good" priors is studied in, e.g. (Kraaij et al., 2002).

corporated primary textual query or previous search sessions results may be used to define the prior value of $P(T = i)$ more accurately. Recall that by definition 2, $P(T = i)$ is the probability that the search will be completed successfully immediately after object i is shown to the user.

The meaning of the first term in the numerator of equation (1) is explained by the following definition:

Def. 3 $P(\delta^1, \ldots, \delta^n | T = i)$ is the probability that the user marks the displayed set of objects in a certain way, given that i is the target for him/her.

To determine this joint conditional probability we assume (for the time being) that given the target, the user picks each of n candidates independently of other objects present on the screen. This assumption is similar to term independence assumption used in text retrieval. Then equation (1) becomes

$$P(T = i|\delta^1, \ldots, \delta^n) = \frac{\prod_{s=1}^{n} P(\delta^s|T = i)P(T = i)}{P(\delta^1, \ldots, \delta^n)}. \quad (2)$$

The denominator serves as normalizing factor, and, for the purpose of ranking, it can be replaced with a constant.

2.3 Retrieval in terms of Bayesian network

The search algorithm together with the topographic matrix can be graphically represented as a Bayesian Network (Fig. 2). The conditional probabilities $P(\delta_i|T = j)$ indicate the influence (or *casual impact*) of the fact that a user regards a certain element i of the collection as "somewhat related" to the target j in a search session. The events observed during an iteration are the values of random variables of the network. These values, or *states* are as follows: the node representing a user U looking for an object T which can be any single element of the collection ($T \ni \{1, 2, \ldots, n\}$), and binary variables $\delta_i, i \in \Im$, representing the elements that are judged by the user. They may be marked as similar to the target T, ($\delta_i = 1$), or not ($\delta_i = 0$). Each state is associated with a certainty.

For each user U we may introduce a different prior $P(T|U)$. Every element, and only one in the collection might be the target for a certain user in a given search session. Thus the target can be identified by the maximum probability of $P(T = j|U)$. In our network we talk about *the user model*, since the concept of "target object" has meaning only with respect to a particular user who wants to find the object. However, the user him-/herself is not part of the data structure, and needs to be separated from it.

When the user selects a displayed object as relevant to his/her target ($\delta_2 = 1$ in Fig. 2), he/she is

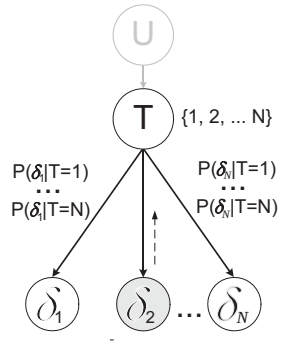


Figure 2: Bayesian Network representation.

doing reasoning in the direction opposite to the casual arrows (the dashed line). Thus the certainty of T changes, which in its turn creates new certainties of not-yet displayed elements δ_1 up to δ_N. Thus, when nothing is known about the state of T and evidence is received at δ_2, nodes $\delta_1 \ldots \delta_N$ are dependent, which means that information on either event affects the certainty of the other, in accordance with the tables attached to the casual arrows. These tables are in fact columns from the topographic matrix. However, when the state of T is known for certain, then its children are independent: information on one has no effect on another. When the retrieval system is at work we repeatedly get new cases, and we learn from these cases. It is a general practice in retrieval systems to discard the results of this learning after the search is concluded, since the user node remains unknown. Because the system is re-initiated for every new query, this method is called *short-term learning*.

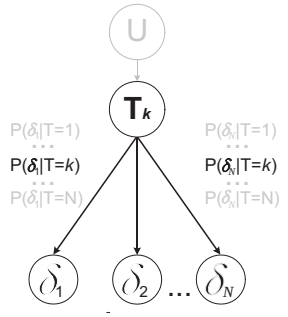


Figure 3: The network structure after the target is identified. The values from the topographic matrix subject to update are shown in black

After T is initiated, i.e. the target of the search is identified, some conditional probabilities, namely, the rows of the topographic matrix corresponding to the target object can be updated (Fig. 3). The

information obtained from a given retrieval session is used for *long-term learning*. The purpose of the update is to increase the conditional probability to be selected by the user of all objects that the user indeed selected. At the same time the connections to the objects that were marked by the user as non-relevant, may be punished.

In retrieval systems that try to learn from the user interaction, the following assumption is made explicitly or implicitly (Müller et al., 2000a; Ishikawa et al., 1998):

> The documents that the user marks as "relevant" are *similar to each other* with respect to a (hidden) feature. The documents that are discarded by the user do not necessarily have a common feature.

One should be careful about grouping the positively marked objects into a class of neighbours. The user who selects relevant objects, compares them to the target he/she is looking for, and not to each other!

In the future we would like to receive some evidence about the user model, which may affect the update strategy, and the prior distribution. However, a simple assumption about the user who wants to find the target and responses consistently, can serve as a generic user.

3 Display update schema

After the feedback is received from the user, $P(T = i|\delta^1, \ldots, \delta^n)$ is calculated according to the Bayes' rule. Then the new evidence should be received from the information variables δ_j that form the display set to present to the user in the next round.

3.1 What is *display update?*

The display update is an important part of the search process, since the speed and quality of the search depends on it. Each iteration should bring us closer to the target. "Closer to the target" may have various interpretations, such as *(a)* the posterior probability $P(T = i)$ of the element is increasing, or *(b)* the target element approaches the top of the ranked list or even *(c)* the expected number of remaining iterations decreases. Recall that the goal is not only to identify the target, but to do it in few iterations in a limited amount of time. It is however not clear which strategy is optimal with respect to both iteration number and calculation costs, as one can see below.

As an illustration to the various display schema, consider the following story:

> A person wanted to find a friend's house in a big city and got lost. The situation is as follows: the friend has a map of the area for pedestrians, the Lost person has a satellite mobile phone. The Lost can call her friend as often as she likes, paying $10 per minute, to ask for the way. The friend at home looks up her position on the map and gives to the Lost one directions.

The approach that we call *naïve* suggests that the friend at home directs the lost friend along the shortest way, but at each crossroads the latter has to make a costly call and ask where she should go next: straight, left, right or maybe back.

We can also advise the friend at home to look up the crossroads where there might be road signs on the way (say, major crossroads) and lead the Lost one towards them, so that from time to time she could see where to go further without calling, or walk in the direction of "through traffic" road sign. This will be called best-selected approach, discussed in Section (3.4). The *most informative* schema, which is also discussed briefly below, is to send the friend to the highway, where all the road signs are present (but it takes *a lot* of walking!).

3.2 Best target approach (*naïve*)

The probability that the target is an element i, $P(T = i)$, can be considered as the score that the element receives during the session. In the naïve model we show to the user the most-probable objects, hoping that the target is among them. This is a simple application of a standard technique used in information retrieval under the name *probability ranking principle* (Robertson, 1977). We note that the denominator in equation (2) does not depend on a particular element we want to rank, and can be replaced with a value that is a constant given object i, since this will not affect the ranking order. Thus, we replace the denominator, noting that in general $P(\delta^1, \ldots, \delta^n) \neq \prod_{s=1}^{n} P(\delta^s)$:

$$P(T = i|\delta^1, \ldots, \delta^n) \propto \frac{\prod_{s=1}^{n} P(\delta^s|T = i)P(T = i)}{\prod_{s=1}^{n} P(\delta^s)}. \quad (3)$$

It is interesting to note that by assuming that each object is selected/de-selected independently of others, we can treat the display set as a series of single objects shown one after another, where the order becomes unimportant.

Any monotonic transformation of the ranking function (3) will produce the same ordering of the objects. Instead of using the product of weights, the formula can be implemented by using the sum of logarithmic weights. All objects that have the ratio $P(\delta^s|T = i)/P(\delta^s)$ close to 1 give almost zero contribution to the score ($\log(1) = 0$). Their probability to be selected by the user does not depend on the user's target.

The obtained values are used to select n objects for the next display set. These are the objects that

have the highest score and have not been shown to the user yet. Strictly speaking, the objects that have been shown to the user have zero probability to be the target, since for them it is known for sure that they are not the target.

The probability ranking principle fits well in the framework of text retrieval, since texts require some time for reading and relevance judgment. Therefore only few feedback iterations are possible. When visual information is presented on the screen, a quick glance is sufficient to evaluate the results, and a new round of feedback can be started at once. Hence the strategy of the display update can be changed, allowing the user to evaluate as large data regions as possible, before the ranked list of results is produced.

3.3 Best information approach (costly)

We may choose the next display set in such a way, that it would maximally reduce the uncertainty of the system, based on the expected amount of on information that could be obtained (Cox et al., 2000; Zhang and Chen, 2002). Such an approach is used in machine learning and it is considered optimal with respect to the number of iterations. We use information theory (Guiasu, 1977) to find the optimal display set for the next iteration. The information I obtained from a display set $(\delta^1, \ldots, \delta^n)$ is

$$I(\delta^1, \ldots, \delta^n) = -\sum_{\mathcal{S}} P(\delta^1, \ldots, \delta^n)\ \log_2 P(\delta^1..\delta^n), \tag{4}$$

where

$$P(\delta^1, \ldots, \delta^n) = \sum_{i \in \Im} P(T = i) \prod_{s=1}^{n} P(\delta^s | T = i).$$

$\mathcal{S}$ is a set of all possible combinations of selected and de-selected objects in the display set $(\delta^1, \ldots, \delta^n)$ that may occur in the following iteration. Here as in (2) we assume that the user selects each object independently of other objects presented on the screen. Note that the denominator in equation (2) is the probability of only one possible feedback on the display set. There are 2^n possible ways of selecting/deselecting n objects. As one can see, the information-based display update schema is costly. Straightforward scanning the collection and selecting all possible combinations of n objects, and taking the best subset, is exponential with respect to the collection size. In PicHunter (Cox et al., 2000), Monte Carlo sampling was used to find a sub-optimal solution. Thus, although the best-information approach may be optimal with respect to the number of iterations, it is far from optimal when speaking about the computation intensity and total time spent on the search. However, the implication that information theory gives is the following: to reduce the number of communication rounds, each cycle should contain as much of (unknown) information as possible, and one should bear in mind that deviation from the optimal strategy may lead to the decrease of information gain and therefore result in declined quality of the search.

3.4 Best selected object approach (intuitive)

As we noticed, to minimize the number of iterations when selecting the new display set, we want to show such objects to the user that would result into best information about the "location" of the target in the database. If we take the case of text retrieval, in some situations the target can be, for instance, a list of articles about the subject. Such articles retrieved as the result set are likely to have the probability to be the target relatively high compared to other documents. However it is more likely that the user will prefer a document containing a (complete) list of references along with some most relevant documents from that list to several relevant documents, that are very similar to each other, but do not cover all the subject.

Intuitively, a group of similar elements in the collection is described best not with the element that only has the highest score $P(T = i)$, but by the one that, in the topographic matrix, is pointed by the largest number of elements that have high probability to be the target. It is easy to show that the maximum information can be expected from the user response to one displayed object when the probability of the object to be selected is exactly 1/2. In this case the uncertainty about the following user action is largest. Let us have a closer look at $P(\delta_i)$. As mentioned in Section (2.1), this term denotes the probability of i being selected by the user in the current iteration, despite the target. We can evaluate it as follows:

$$\begin{aligned} P(\delta_i) = P(\delta_i | T = j) P(T = j) + \\ P(\delta_i | T = j) P(T \neq j). \end{aligned} \tag{5}$$

The probability of a single object *not* to be the target $P(T \neq j)$ is unknown, but, assuming that the target exists in the collection and is unique, we note that

$$P(i \neq T) = P\left(\bigcup_{\substack{j \in \mathcal{U} \\ j \neq i}} (T = j) \right), \tag{6}$$

i.e. if i is not the target then some other element in the collection can be the target.

From (5) and (6), the probability of an object to

be selected can be rewritten as

$$P(\delta_i) = P(T=i) + \left(\sum_{\substack{j \in \mathcal{U}, \\ j \neq i}} P(\delta_i | T=j) P(T=j) \right), \quad (7)$$

where the sum is greater than or equal to zero. If the candidate object is selected based on $P(T=i)$, then one has to specify what is the optimal value of this probability or what is the selection criteria. Note that the "largest value of $P(T=i)$ criterion" gives in most cases an arbitrary value of $P(\delta_i)$ and, thus, not optimal from the information gain point of view[2]. The drop in information gain may ultimately lead to unnecessary search iterations.

A similar statement holds for the display set consisting of more than one element. The optimal, with respect to information, selection consists of independent objects with equal probability for every outcome (see equation (4)). In other words, the optimal selection is such that for the display set under consideration $P(\delta^1, ..., \delta^n) = 1/2^n$ for every possible combination of candidate objects. This consideration is especially important in the begin phase, when all probabilities $P(T=i)$ are not quite distinguishing. Then the maximum information criteria can be, without too big loss of information, reduced to $\tilde{I} = max\left[P(\delta^1, ..., \delta^n)\right]$, which in turn can be approximated by maximum of the product of the corresponding $P(\delta^s)$. This maximum is delivered by the first n elements with the largest value of $P(\delta)$. Every selection based on $P(T=i)$ will be different from those based on $P(\delta_i)$ and thus systematically further from the approximation based on "best $P(\delta_i)$" selection.

This update strategy, however, has its limitations. Computation of $P(\delta_i)$ according to (7) requires scanning all the topographic matrix for each element. For a collection of useful size this will inevitably become a bottleneck. Our approach to the data organisation, when the topographic matrix contains quite many "holes", overcomes this problem. More detail about the construction of the initial topographic matrix is given in the next section.

4 Implementation

4.1 Topographic matrix

The topographic matrix containing conditional probabilities plays an important role in our retrieval model. We performed feature extraction and normalisation, to obtain $N \times N$ kick-off values for the collection of N elements. We store only a fraction of them, assuming that for a large number of objects the following holds:

$$P(\delta^s | T=i) = P(\delta^s), \quad (8)$$

i.e. the fact that the target is i does not affect the probability for s to be selected by the user.

By dropping useless connections between the elements that have weak or no influence on each other, the waste of time and disk space is avoided. In addition, we preserved not only the most similar objects, but also those that are the furthest from each other, assuming that they have negative impact, i.e. i is unlikely to be marked by the user as resembling the target when j is the target: $P(\delta^i | T=j) \to 0$. The question is, what are the assumed, or *default*, values of the dropouts? It is interesting to notice that in (8) the elements are the ones that do not affect the score in the naïve display update scheme. The missing conditional probabilities are simply ignored when updating the object scores according to equation (3). These scores, strictly speaking, are not probabilities, although they are derived with the help of probability theory. Setting $P(\delta^s | T=i)$ to zero would inevitably turn the whole equation (2) into zero, since we certainly leave most of the connections out. In similar situation in content-based text retrieval, (Hiemstra, 2000) used linear interpolation, introducing the notation of *importance* for the query term. We will simply use equation (8) to obtain the default value, retreating to $P(\delta^s)$ as a substitute for the missing $P(\delta^s | T=i)$. This is referred to as a "back-off model".

4.2 Available TREC data

As the data set we used the test set of TREC-2002 video collection, containing $N = 8869$ key frames extracted automatically from video data. To initiate the topographic matrix we used colour-based features, similar to those described in (Stricker and Orengo, 1995). For each image we took three central moments in HSV colour space: average value, variance and skewness in combination with weighted L_1 (Manhattan) distance. This compact feature set has quite good discriminating ability. The values of pair-wise distances varied from 0 to 7 and needed to be brought to probability range of $[0, 1]$. Simple division by the largest value (or the sum of all values) may distort the real distances, since an outlying value of 7 would cause squeezing the rest of similarities, most of which lied in $[1, 3]$, into a small interval, making the features undistinguishing. Usually as an approximation the "3-sigma" rule is applied (see e.g. (Rui et al., 1997b; Su et al., 2001)), because the features are processed online when the timing is constrained. Since we address the feature space offline, we could afford using the "full" normalisation that consisted of the following steps:

[2]Generally speaking, $P(\delta_i)$ is always greater than the corresponding $P(T=i)$, with two exceptions: when i is the target itself, and then $P(\delta_i) = P(T=i) \equiv 1$. This case is uninteresting in our framework.

1. Calculate pair-wise distances in the HSV metric feature space. Make sure that the pair-wise distance values have a distribution resembling the Gaussian.
2. Assuming normal distribution of pair-wise distances, for each pair of objects calculate the probability of their metric distance to be smaller or equal than its value using a cumulative normal distribution. This operation will re-scale the similarities with respect to their frequency of occurrence.
3. Subtract the obtained probability value from 1. This inversion assigns the largest probability for the objects that have the smallest distance in the selected feature space.

We plotted histograms of pair-wise distances in the selected feature space (Fig. 4) and found that logarithms of the distances are close to the shape of normal curve[3]. The log-igification of the similarities gives more distinguishing scale to the elements that have small distance from each other, which is a desired property, since we are more interested in differentiating between elements that affect each other rather than between those that seem independent.

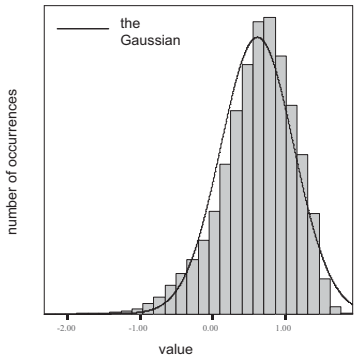


Figure 4: Example histogram of log-ifyed pair-wise distances of the collection, random sample of 3%.

The normalization is computationally quite intensive to perform for the whole dataset, since the major part of it would be discarded. We calculated iteratively the mean and standard deviation, and for each pair of images checked the metric distance before calculating the probability. Finally, we added neighbours for those elements that had too few close connections, disregarding the threshold. In this way we obtained an initial topographic matrix which can be updated based on the successful search sessions.

[3]Colour histograms and L_0 distance (*histogram intersection*) yielded similar distribution, with somewhat less steep right tail.

The optimal threshold, or the cut-off value for the dataset, is a topic for further investigation. The question is, how many connections from the topographic matrix can be sacrificed to performance so that the search quality would not drop dramatically. If we use a 9-element display set and leave 10% of connections, then, in the case when the elements on the screen are representing 9 non-overlapping regions from the collection, we have 90% of the collection affected by the user feedback. We realise that such a display update is unfeasible unless the user's information need is known. The upper bounds for the cut-off value is 1.5 of standard deviation for *neighbours* and 1.8 σ for *anti-neighbours*, which gives roughly 10% of the data. In any case we stored at least 1% of the elements as closest neighbours.

Ideally, if we leave only random $\bar{n} \approx 1.1\%$ connections, then in the case of display set containing 2 elements the expected number of objects that are connected to them would approximately be $8869 \cdot (0.011 * 0.011) = 1.073149,$. i.e. at lest one element. Since the remaining connections are not (completely) random, some meaningful results could already be expected with even smaller $\bar{n}$. However, when the display set contains $n > 2$ objects, it is unlikely that there will be an object connected to all n candidates, but the expected number of images connected to at least 2 of them is C_n^2, and a new display set can be selected among them.

4.3 "Toy" collection

In addition to the real dataset we used generated data containing geometric shapes: triangles, squares and circles of different colours, in various combinations. Some images were duplicated, and hand-drawn objects were added as noise. Part of the collection was duplicated once more and the colours were inversed. We used three feature sets with this collection: one addressed colours, and thus ignored the geometric shapes. Second contained a higher-level feature set, taking into account only type and the number of each kind of geometric shape, regardless the colour. Finally, the third feature representation was the combination of shape-based and colour-based features with weights, respectively, 0.2 and 0.8. The mapping of distances to probabilities was performed in the same way as in the TREC collection.

4.4 User interface

Before starting the search, the user is given a short introduction and instructions, how to give the feedback. The target image is then picked randomly from the collection. The user should find the target image, which for convenience is always present on the screen. The display set for the TREC data collection consisted of 9 images, and the feedback was ternary: the user was allowed to mark whether the

image was "good" or "bad", or set the pointer into neutral position ("don't know", default value). For the toy collection we showed only 4 images, because the collection size is small.

We started testing with the naïve update schema. The user was supposed to do one of the tree actions: (1) Select suitable images and press "Feedback" or (2) Select the target image if it is on the screen and press "Found" to retrieve the target and the ranked list of neighbours, or (3) Press 'Cancel" to terminate the unfinished search and get the best ranking images on the screen anyway. Only action (2) was considered as a successful search. All user actions are logged, and the distribution of $P(T = i)$ may be reconstructed for each moment of the search. In the best-selected display update, the user may only see the current ranked list by pressing "Found" and, if necessary, continue search by giving the feedback. The target is then indicated by the user as an extra step.

5 First conclusions

There are some conclusions that can be drawn already at the initial stage of the experiments:

1. Colour-based features alone are confusing for the images with contrasting colours. These features are far to low level, and large relevance feedback statistics should be collected before the topographic matrix is updated to the semantic level.
2. The naïve and best-selected display update schema both tend to select objects that are very similar to each other. This is especially obvious in the toy-collection, where some part of the data is identical to another part, with respect to geometric feature space. We expect that this effect would be decreased for the best-selected scheme by introducing a penalty function $\mathcal{P}$, which would improve approximation to the display update made in section 3.4.

   Since in the naïve scheme, the best targets in the current iteration are displayed to the user in the next round, the updated display consists of the closest neighbours of the good examples selected by the user, and, respectively, the furthest objects from the bad examples, and in this form mimics k-nearest neighbour retrieval technique. No wonder that k nearest neighbours are likely to be nearest neighbours of each other!
3. When binary feedback is used, the user's inconsistency is especially harmful, since images from the same class occasionally get different marks (one is selected, the other one is not). As mentioned in (Müller et al., 2000a), too much of negative feedback may damage the search session. However, the use of the "neutral" feedback button in the interface reduces the quality of the search, since it wastes screen space (and the user's attention). Alternatively, overwhelming influence of negative feedback can be eliminated by more careful selection of the display set and reducing (or even eliminating) the amount of anti-neighbours in the topographic matrix.

6 Future work

In the experiments we plan to clarify the following points:

1. What is the optimal strategy for the display update with respect to our data structure?
2. What is the optimal amount of connections in the topological matrix?
3. How does "neutral" feedback affect the search quality; What effect is brought by the presence of anti-neighbours in the database?
4. How to integrate transcripts of speech recognition of the video material into the system?

We need to perform a number of experiments, clarifying the questions listed above. In the experimental phase the target is always picked up from the dataset. To perform automatic simulation, we will use TREC-2002 topics as possible targets. The result sets provided by the TREC as answers to each query, will indicate positive feedback from the user. In this way we are able to clarify some research questions. The advantage of the automated system is that the exact same relevance judgement is done for different setups, and the same target may be retrieved many times by different versions of the system.

We plan two types of search: unbounded and limited. In the unbounded version the user is free to search as long as needed. In the limited conditions the user is allowed to make only a certain number of iterations, and then the resulting ranked list is studied. The checkpoints for the results are determined after 10 ($\log_2(8869/9) = 9.945$, expected number of rounds in case of optimal strategy) and 20 iterations.

We determine the quality of the search by the rank of the target object achieved at the checkpoints, and the highest rank achieved by the target element. In the unbounded version, the criteria are the number of iterations before seeing the target and the number of successful searches with a given number of iterations. Precision-recall graphs, a traditional method of evaluation the search quality, have a somewhat different meaning when the assumption that the target is unique and exists in the collection is made. Nevertheless, the subjective manual judgement of relevance of the result set is feasible. To evaluate any of the above strategies, we will also look at the

convergence of the rank of the target object during the search session.

7 Acknowledgement

The authors acknowledge *KPN Research, The Netherlands*, for financial support.

References

S. Aksoy and R. M. Haralick. 2000. Probabilistic vs. geometric similarity measure for image retrieval. In *IEEE Conf. Computer Vision and Pattern Recognition*, 06.

C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. 1999. Blobworld: A system for region-based image indexing and retrieval. In *Third International Conference on Visual Information Systems*. Springer.

I. J. Cox, M. L. Miller, T. P. Minka, T. V. Papathomas, and P. N. Yianilos. 2000. The bayesian image retrieval system, pichunter: Theory, implementation, and psychophysical experiments. *IEEE Tran. On Image Processing*, 9(1):20–37.

A. P. de Vries, N. Nes N. Mamoulis, and M. L. Kersten. 2002. Efficient k-NN search on vertically decomposed data. In *ACM SIGMOD International Conference on Management of Data*, Madison, WI, USA, June.

C. Faloutsos and K.-I. Lin. 1995. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In Michael J. Carey and Donovan A. Schneider, editors, *Proc. of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, California, 22–25.

M. Flickner, H. S. Sawhney, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. 1995. Query by image content, the QBIC system. *IEEE Computer*, 28(9):23–31.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. 1995. *Bayesian Data Analysis*. Chapman & Hall.

D. Geman and R. Moquet. 1999. A stochastic feedback model for image retrieval. Technical report, Ecole Polytechnique, 91128 Palaiseau Cedex, France.

S. Guiasu. 1977. *Information theory with applications*. New York: McGraw-Hill.

D. Hiemstra. 2000. A probabilistic justification for using tf.idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2):131–139.

Y. Ishikawa, R. Subramanya, and C. Faloutsos. 1998. MindReader: Querying databases through multiple examples. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 218–227, 24–27 .

W. Kraaij, T. Westerveld, and D. Hiemstra. 2002. The importance of prior probabilities for entry page search. In *Proc. of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 27–34. ACM Press.

M. Maron and J. Kuhns. 1960. On relevance, probabilistic indexing, and information retrieval. *Journal of the ACM 7*, pages 216–244.

H. Müller, W. Müller, S. Marchand-Maillet, T. Pun, and et al. 2000a. Strategies for positive and negative relevance feedback in image retrieval. In *Proc. of International Conference on Pattern Recognition (ICPR'2000)*, Barselona, Spain, September.

H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun. 2000b. Long-term learning from user behavior in content-based image retrieval. Technical Report 00.04, Computer Vision Group, Computing Centre, University of Geneva, rue Gnral Dufour, 24, CH-1211 Genve, Switzerland, March.

S. E. Robertson. 1977. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304.

Y. Rui, T. Huang, S. Mehrotra, and M. Ortega. 1997a. Automatic matching tool selection using relevance feedback in MARS. In *Int. Conf. on Visual Information Retrieval*.

Y. Rui, T. Huang, S. Mehrotra, and M. Ortega. 1997b. A relevance feedback architecture in content-based multimedia information retrieval systems. In *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries, in conjunction with IEEE CVPR*.

M. A. Stricker and M. Orengo. 1995. Similarity of color images. In *Proc. of the Storage and Retrieval for Image and Video Databases III*, pages 381–392, San Diego/La Jolla, California, USA, February.

Z. Su, S. Li, and H. Zhang. 2001. Extraction of feature subspaces for content-based retrieval using relevance feedback. In *ACM Multimedia*, pages 98–106.

N. M. Vasconcelos and A. B. Lippman. 2000. A probabilistic architecture for content-based image retrieval. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*.

P. Wu and B. S. Manjunath. 2001. Adaptive nearest neighbor search for relevance feedback in large image databases. In *Proc. of ACM International Multimedia Conference*, Ottawa, Canada, October.

C. Zhang and T. Chen. 2002. An active learning framework for content based information retrieval. Technical Report AMP 01-04, Advanced Multimedia Processing Lab, Electrical and Computer Engineering Carnegie Mellon University Pittsburgh, PA 15213, March.

Content-Based Image Retrieval: Color-selection exploited

E.L. van den Broek, L.G. Vuurpijl, P. Kisters, and **J.C.M. von Schmid**
Nijmegen Institute for Cognition and Information
P.O. Box 9104
6500 HE Nijmegen, The Netherlands
e.vandenbroek@nici.kun.nl

Abstract

This research presents a new color selection interface that facilitates query-by-color in Content-Based Image Retrieval (CBIR). Existing CBIR color selection interfaces, are being judged as non-intuitive and difficult to use. Our interface copes with these problems of usability. It is based on 11 color categories, used by all people, while thinking of and perceiving color. In addition, its usability is supported by Fitts' law. The design of the color selection interface provides fast, unambiguous, intuitive, and accurate color selection.

1 Introduction

With a median of 14.38 images on each web page (Lyman and Varian, 2000) the World Wide Web (WWW) consists to a great extent of images (Lew, 2000). Unfortunately, most Image Retrieval engines are text-based and do not provide the means for searching on image content. Given the exploding market on digital photo and video camera's, the fast growing amount of image content further increases the need for image retrieval engines.

Although query-by-text methods are fast and reliable when images are well-named or annotated, they are incapable of searching in unannotated image collections. Content-Based Image Retrieval (CBIR) methods are capable of searching in such collections. However, present methods used in CBIR are not fully reliable, nor fast enough to handle image databases beyond a closed domain and the techniques used are still subject of development.

The research described here, is part of the research line Eidetic[1]. This line of research is targeted on the integration of four query-paradigms, each of them object[2]-based:

1. Query-by-text, a query is done by way of either a single keyword or a description of the image.
2. Query-by-example, an example image is used. Its features (e.g. color, texture, and shape), are extracted to facilitate CBIR.
3. Query-by-sketch, this query paradigm was based on the findings of Schomaker et al. (1999), who state that users can sketch the shape of an object and use this as a query;
4. Query-by-color, the color of an object in the image, is defined and used for querying.

The query paradigms query-by-text, query-by-example, and query-by-sketch are implemented in the current system Vind(x) (see `http://kepler.cogsci.kun.nl/vindx` and Vuurpijl et al. (2002)).

For the present paper, we focus on the fourth query method: query-by-color and in particular on how color can be selected. A color selection interface, based on theories of human color perception, is presented. This is done in such a way, that fast and accurate color selection is promoted. Rui et al. (1998) emphasize the relevance of user-feedback and propose that the user should be involved in the complete development of new CBIR engines. In our opinion this is not the only demand that has to be met for a successful CBIR engine. Available information concerning the users cognitive abilities should be considered as well. This holds for all three components of CBIR-engines:

[1] Eidetic: Intelligent Content-Based Image Retrieval is a project within the Token2000 research line. See also: http://www.token2000.nl

[2] An object is defined according to Merriam Webster's online dictionary (http://www.m-w.com) as: A thing that forms an element of or constitutes the subject matter of an investigation or science.

1. Definition of the query *by the user* (i.e. input of *content*)
2. The image retrieval engine, conducting *intelligent* image analyses (i.e. based on and adapted to the *the users' characteristics*).
3. The presentation of retrieval results (i.e. *the output to the user*).

Based on this perspective we have conducted the research toward a new color selection user-interface.

We will first briefly discuss *human color perception* and practical and technical shortcomings accompanying color perception and selection. Next, color spaces, used in both CBIR and in modeling human color perception, will be discussed. This overview will lead us to a review of existing color selection user-interfaces. After this, we will present the research underlying the new color selection user-interface, followed by the introduction of this new color selection interface. In addition, the importance of using query-by-color is illustrated, especially in relation to shape. We end this paper with plans for future research and the final conclusions.

2 Color perception

Human color perception is a complex function of context, for example: illumination, memory, object identity, culture, and emotion can all take part (Czajka, 2002; Schulz, 1998; Kay, 1999). As already mentioned by Forsyth and Ponse (2002): "It is surprisingly difficult to predict what colors a human will see in a complex scene; this is one of the many difficulties that make it hard to produce really good color reproduction systems. Human competence at color constancy is surprisingly poorly understood. The main experiments on humans (McCann et al., 1976; Arend and Reeves, 1986) do not explore all circumstances and it is not known, for example, how robust color constancy is or the extend to which high-level cues contribute to our color judgments." So, we are not even close to having a good model for human color perception.

Modeling human color perception becomes even more complex due to a range of technical shortcomings. The colors perceived on photographs, for example, can be quite a poor representation of the color of the surfaces when being viewed directly. Furthermore, there is a loss of "color authenticity", due to the compression techniques, used on images that are presented on the Internet. Finally, we want to mention monitor settings (e.g. brightness, contrast) as color disturbing factors. Both the complexity of human color perception and the technical shortcomings are sources of variability, making color a "noisy" source of information.

3 Color spaces

Many attempts have been made to model color perception (Forsyth and Ponse, 2002) by researchers of various fields: psychology, perception, computer vision, image retrieval, and graphics. Some of these resulted in well defined color spaces. The list of color spaces is almost endless. A few of the most important color spaces are:

1. RGB (Red, Green, and Blue)
2. HSV (Hue, Saturation, and Value; also named: HSB and HSL (HLS))
3. CMYK (Cyan, Magenta, Yellow, and Black (Key))
4. CIE (Centre International d'Eclairage)[3] XYZ
5. Munsell[4]
6. RBW (Red, Blue, and White) (Shirriff, 1993)

In the present research, we adapt the W3C [5] definition of color spaces: "A color space is a model for representing color numerically in terms of three or more coordinates." [6]

Color spaces are needed in:

1. The representation of color-ranges.
2. The manipulation of colors, as is done in the graphics industry.
3. Mixing of colors.
4. The retrieval (i.e. matching) of colors.

So it is evident that CBIR engines, using color as feature, need a color space for color matching. However, often CBIR engines also use a color

[3] http://www.cie.co.at/ciecb/
[4] http://www.munsell.com
[5] The World Wide Web Consortium
[6] http://www.w3.org/Graphics/Color/sRGB.html

space in their color selection interface. The color selection interface we developed, has the goal to provide an optimal guidance for the user in the color selection process during querying. Our color selection interface is color space independent, because we have chosen to present a limited number of colors and thus require no color manipulation. Only the advanced features of the color selection interface need an underlying color space, for computing intermediate colors.

Until now, it is still a topic of discussion which color space is the most intuitively one for humans (Douglas and Kirkpatrick, 1996). It is essential that the color space used, follows human color perception as closely as possible because the user is the final judge, of the performance of the system using the color space.

4 Color selection interfaces reviewed

The need for a review of the interfaces of CBIR engines in general is still present. Existing reviews, such as those of Veltkamp and Tanase (2000), Veltkamp et al. (2001), and Venters and Cooper (2000) emphasized the various image retrieval techniques and not their interfaces. Others such as Steiner (2002) did only briefly discuss the usability of 36 freely available web based color selectors. This, despite the fact that a lot of performance can be gained using a well-designed user-interface, improving human-computer interaction.

We did, therefore, conduct a new review on ten online CBIR engines, emphasizing interface aspects and judging human-computer interaction. Ten subjects participated They were asked to have special attention for the color selection user-interfaces. However, despite the fact that most CBIR engines do use colors they lack the presence of a color selection user-interface. An exception is, for example, IBM's CBIR engine QBIC[7] [8], it has reserved a prominent place for color selection facilitating query-by-color.

The information gathered by the reviews of online CBIR engines did not provide enough information, due to the lack of color selection user interfaces used. We, therefore, additionally reviewed several color selection user-interfaces,

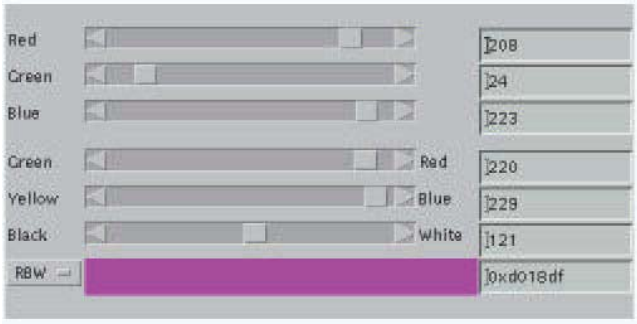


Figure 1: Shirriff's color selector[13], using solely sliderbars.

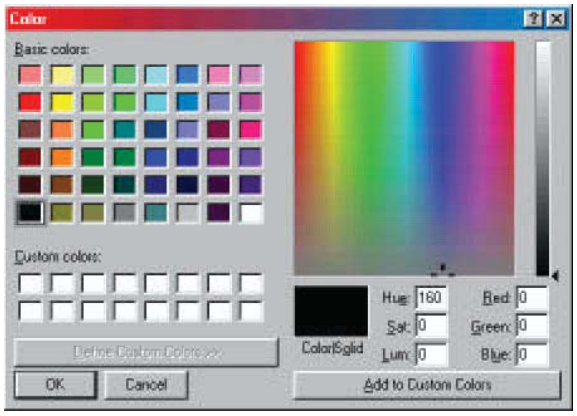


Figure 2: The EditPlus 2 color selector.

suitable for online use. An exhausting number of color selection interfaces has been developed in the last 10 years. Some well known, such as the color selection interface of Powerpoint, some less familiar such as that of EditPlus (see Figure 2). Others are online Java-applets and applications [9] [10] [11].

The reviewed color selectors can be divided into three groups depending on their color definition method.

1. Slider-bars represent the axes of the used color space (see Figure 1), providing a way to define a color. In addition, these color selection interfaces present a panel in which the defined color is shown. In some of

[7] http://wwwqbic.almaden.ibm.com/
[8] http://www.hermitagemuseum.org/fcgi-bin/db2www/qbicSearch.mac/qbic?selLang=English
[9] http://burtleburtle.net/bob/java/color/color.html
[10] http://www.righto.com/java/colorselector.html
[11] http://java.sun.com/products/jlf/ed2/book/HIG.Dialogs4.html

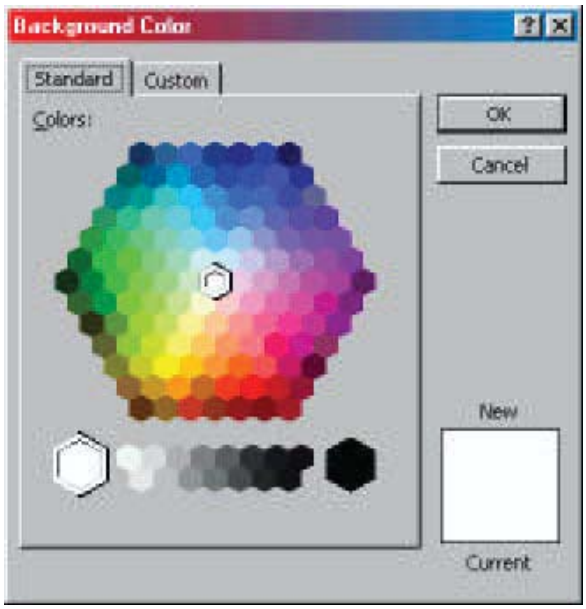


Figure 3: The MS PowerPoint 2000 "standard" color selector.

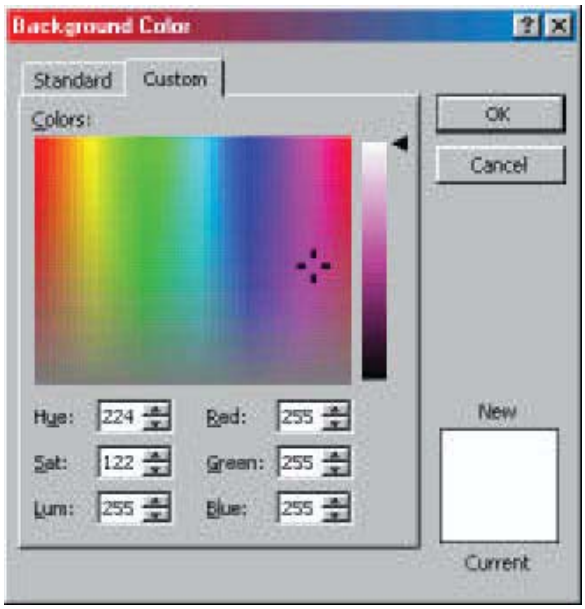


Figure 4: The MS PowerPoint 2000 "custom" color selector.

these interfaces, the sliderbars are combined with numeric fields where the sliderbar position is presented (see Figure 1). The numeric field can be accessed directly for fast color definition, for example when the RGB model is used, a user might know the numeric value for a certain color.

2. A discrete color matrix, with each cell having its own color (see Figures 2 and 3).

3. A square that describes and is perceived as a continuous 2-dimensional representation of the used color space (see Figures 2 and 4). The color of choice, can be directly chosen in the panel. In some color selection interfaces the square is extended with a panel, representing a third dimension of the color space (see Figures 2, 3, and 4).

Some interfaces involve two or three of these color selectors in parallel (see Figures 1, 2, 3, and 4). In such cases, the user can determine which interface to use. Other color selectors provide the possibility to replace the presented colors with a larger set of colors. Hence, the user is often given the choice to use at least two color spaces (see Figures 1, 2, and 4).

We will now discuss a usability study on color selection conducted by Everly and Mason (1999), who adapted another method of research then the reviews just discussed. They measured speed, accuracy, and ease of use of four color selection user-interfaces. This was done for four color selection methods: Apple's Crayon (see Figure 5), HSV, RGB, and CMYK color selectors. On *all* three criteria the Crayon color selector, which uses a discrete presentation of colors, outperformed the other three color selectors. With this, experimental proof has been given that the discrete presentation of colors for color selection purposes, is the best choice. The Crayon color selector provided a choice between 60 colors. This is only a small subset of the colors that can be selected, compared with the other color selection interfaces. However, in the next section we will argue in favor of an even further decrease of the number of colors presented in color selection interfaces.

The results of the study of Everly and Mason (1999) and our review, can be summarized as a triplet of main problems with regard to most of the existing color selection interfaces :

1. Most color selection interfaces require that the user is familiar with the presented color space. Imagine using 3 sliderbars, repre-

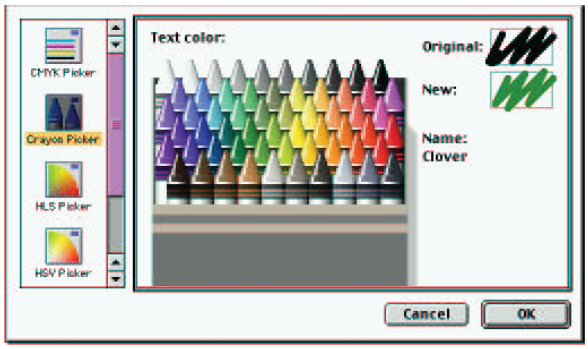


Figure 5: Apple's crayon color selector.

senting the Red, Green, and Blue axes of the RGB-model, for defining a color such as pink.

2. The interfaces provide multiple ways to define a color, which is confusing for the users.
3. The users described the color selection interfaces as non-intuitive and often too "complex".

The problems mentioned above are strongly related. The question emerges, why such color selection interfaces have evolved. Color selectors in the graphics industry were present years before the first CBIR engine was born. However, color selectors for the graphics industry do have other demands than color selectors for CBIR. Subtle level crossings for example, do not have to be made in a CBIR context but are very regular in graphics design.

In CBIR no differentiation between more-or-less similar colors is required. The presentation of hundreds or even several thousands of colors was reported as almost overwhelming for the users and with that as being inefficient. Nevertheless, current CBIR engines that allow the user to specify colors (e.g., QBIC[14][15]), exhibit such an interface.

In the next section of this paper two forms of fundamental research are described, underlying our new color selection interface. In our opinion, are these research forms a very rich source of information, essential in the development of interactive applications, such as CBIR engines.

[14] http://wwwqbic.almaden.ibm.com/

[15] http://www.hermitagemuseum.org/fcgi-bin/db2www/qbicSearch.mac/qbic?selLang=English

5 The practical use of fundamental research

We will now guide you through fundamental research in several fields, illustrating two concepts: color categories and Fitts' law, both relevant to the design of a color selection interface.

5.1 Color categories

Humans have a relatively poor color memory over the long term. They tend to remember colors as members of categories. Most people distinguish 11 color categories (Goldstone, 1995; Kay, 1999; Lai, 2000; Shirriff, 1993; Uchikawa and Ikeda, 1986). This set of colors can be divided in 6 *primary colors* (black, white, red, green, blue, and yellow) and 5 *secondary colors* (gray, brown, purple, pink, and orange). Note that this is another division than the often used distinction between the primary colors(red, green, and blue), the secondary colors (yellow, magenta, cyan), and the *none-colors*(black, white, and gray).

There is a range of explanations for the existence of color categories, one of the strongest is the Sapir-Whorf view (Whorf, 1956). According to this view, linguistic categorization can influence non-linguistic perception and cognition. So, the higher frequency of appearance of basic color categories compared to other colors and the names given to these, provide an explanation for the more rapid recognition of these colors.

Basic color categories are relatively insensitive to various sources of variability, such as: illumination, memory, object identity, culture, and emotion. All people tend to categorize colors in the same way.

Both the fast recognition of the 11 basic colors and the fact that they are perceived by *all* people as being "that" color are great advantages, if used in a color selection interface.

5.2 Fitts' law

A completely different point of view is that of human motorics. An interface has a certain complexity from the perspective of human motoric capabilities. As every computer-user experiences quite regularly, small objects on the computer screen are more difficult to select, than larger are. This intuitive statement is subscribed by Dix et al. (1998): "Since users find it difficult to manipulate small objects, targets

should generally be as large as possible". This can be explained, using Fitts' law (Fitts, 1954), which is commonly expressed in the following form:

$$T = a + blog_2(\frac{A}{W} + 1),$$

where T is the selection time of a target of width W, with A as the distance that has to be moved. a and b are empirically determined constants.

Given the constraint of the limited surface of a computer screen, the more colors, the smaller the area available for a color. A color selection interface presenting a small amount of colors would therefore, according to Fitts' law, result in an enormous decrease in selection time, compared to an interface presenting a larger amount of colors.

This is an important advantage for a CBIR color selection interface. As Dix et al. (1998) already stated: "Speed and accuracy of movement are important considerations in the design of interactive systems."

6 The Vind(x) color selector

In the previous sections we have discussed the ingredients necessary to define a color selection user-interface for a CBIR engine. More than anything else, has the complexity of human color perception and its relation to human-computer interaction been illustrated. A number of issues that have to be taken into account, can be extracted:

1. From the perspective of usability, a limited set of colors is sufficient for CBIR color selection.
2. The large variability in human perception of color due to: Human memory, differences between people (e.g. culture and language), and differences in perception within people (i.e., changes in perception over time).
3. Environmental noise is present, due to: Environmental conditions (e.g. lighting) and technical shortcomings (e.g. monitor settings and compression techniques).
4. The lack of a satisfactory color space.
5. Reviews of color selection interfaces.
6. Usability research.
7. People think in and perceive colors in terms of the 11 color categories.
8. Human motoric capabilities, e.g. Fitts' law.

Our main concern was to develop a color selection user-interface that was insensitive to the diverse sources of variability in human-computer interaction, concerning color perception, definition of color, and selection of color.

We did not want to depend to heavily on one of the color spaces, because none of them has proven to work correct in all circumstances. We also took human motoric capabilities into account. Furthermore, we did not want to be dependent on environmental variables and technical shortcomings. Above all, we wanted our color selection interface to be of use for all users. So, the interface had to be robust concerning both the variability in color perception within and between people.

We have chosen to exploit the fact that all humans tend to think and perceive colors in 11 basic color categories. Summarizing, we can state that there are three advantages in using the 11 basic color categories:

1. They are robust to variability between people (i.e. All people are different and so is their color perception.).
2. They are robust to variability within people (i.e. People have changing moods and, for example, also changing perceptual abilities.).
3. There is no need for the use of a color space, because we present a limited number of colors and thus require no color manipulation.
4. An important additional advantage of this limited set of colors, is that there is space for a relatively large area for each of the colors, on the screen.

Based on these four advantages we have chosen to take the 11 basic color categories as the basis for our color selection user-interface. Please note that the classification of the 11 colors in a matrix of 4 by 3, provided space for an additional color. From the point of purely practical considerations, we have chosen to use this place. The 11 colors were, therefore, extended with the color "Cyan", a secondary color, often

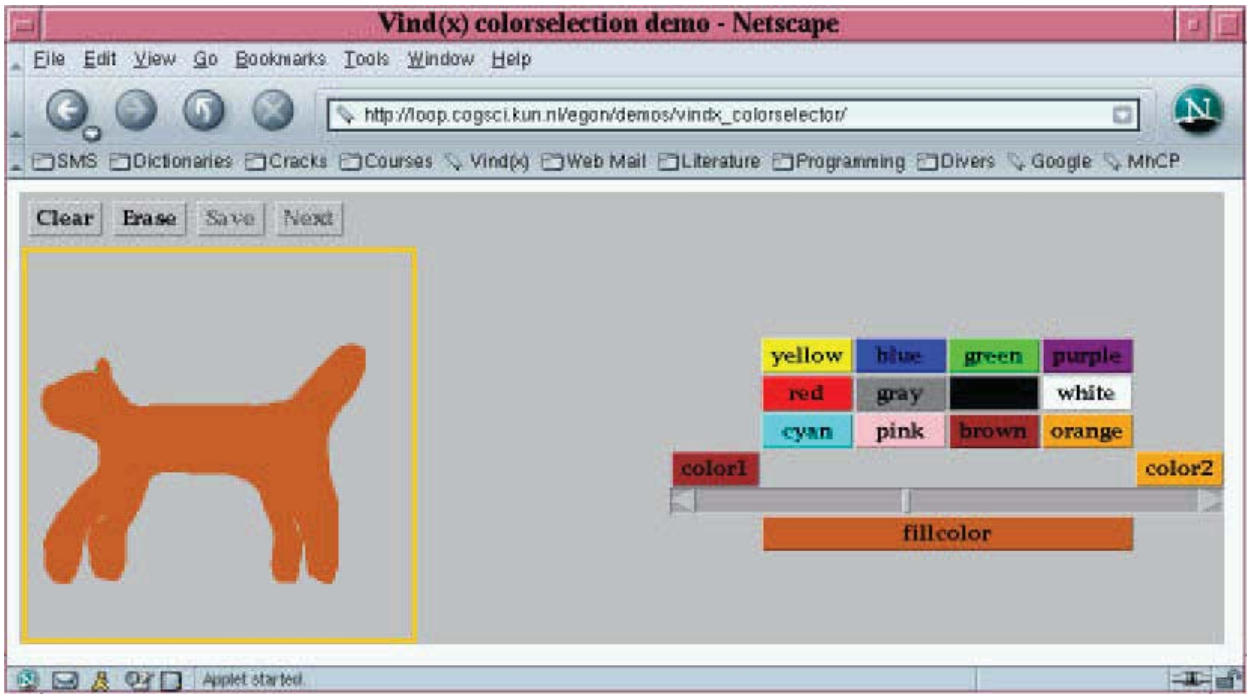


Figure 6: The Vind(x) color selector.

associated, for example, with the air and with the see.

We placed prototypes of each of the basic color categories in a matrix. The RGB-values of the prototypes are determined, using the W3C standards. This color selection user-interface assists in a fast, unambiguous, intuitive, and accurate choice of color.

However, we did also want to take into account that the users may want to specify a particular color that is not one of the 11 basic colors. To cope with this incidental need, our color selection user-interface involves the possibility to mix two colors. With this additional feature, the color selection interface is augmented with the possibility to specify a color of choice. This is in line with the recommendations of Everly and Mason (1999).

Although the 11 color categories will remain the basis of our color selection, the way they are presented is still a topic of discussion. The ordering of the 11 basic colors is not a trivial problem. The complexity of the design of a user-interface that has to be both functional and eye-appealing remains challenging, due to the lack of formalized design rules (Czajka, 2002). However, we will in the near future conduct a series of experiments, in which a number of alternative presentations of the color selection user-interfaces will be compared.

The fully operational online demo version is presented available at: `http://loop.cogsci.kun.nl/egon/demos/vindx_colorselector/`. In Figure 6 a screenshot of the color selection user-interface is shown.

7 Shape and color

Color is for both human perception and CBIR of great importance and is therefore given much attention by researchers. However, the relation between color and shape has received little attention. This relation is two-folded:

1. Color influences human object recognition.
2. The shape category of an object influences the perceived color of it.

The influence of color perception on object recognition is described by Goldstone (1995) in his article "Effects of categorization on color

perception" he states that: "high-level cognitive processes do not simply operate on fixed perceptual inputs; high level processes may also create lower level percepts".

The relation between shape and perceived color is observed by Sacks (1995) who describes the horror people experience when perceiving objects, after they lost their ability to see color. Meadow (1974) described the disabilities his patients had in distinguishing between objects, due to the loss of the ability of seeing color. He further notes that these problems are especially important for the recognition of those objects that rely on color as a distinguishing mark.

So, shape and color influence each other very strongly and can not be separated. Therefore, we have chosen to integrate the described color selection interface, with the sketch pad of CBIR engine Vind(x)[16] (Figure 6). This sketch pad is used to define an object of interest. However, the literature discussed above, illustrates the need for merging image retrieval, using query-by-color, and image retrieval, using query-by-sketch. How this should be done will probably remain a question for at least several years.

8 Future research

This paper illustrates, the complexity of the design of the first component of CBIR engines: the query definition interface. We have presented guidelines for the design of a part of the first component of CBIR engines: the color selection interface. However, the research toward design-rules for the other parts of the query definition interface, is still in progress. Parallel to this, we are working on the development of the second component of the query-by-color engine: the color matching engine. Here, we again try to adapt the system as much as possible to the user. We aim to embed important human color perception features in the color matching engine (e.g. the principle of color constancy). After the development of this new color matching engine, we will focus on the development of the third component of CBIR engines: The interface presenting the results of CBIR. Such an interface should give the user as much information as possible regarding the processes, underlying CBIR. This facilitates understanding the use of the features (shape, color, and texture) and decreases the errors made using them.

Until now, we have discussed the features shape and color, but have not mentioned texture as a feature. Especially, for image retrieval, using query-by-color, texture has to be taken into account as an important additional variable. The integration of texture in query-by-color is an interesting, but also highly complex topic of discussion. For now, we question, if people can define texture, and even if we assume that they can, how an interface providing the facility to define texture, has to be developed?

Both the research toward the user-interfaces, of query definition and presentation of the CBIR results on the one side, and the research toward the color and texture matching engine on the other side, will be used in the adaptation and the extension of the CBIR-engine Vind(x).

9 Conclusions

The present paper has given an overview of a broad range of topics related to color selection. This overview emerged from the research toward a color selection interface for the CBIR engine Vind(x). To fulfill the need for a color selection user-interface for CBIR engines, we, as a first step, conducted three lines of research:

1. Fundamental research
2. A review on color selection interfaces
3. A usability study

From this triplet of research lines a number of issues was extracted:

1. The presentation of a limited set of colors can be sufficient and is even most efficient for color selection in CBIR.
2. People tend to think in terms of 11 color categories and perceive colors as belonging to one of these 11 color categories.
3. People lack the knowledge, concerning color spaces, which is needed in most CBIR color selection interfaces.
4. The size of the surface available on the screen, for each color, influences the amount of time needed for selection of these colors. The larger the surface available for each color the faster the color selection will take place.

[16] http://kepler.cogsci.kun.nl/vindx

These issues revealed a new view on color selection in CBIR. Color selection in CBIR has completely other demands then, for example, color selection in graphics industry, where the users are all experts. Hence, there is a need for the development of a user-interface that is relatively robust toward variability in user-experience and perception, between and within users. The three most important advantages of our new color selection interface are:

1. A decrease in selection time.
2. High usability, compared to existing selection interfaces.
3. An intuitively working interface

These advantages are supported by both the 11 color categories and Fitts' law. So one can conclude that the presented interface, facilitates good human-computer interaction.

However, so far we have not taken into account a relatively large group of users. Assuming that in time CBIR engines will be used in a range of settings, by numerous users, the group of users suffering from a form of color-blindness (e.g. red-green or yellow-blue color-blindness) is significant. We have implemented a simple solution for this problem: The colors presented in the color selection interface, are labeled with their name. Such a solution is only possible with a limited set of colors, such as the 11 basic colors. In such a limited set of colors, each of them can have a large enough area available on the screen, to present their label in a readable size of font.

This last advantage of the color selection user-interface illustrates the strength of its simplicity. The color selection user-interface, presented in the current paper, provides fast, unambiguous, intuitive, and accurate color selection. In addition, a number of requirements on color selection interfaces, especially in CBIR are listed in this paper. These requirements can serve as guidelines for the development of color selection interfaces in the future.

Acknowledgments

The Netherlands organization for scientific research (NWO) is gratefully acknowledged for funding the Eidetic project, in which this research was done.

In addition, we thank the reviewers for their comments. They pointed out both the weak and strong points of the first version of this paper, in a very accurate and pure style.

References

L.E. Arend and A. Reeves. 1986. Simultaneous colour constancy. *Journal of the Optical Society of America*, (3):1743–1751.

K. Czajka. 2002. Design of interactive and adaptive interfaces to exploit large media-based knowledge spaces in the domain of museums for fine arts. Master's thesis, University of Applied Science Darmstadt: Media System Design.

A.J. Dix, J.E. Finlay, G.D. Abowd, and R. Beale. 1998. *Human-Computer Interaction.* Pearson Education England, 2nd edition.

S. Douglas and T. Kirkpatrick. 1996. Do color models really make a difference? In M. J. Tauber, V. Bellotti, R. Jeffries, J. D. Mackinlay, and J. Nielsen, editors, *Proceedings of CHI 96: Conference on Human Factors in Computing Systems*, pages 399–405. ACM, New York: ACM Press.

D. Everly and J.S. Mason. 1999. Color selection methods using the color picker. http://www.otal.umd.edu/SHORE99/jsmason/.

P. M. Fitts. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47:381–391.

D.A. Forsyth and J. Ponse. 2002. *Computer Vision: A modern approach.* Prentice Hall.

R.L. Goldstone. 1995. Effects of categorization on color perception. *Psychological Science*, 5(6):298–304.

P. Kay. 1999. Color. *Journal of Linguistic Anthropology*, (1):29–32.

T.-S. Lai. 2000. *CHROMA: A photographic image retrieval system.* Ph.D. thesis, School of computing, Engineering and Technology, University of Sutherland, UK.

M.S. Lew. 2000. Next generation web searches for visual content. *Computer*, 33(11):46–53.

P. Lyman and H.R. Varian. 2000. How much information. http://www.sims.berkeley.edu/research/projects/how-much-info.

J.J. McCann, S.P. McKee, and Taylor. 1976.

Quantitative studies in retinex theory. *Vision Research*, 16:445–458.

J.C. Meadows. 1974. Disturbed perception of colours associated with localized cerebral lesions. *Brain*, 97:615–632.

Y. Rui, T.S. Huang, M. Ortega, and S. Mehrotra. 1998. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions on circuits and systems for video technology*, 8(5):644–655.

O Sacks. 1995. *An anthropologist on Mars*. New York: Knopf.

L. Schomaker, L. Vuurpijl, and E. de Leau. 1999. New use for the pen: outline-based image queries. In *Fifth International Conference on Document Analysis and Recognition*, pages 293–296. IEEE.

A. Schulz. 1998. *Interface design - Die visuelle gestaltung interaktiver computeranwendungen*. Rohrig Verlag, St. Ingbert.

K.W. Shirriff. 1993. The rbw color model. *Computers & Graphics*, 5(17):597–602.

N. Steiner. 2002. A review of web based color pickers. http://www.webgraphics.com/feature-002.php.

K. Uchikawa and M. Ikeda. 1986. Accuracy of memory for brightness of colored lights measured with successive comparison method. *Journal of the Optical Society of America*, 3(1):34–39.

R. Veltkamp and M. Tanase. 2000. Content-based image retrieval systems: A survey. Technical report, Department of Computing Science, Utrecht University. http://www.aa-lab.cs.uu.nl/cbirsurvey/cbir-survey/.

R. Veltkamp, H. Burkhardt, and H. Kriegel. 2001. *State-of-the-Art in Content-Based Image and Video Retrieval*. Kluwer Academic Publishers.

C.C. Venters and M. Cooper. 2000. A review of content-based image retrieval systems. internal report, JTAP. http://www.jtap.ac.uk/reports/htm/jtap-054.html.

L. Vuurpijl, L. Schomaker, and E. van den Broek. 2002. Vind(x): Using the user through cooperative annotation. In *Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition*, pages 221–226.

B.L. Whorf, 1956. *Language, thought, and reality: Selected papers of Benjamin Lee Whorf*, pages 233–245. Cambridge, M.A.: MIT Press. [Original work published 1941].

Combining Morphological and Ngram Evidence for Monolingual Document Retrieval

Jaap Kamps and **Christof Monz** and **Maarten de Rijke**

Language & Inference Technology Group, ILLC, U. of Amsterdam
Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands
E-mail: {kamps,christof,mdr}@science.uva.nl

Abstract

We report on experiments in which we merged the results of linguistically informed and linguistically ignorant approaches to retrieval for European languages. We found that even high-quality base runs can be improved by means of fairly simple techniques for merging them with other runs, although the improvements no longer seem to be as dramatic as those reported on previous experiments on smaller collections than we used and with retrieval engines that are not as highly optimized as the one used in our experiments.

1 Introduction

It's a widely held belief that deep linguistic analysis does more harm than it helps in document retrieval (Lewis and Sparck Jones, 1996). Morphology seems to provide the level of analysis that is appropriate for document retrieval. Especially for non-English European languages there is evidence that linguistically informed morphological analyses helps improve effectiveness. For instance, in combination with lexical-based stemming compound splitting improves retrieval effectiveness for Dutch and German (Monz and de Rijke, 2002). Unfortunately, for many European languages other than English, lexical resources are hard to obtain or even non-existent. For this reason, various teams working in document retrieval for such languages have developed language independent morphological normalization tools, often based on ngrams (CLEF, 2002).

Rather than choosing for either linguistically motivated morphological approaches or linguistically ignorant ngram-based approaches for retrieval for European languages, our strategy is to merge the results of the two approaches. Assuming that high-quality morphological and ngram-based runs identify mostly the same relevant documents, but different non-relevant documents, such combinations should yield improvements in retrieval effectiveness over both base runs.

In this paper we report on experiments carried out in monolingual document retrieval for Dutch, French, German, Italian, and Spanish, using the collections and assessments made available in the CLEF evaluation campaign (CLEF, 2002). We found that even high-quality base runs can be improved by means of fairly simple techniques for merging them with other runs, although the improvements no longer seem to be as dramatic as those reported on previous experiments on smaller collections than we used and with retrieval engines that are not as highly optimized as the one used in our experiments. The parameters that we used to create the optimal combination of runs are collection dependent but they do seem to be fairly robust across topics.

2 Experimental Setup

All experiments were carried out with the **FlexIR** system developed at the University of Amsterdam (van Hage et al., 2002). **FlexIR** is a vector-space retrieval system; for the experiments for this paper it was used with the Lnu.ltc weighting scheme and with blind feedback turned on.

Table 1 lists the characteristics of the document collections that we used for the experiments in this paper. They are part of the document collections made available within the CLEF campaign.[1]

The topics used in the experiments were Topics 41–90 and 91–140; these were the topics used at CLEF-2001 and CLEF-2002, respectively. For evaluation purposes we used the qrels provided by the CLEF organizers.

3 Three Types of Runs

For Dutch, French, German, Italian, and Spanish we created three types of runs: morphological, ngram-based and merged.

3.1 Morphological Runs

The three main morphological phenomena, i.e., inflection, derivation, and compound words, all affect

[1] As of 2002, CLEF also includes Finnish and Swedish in the monolingual track. Unfortunately, we did not have access to (linguistically informed) morphological normalization tools for these languages.

Language	Collection	Year	Documents	Size (in MB)
Dutch	Algemeen Dagblad	1994/1995	106,483	241
	NRC Handelsblad	1994/1995	84,121	299
French	Le Monde	1994	44,013	157
	SDA French	1994	43,178	86
German	Der Spiegel	1994/1995	13,979	63
	Frankfurter Rundschau	1994	139,715	320
	SDA German	1994	71,677	144
Italian	La Stampa	1994	58,051	193
	SDA Italian	1994	50,527	85
Spanish	Agencia EFE	1994	215,738	509

Table 1: The document collections used.

the effectiveness of text retrieval. Documents are not retrieved if the search key does not occur in the index. For effective retrieval morphological processing is needed in most languages to handle inflected word forms. The morphological normalization may be stemming or lemmatization. In *stemming* affixes are removed from word forms (Porter, 1980); the output is a common root or stem of different forms, which is not necessarily a real word. In (lexicon-based) lemmatization word forms are turned into base forms which are real words. Morphological analysis also allows one to split compounds into their component words.

For each of the languages we used a lexical-based stemmer, or lemmatizer, where available. For Dutch we used MBLEM, a memory-based lemmatizer developed at Tilburg University (van den Bosch and Daelemans, 1999); for French, German, Italian we used TreeTagger (Schmid, 1994), and for Spanish we used a Porter stemmer (CLEF-Neuchâtel, 2002).

For Dutch and German we complemented our lemmatizers with a compound splitter to analyze complex words such as *Autobahnraststätte* (English: highway restaurant) and *Vredesverdrag* (English: peace agreement). In addition to these noun-noun compounds there are several other forms of compounding, including verb-noun (e.g., German: *Tankstelle*, English: gas station), verb-verb (e.g., German: *spazierengehen*, English: taking a walk), noun-adjective (e.g., Dutch: *werkeloos*, English: unemployed), etc.. We used simple compound dictionaries, that consist of complex words and their parts, where each part is lemmatized; see (Monz and de Rijke, 2002) for further details.

For retrieval purposes, each document in the Dutch and German collections is analyzed and if a compound is identified, both the compound and all of its parts are added to the document. Compounds occurring in a query are analyzed in a similar way: the parts are simply added to the query, while keeping the compound.

3.2 Runs Based on Ngrams

In information retrieval ngrams have become a popular technique for identifying index terms; see, e.g., (Mayfield and McNamee, 1999; Savoy, 2001) for some recent examples of systems using ngrams. We fixed the ngram length to be the largest integer smaller than the average word length. For Dutch, German, Italian, and Spanish we used ngram length 5, and for French we used ngram length 4; see Table 2 for an overview. For each word we stored both the word itself and all possible ngrams that can be obtained from it without crossing word boundaries. For instance, the Dutch version of Topic 108 contains the phrase *maatschappelijke gevolgen* (English: societal consequences); using ngrams of length 5, this becomes:

> *maatschappelijke maats aatsc atsch tscha schap chapp happe appel ppeli pelij elijk lijke gevolgen gevol evolg volge olgen*

Some authors adopt ngram-based approaches in which ngrams are allowed to span word boundaries; see e.g., (McNamee and Mayfield, 2002). We did not find any consistent significant improvements in allowing ngrams to cross word boundaries, and stuck to our present set-up for reasons of space efficiency.

Stopword removal was done before ngrams were formed; we determined the 400 most frequent words, then removed from this list content words that we felt might be important despite their high frequency. We did not use a 'stop ngram' list. Diacritic characters were not replaced by the corresponding non-diacritic letters.

3.3 Merging Runs

We merged our morphological and ngram-based base runs in the following manner. First, we normalized the retrieval status values (RSVs), since different runs may have radically different RSVs. For each run we reranked these values in $[0.5, 1.0]$,

	Dutch	French	German	Italian	Spanish
Avg. word length	5.4	4.8	5.8	5.1	5.1
Ngram length	5	4	5	5	5

Table 2: Average word length and ngram length used for the ngram base runs.

using

$$RSV_i' = 0.5 + 0.5 \cdot \frac{RSV_i - min_i}{max_i - min_i},$$

and assigned the value 0.5 to documents not occurring in the top 1000; this is a variation of the Min_Max_Norm considered by Lee (Lee, 1997a). Next, we assigned new weights to the documents using a linear interpolation factor λ representing the relative weight of a run:

$$RSV_{new} = \lambda \cdot RSV_1 + (1 - \lambda) \cdot RSV_2.$$

For $\lambda = 0.5$ this is similar to the summation function used by (Fox and Shaw, 1994; Belkin et al., 1995; Lee, 1997a).

Table 3 lists our non-interpolated average precision scores for CLEF 2002, for the morphological and ngram-based base runs, and for the merged runs. The figures in brackets indicate the improvement of the merged run over the best underlying base run. For all languages, the merged run outperforms the underlying base runs. Moreover, these improvements occur at all recall levels, as illustrated by the P/R plots for German (CLEF 2002) in Figure 1. However, the relative improvements are far less dramatic than the 25% improvements reported in the literature (Lee, 1997b; Lee, 1997a), which were obtained using low-quality runs (by today's standards).

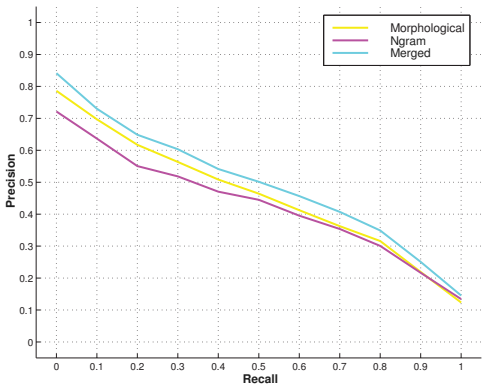


Figure 1: 11pt interpolated avg. precision for German, using the CLEF 2002 topics.

The optimal interpolation factors λ were obtained experimentally. Figure 2 suggests that the optimal interpolation is very stable across topic sets.[2]

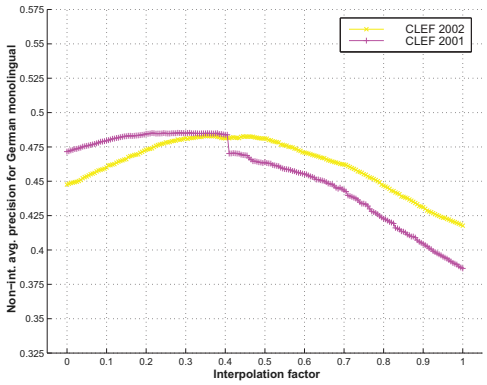


Figure 2: The interpolation factor λ. Effect on non-interpolated avg. precision scores for German, at CLEF 2001 and 2002, where $\lambda \in [0, 1]$.

Figure 3 shows that λ can be chosen from a broad interval of values without dramatic penalties in terms of non-interpolated avg. precision scores.

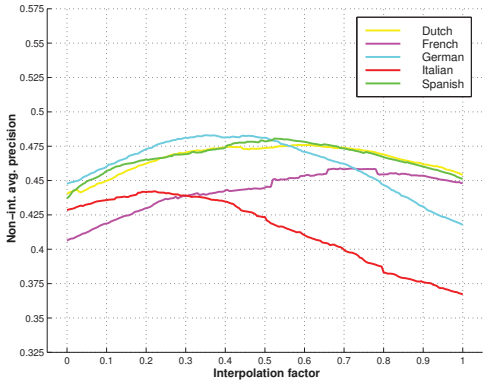


Figure 3: The interpolation factor λ. Effect on non-interpolated avg. precision scores for Dutch, French, German, Italian, and Spanish at CLEF 2002, with $\lambda \in [0, 1]$.

[2] Note that there is a marked discontinuity in the CLEF 2001 curve at 0.4; we have observed similar — but less dramatic — drops in curves for other merged runs, but have not found an unequivocal explanation yet.

	Dutch	French	German	Italian	Spanish
Morphological	0.4404	0.4063	0.4476	0.4285	0.4370
Ngram	0.4542	0.4481	0.4177	0.3672	0.4512
Merged	0.4760	0.4589	0.4830	0.4422	0.4806
	(+4.8%)	(+2.4%)	(+7.9%)	(+3.2%)	(+6.5%)

Table 3: Non-interpolated average precision scores for CLEF 2002.

4 Discussion

The following rationale has been put forward for combining (high quality) runs: try to maximize the overlap of relevant documents between the base runs, while minimizing the overlap of non-relevant documents (Lee, 1997a); this way, the RSVs of relevant documents should get a boost, but those of non-relevant documents not. The following coefficients $R_{overlap}$ and $N_{overlap}$ have been proposed for determining the overlap between two runs run_1 and run_2:

$$R_{overlap} = \frac{R_C \times 2}{R_1 + R_2} \qquad N_{overlap} = \frac{N_C \times 2}{N_1 + N_2},$$

where R_C (N_C) is the number of common relevant (non-relevant) documents, and R_i (N_i) is the number of relevant (non-relevant) documents in run_i ($i = 1,2$). (A document is relevant if its relevance score in the qrels provided by CLEF is equal to 1.)

Table 4 shows the overlap coefficients for the base runs used to produce merged runs; the coefficients are computed over all topics.

Contrary to Lee (Lee, 1997a)'s rationale, for our high quality base runs there does not seem to be an obvious correlation between the overlap coefficients and the improvements obtained by combining them.

5 Conclusions

We reported on experiments in which we merged the results of linguistically informed and linguistically ignorant approaches to retrieval for European languages. We found that even high-quality base runs can be improved by means of fairly simple techniques for merging them with other runs, although the improvements no longer seem to be as dramatic as those reported in the literature.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback, and Vera Hollink for her technical support. Jaap Kamps was supported by the Netherlands Organization for Scientific Research (NWO), grant # 400-20-036. Christof Monz was supported by the Physical Sciences Council with financial support from NWO, project 612-13-001. Maarten de Rijke was supported by grants from NWO, under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 220-80-001, and 612.000.207.

References

N.J. Belkin, P. Kantor, E.A. Fox, and J.A. Shaw. 1995. Combining evidence of multiple query representations for information retrieval. *Information Processing & Management*, 31(3):431–448.

CLEF-Neuchâtel. 2002. CLEF resources at the University of Neuchâtel. `http://www.unine.ch/info/clef`.

CLEF. 2002. Cross-Language Evaluation Forum. `http://www.clef-campaign.org`.

E.A. Fox and J.A. Shaw. 1994. Combination of multiple searches. In *Proceedings TREC-2*, pages 243–252.

J.H. Lee. 1997a. Analyses of multiple evidence combination. In *Proceedings SIGIR'97*, pages 267–276.

J.H. Lee. 1997b. Combining multiple evidence from different relevant feedback networks. In *Database Systems for Advanced Applications*, pages 421–430.

D.D. Lewis and K. Sparck Jones. 1996. Natural language processing for information retrieval. *Communications of the ACM*, 39(1):92–101.

J. Mayfield and P. McNamee. 1999. Indexing using both n-grams and words. In E.M. Voorhees and D.K. Harman, editors, *The Seventh Text REtrieval Conference (TREC-7)*, pages 419–423. NIST Special Publication 500-242.

P. McNamee and J. Mayfield. 2002. Scalable multilingual information access. In C. Peters, editor, *Working Notes for the the CLEF 2002 Workshop*, pages 133–140.

C. Monz and M. de Rijke. 2002. Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck, editors, *Proceedings CLEF 2001*, LNCS 2406, pages 262–277. Springer Verlag.

M. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

J. Savoy. 2001. Report on CLEF-2001 experiments. In C. Peters, editor, *Working Notes for the CLEF 2001 Workshop*, pages 11–20. ERCIM-01-W04.

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*.

A. van den Bosch and W. Daelemans. 1999.

	Dutch	French	German	Italian	Spanish
$R_{overlap}$	0.9443	0.9606	0.9207	0.9021	0.9172
$N_{overlap}$	0.3790	0.5187	0.4180	0.4510	0.5264

Table 4: Degree of overlap among relevant and non-relevant documents for the base runs used to form the merged runs. The coefficients are computed over all CLEF 2002 topics.

Memory-based morphological analysis. In *Proceedings ACL'99*, pages 285–292.

W. van Hage, V. Hollink, J. Kamps, C. Monz, and M. de Rijke. 2002. The **FlexIR** information retrieval system. Manual, Language & Inference Technology Group, ILLC, U. of Amsterdam.

AlphaDMax^{TM}: an integrated tool for biomedical information retrieval and extraction

Kristof Van Belleghem
PharmaDM
Ambachtenlaan 54/D
B-3001 Leuven, Belgium
kristof.vanbelleghem@pharmadm.com

André Vandecandelaere
PharmaDM
Ambachtenlaan 54/D
B-3001 Leuven, Belgium
andre.vandecandelaere@pharmadm.com

Luc Dehaspe
PharmaDM
Ambachtenlaan 54/D
B-3001 Leuven, Belgium
luc.dehaspe@pharmadm.com

Abstract

Automated access to information in free-style text documents involves two main steps: (1) the relevant documents are collected from a repository of documents (information retrieval), and (2) information of interest is extracted from the text. AlphaDMax^{TM}, a text mining tool for the biomedical literature, relies on the PubMed search engine for document retrieval, and on a combination of co-occurrence analysis and natural language processing for the extraction of relationships between concepts of interest. Concepts and relations of interest are specified by means of search templates that are supported by appropriate ontologies. Ontologies for proteins, small molecules, biological processes, diseases and biological locations have been constructed based on the Medical Subject Headings (MeSH) by the National Library of Medicine (NLM). User-defined ontologies or the Gene Ontology (GO) can be loaded as well. Extracted relationships can be incorporated in a data warehouse for further data mining. The results are visualised within a data cube.

Introduction

The biomedical literature is a cornucopia of information that can be used to complement or validate experimental data. Accessing this information involves two main steps: (1) the relevant documents are collected from a repository of documents (information retrieval), and (2) information of interest is extracted from the text (Information Extraction). In order to achieve this, traditional retrieval packages rely on keyword searches and combinations of search terms ("bag-of-words" approach) without taking the relationships between search terms as explicitly stated in the text into account (see e.g. Salton (1989), Berry *et. al.* (1995), Andrade and Valencia (1998)). This typically results in high recall but poor precision. In contrast, approaches based on Natural Language Processing (NLP) incorporate text structure (syntactic and semantic structure) in the extraction process, thus, allowing for more accurate detection of relationships (Allen (1995), Brill and Mooney (1997), Craven and Kumlien (1999)). NLP approaches are becoming increasingly popular. Nevertheless, NLP often has a hard time in dealing with the complex structure of sentences and terms in biomedical abstracts. In keeping with current trends, AlphaDMax^{TM} therefore uses a combination of NLP and co-occurrence, supported by a domain-specific knowledge base, to extract user-specified information from free-style text documents (see e.g. Tanabe *et al.* (2002), Krauthammer *et al.* (2002) for recent applications of a hybrid approach).

1 Methods

The overall approach of AlphaDMax™ to text mining is illustrated in Figure 1. Given a subject of interest AlphaDMax™ retrieves the relevant documents from an appropriate free-text document repository. Next, retrieved documents are submitted to a shallow syntax parser (Daelemans *et al.* (1999), Buchholz *et al.* (1999)). AlphaDMax™ uses a combination of NLP techniques with co-occurrence analysis and domain specific synonym lists and ontologies in order to extract the relationships that match the relation(s) specified in the query template. Articles are ranked according to relevance with regard to the particular query, thus, improving precision. The extracted information can be integrated and visualised within the data cube defined by the involved ontologies.

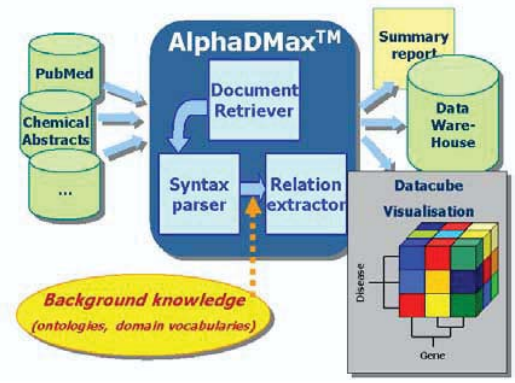


Figure 1: The AlphaDMax™ approach to textmining. (see text for details)

Data cubes allow for a comprehensive view on the extracted information. Upon request, the system also generates a summary report that lists the relevance ranking of the articles together with an overview table of extracted relations. Finally, the extracted relationships can be stored in a data warehouse where they can be combined with other (experimental) data and accessed by data mining engines[1].

[1] AlphaDMax™ can be used as a stand-alone tool or in combination with PharmaDM's DMax™ relational data mining system.

1.1 Information retrieval

AlphaDMax™ is currently geared towards searching MEDLINE[2] and uses the existing PubMed[3] search engine for document retrieval. If connected to the internet, the system sends retrieval requests to PubMed. Consequently, the recall of AlphaDMax™ is equal to the recall of PubMed, and the system does not improve on the existing search engine.
Retrieval queries to PubMed can be stored for off-line or future use. If a local repository of MEDLINE abstracts is available, PubMed queries are emulated for the retrieval of the documents from that repository. The ranking of the documents is as returned by PubMed.

1.2 Information extraction

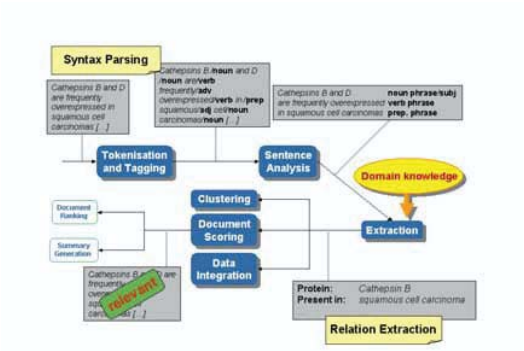


Figure 2: The information extraction process. (see text for details)

The information extraction process is illustrated in figure 2. The extraction is supported by a general purpose shallow syntax parser based on a k-nearest neighbour algorithm and trained on a Wall Street Journal corpus[4] (Daelemans *et al.* (1999), Buchholz *et al.* (2001)). Parsing involves sentence recognition, tokenisation, "part-of-speech" tagging and sentence analysis. Thus, the sentence "*Cathepsins B and D are frequently*

[2] MEDLINE is the repository of abstracts from biomedical journals of the U.S. National Library of Medicine (http://www.nlm.nih.gov/)

[3] PubMed can be accessed at http://www.ncbi.nlm.nih.gov/

[4] The used syntax parser was developed by Walter Daelemans and colleagues (University of Tilburg and University of Antwerp) and provided by Textkernel BV, Amsterdam, The Netherlands

overexpressed in squamous cell carcinomas" is transformed into "*{Cathepsins B/noun and D/noun}noun phrase/subject { are/verb frequently/adv overexpressed/verb}verb phrase {in/prep squamous/adjectiv cell/noun carcinomas/noun}prepositional phrase*". Given a query specified by the user, such as

<protein><expressed in><biological location>

during the actual extraction process the system looks for instances of relations, i.e. relationships, that comply to the pattern

<protein/subj><express/verb><location/prepositional phrase>.

Applied to the example, substituting "protein" by "cathepsin B" and "cathepsin D", "express" by "overexpressed", and "location" by "squamous cell carcinomas" matches the query pattern, hence, the relationships

express(cathepsin B, squamous cell carcinoma)

and

express(cathepsin D, squamous cell carcinoma)

are recognised by the system. Because the syntactic information as revealed by the shallow syntax parser is used for the extraction of these relationships, we refer to the pattern matches that identified them as NLP matches. Since shallow parsing does not always results in correctly parsed sentences, we use co-occurrence analysis as a fall-back method in cases were no NLP match was found. By default, NLP matches are rated higher than co-occurrence matches, however, the user can modify the absolute and relative weights of both types of matches.

In order to improve the quality of parsing, domain-specific vocabularies are used to ensure that typical biomedical terms are detected and interpreted correctly. Domain knowledge in the form of designated ontologies is further used during the information extraction. Public domain ontologies as well as user-built ontologies can be loaded into AlphaDMax^{TM}.

The extracted information can be used to score the documents, or it can be integrated with data from other sources (e.g. experimental data), or clustered in order to reveal potentially interesting combinations of e.g. proteins and diseases. Documents are scored based on the number and the quality of extracted relationships. Given the scores for each document, the documents are ranked according to their relevance with regard to the query, and a summary report of the extracted relationships can be produced.

In order to improve response times, documents can be pre-processed prior to retrieval and stored in a repository of parsed documents.

2 Results and Discussion

As a first application of AlphaDMax^{TM} we have searched a database of approx. 70000 MEDLINE abstracts on human proteins for two types of relations: (1) an "influence" relation, and (2) a "present in" relation. The "Influence" relation can be formulated as a pattern <actor><influence><object> in which actor and object can be substituted by a gene, a protein, a transcript, a small molecule, a biological process or a disease. The "presence" relation was treated as a generalisation of the "express" relation of section 1.2 with a similar query pattern. Ontologies for proteins, genes, transcripts, small molecules, biological processes, diseases, and biological locations were constructed out of the Medical Subject Headings (MeSH)[5].

As mentioned in the methods section we distinguished between NLP and co-occurrence matches. We also considered partial NLP matches i.e. matches in which a syntactic match was found for only two of the three members in the patterns. We assessed the performance of AlphaDMax^{TM} using four sample sets of abstracts dealing with: kinesin (44 retrieved abstracts), porin (102 retrieved abstracts), cathepsin (225 retrieved abstracts) and aromatase (123 retrieved abstracts). These subjects were chosen arbitrarily, except for the fact that the total number of abstracts remained

[5] The 'Medical Subject Headings' (MeSH) is an ontology that is developed and maintained by the National Library of Medicine

manageable for a human domain expert, and each set yielded a balanced number of retrieved relationships for "influence" queries and "presence" queries. (In other words, we did not use test sets that yielded a fair amount of relationships for "influence" queries but few for "presence" queries, and vice versa.) In general, approximately 6% of the relationships were identified by NLP, 82% by co-occurrence analysis and 12% by mixed NLP-co-occurrence. Thus, NLP significantly reduces the recall. On the other hand, more than 80% of the relationships identified by NLP were correct as opposed to 25% of the relationships identified by co-occurrence analysis. The recall of NLP in the current version of AlphaDMax^{TM} is likely to be improved by retraining the syntax parser on a Medline corpus (work in progress in collaboration with Walter Daelemans), and an extension of the domain vocabulary to include all known MeSH terms. (For reasons of efficiency, only terms from the so-called descriptor records have been incorporated in the vocabulary.)

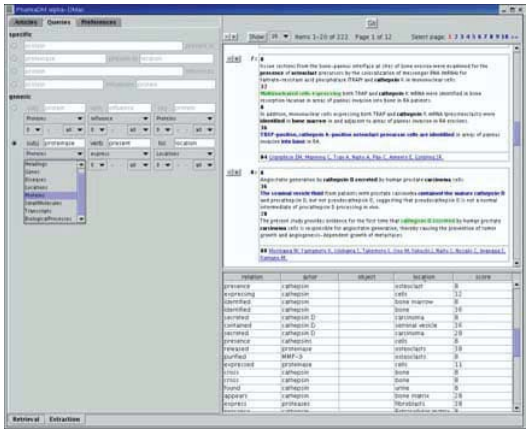


Figure 3: Detailed visualisation of extraction results. Shown are some results of the query <proteinase><present><location> for the retrieved set of abstracts that deal with cathepsin. NLP matches are in blue, partial NLP matches in green and co-occurrence matches in bold. The score of each sentence is indicated in front.

The results can be visualised in several manners, allowing for views that range from a summary to the details of single sentences. Figure 3 shows an example of detailed output. The sentences that match the query are displayed together with their score and the abstract they belong to. The elements in each sentence that match the pattern are highlighted using different colours for the different types of match. This enables the user to judge the performance of the system for this particular query at a glance. It is possible to drill down further to reveal all abstract details including the full text and the parsed sentences.

We are currently working on a more flexible visualisation of the results in AlphaDMax^{TM} by means of data cubes. According to the data cube principle, the ontologies that are involved in the query can be considered as the dimensions of a n-dimensional space, with n the number of ontologies. Thus, the results of the query <protein><influences><disease> fit within the 3-dimensional space defined by the Protein, Disease and Influence ontologies. Using the drill down, roll up and slice operations that are defined on data cubes, within this space the user can navigate to inspect the results at levels of detail ranging from a general summary to single documents or relations. Both retrieval results and extraction results can be visualised in respectively a retrieval cube and an extraction cube. Figure 4 shows part of the data cube for the retrieval of documents that relate proteins to diseases. In the illustration, the retrieval cube has been rolled up to rather general levels of diseases and proteins. Each dimension can separately be drilled down into further detail.

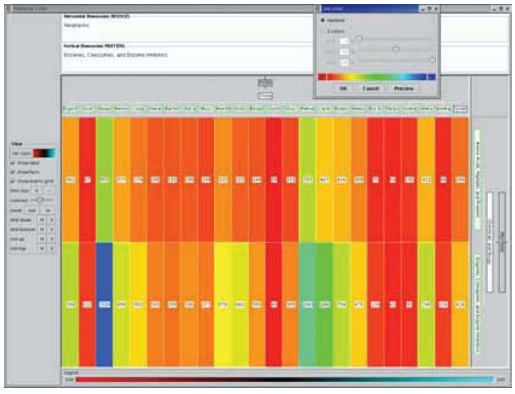


Figure4: Representing retrieved documents and extracted relationships within a data cube. Shown is a subsection of the retrieval cube of abstracts dealing with proteins and diseases.

Each cell in the cube contains the abstracts that deal with the combination of the corresponding disease(s) and protein(s) as given by the ontologies along the two axes. The number of documents is indicated in each cell. Furthermore, each cell is colour coded ranging from red (a low incidence of abstracts) to blue (a high incidence of abstracts). In this example the blue cell corresponds to 2214 abstracts that deal with the neoplasms in the disease ontology and the "enzymes, coenzymes and enzyme inhibitors" branch of the protein ontology.

The following numbers are indicative of the volume of articles that can be processed. These numbers are obtained on an AMD Athlon(TM) XP 1600+ processor, with 500M of RAM assigned to the application. All articles are retrieved from a repository of pre-parsed abstracts (parsing one abstract currently takes up to 20 seconds, so we rely on preprocessing wherever possible). Applying the most general presence query, <protein><present in><location> on a set of 1000 abstracts takes 1 minute and 45 seconds if the abstracts have not previously been loaded, and 1 minute sharp if they have already been loaded for earlier queries. For the most general influence query, these times are respectively 2 minutes and 15 seconds, and 1 minute and 20 seconds ('influence' queries takes longer because the 'influence' ontology is more extensive). These numbers scale slightly better than linear up to approximately 2000-2500 articles. At higher numbers of articles the performance starts to suffer from the 500M memory limit. Raising this limit has a slightly better than linear effect on the number of articles that can be processed without performance loss. Note that up till now we have not yet spent a lot of effort in the performance optimisation. Hence, there still is a margin for improvement.

Conclusion

AlphaDMax^{TM} is a proprietary text mining tool for the biomedical literature developed by PharmaDM. It uses a combination of specialised search engines, natural language processing (NLP) technology and co-occurrence analysis to solving to the problem of extracting information of interest from a repository of free-style text documents. By using a combination of NLP and co-occurrence analysis, AlphaDMax^{TM} is able to return not only a set of extracted relationships, but, in addition, estimates of how correct these relationships can be expected to be.

Acknowledgements

We thank Walter Daelemans (UIA, University of Antwerp, Wilrijk, Belgium) and Jakub Zavrel (Textkernel BV, Amsterdam, The Netherlands) for helpful discussions on syntax parsing.

References

Allen J. (1995) *Natural Language Understanding.* The Benjamin/Cummings Publishing Company, Inc., New York.

Andrade M. A. and Valencia A. (1998) *Automatic extraction of keywords from scientific text.* Bioinformatics, 14/7, pp. 600-607.

Berry M. W., Dumais S. T., and O'Brien G. W. (1995) *Using Linear Algebra for Intelligent Information Retrieval.* SIAM Review 37/4, pp. 573-595.

Brill E. and Mooney R. J. (1997) *An overview of Empirical Natural Language Processing.* AI magazine, 18/4, pp. 13-24.

Buchholz, S., Veenstra J., and Daelemans W. (1999). *Cascaded Grammatical Relation Assignment.* In proceedings of EMNLP/VLC-99, University of Maryland, USA, pp. 239-246.

Craven M. and Kumlien J. (1999) *Constructing Biological Knowledge Bases by Extracting Information from Text Sources.* Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology, pp. 77-86.

Daelemans W., Buchholz S., and Veenstra J. (1999) *Memory-Based Shallow Parsing.* In: "Proceedings of CoNLL-99, Bergen, Norway", pp. 53-60.

Krauthammer M., Kra. P., Iossifov I., Gomez S. M., Hripcsak G., Hatzivassiloglou V., Friedman C., and Rzhetsky A. *Of truth and pathways: chasing bits of information through myriads of articles.* Bioinformatics, 18/S1, pp. S249 S257.

Salton G. (1989) *Automatic Text Processing: The transformation Analysis and Retrieval of Information by Computer.* Addison Wesley Publishing Company, Reading, MA.

Tanabe L. and Wilbur W. J. (2002) *Tagging gene and protein names in biomedical text.* Bioinformatics, 18/8, pp. 1124-1132.

A study about synonym replacement in news corpus

Roxana Angheluta, Marie-Francine Moens
roxana.angheluta@law.kuleuven.ac.be

ICRI - Interdisciplinary Center for Law and Information Technology
Katholieke Universiteit Leuven
Belgium

1 Introduction

An important issue in many fields dealing with text processing (retrieval, summarization, etc) is synonym recognition. In information retrieval, for example, the query can be augmented with synonym words in order to improve the recall of the retrieved articles. In summarization, the synonyms are to be used for detection of redundant passages in a text. The fact that many words are polysemous adds difficulty to the task, involving word sense disambiguation. While in the case of retrieval, the lack of the context for the query words makes their disambiguation quite difficult, in the case of summarization the words are surrounded by context, which theoretically should help in the process of synonym detection. Having a resource with synonyms, the task is to detect synonym words from the text, taking care of the correct sense of the word. In fact the problem can be seen as a problem of disambiguation using a thesaurus.

2 Methods

We used WordNet [7] as the back resource with synonyms. Senses in the WordNet database are represented relationally by synonym sets ('synsets') which are the sets of all the words sharing a common sense. Words of the same category are linked through semantic relations like hyponymy (more specific terms), hypernymy (more general terms), etc. Polysemous words appear in more than one synset. Each synset has a gloss associated. Using the hyponymy/hypernymy network in WordNet (see figure 1; in the brackets is the sense of the first word from the synset), one can compute the relatedness between each pair of synsets as the length of the path between them. It is a classical search problem, resolved by us with iterative deepening algorithm, with a maximum search depth of <MAX_DEPTH>, i.e. words farther away than <MAX_DEPTH> were considered not related. We allowed only certain paths between words, taken from literature [6] (e.g. only one change of direction, where hypernymy=upward relation, hyponymy=downward relation) . We used a binary measure of relatedness.

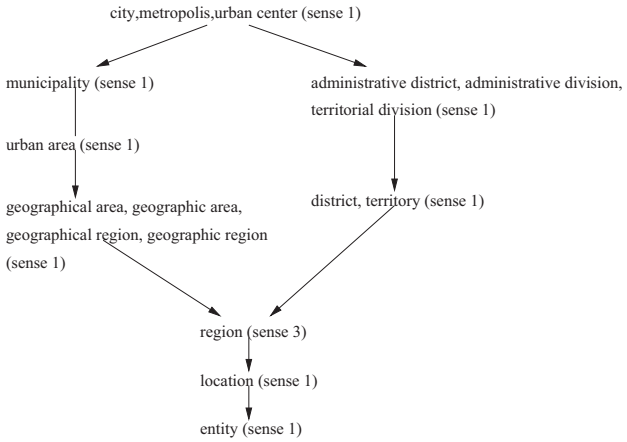


Figure 1: Hypernyms of the first sense of the word *city*

For detection of synonyms we used the algorithm for lexical chains construction described in [1], restricted just to the relation of synonymy. The notion of cohesion was introduced by Halliday and Hasan [3] in 1976 as a device for "sticking together" different parts of text. It is achieved through the use of semantically related terms, reference, ellipsis and conjunctions. The most easily identifiable and the most frequent type is lexical cohesion, created by using semantically related words. An example of a lexical chain is: *technology-science-technology-computer_science-ai-intelligence-field*. The relatedness between chain members is not well defined, but one can agree that at least synonym words should belong to the same chain. Lexical chains have been used for information retrieval [5], for correction of malapropism [6] and for summarization [1], [2].

We consider just the nouns from the text, extracted with a tagger [4]. They are stemmed using the morphological module of WordNet. The algorithm works in two phases: words disambiguation and chain construction (see figure 2).

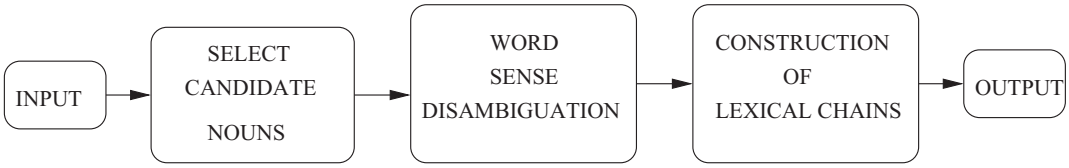


Figure 2: The algorithm for constructing the chains

2.1 Disambiguation

For the word sense disambiguation, we used the notion of component. A component is a cluster of related words which contains the possible combinations of senses for the words inside (a vector of synsets), called interpretations. Each interpretation has a power associated, which is the number of links between its synsets. The interpretations are ordered by descending power. When a new word is processed, it is tried to be put in one of the existing components, if it is related with one of the words from that component. Otherwise a new component is created.

```
process(<new_word>){
inserted=FALSE
 for each existing component <comp>
  for each word <word> from <comp>
   if <new_word> is related with <word>
   then
    insert <new_word> into <comp>
    inserted=TRUE
   endif
  endfor
 endfor
if(inserted==FALSE)
then
 create_a_new_component(<new_word>)
endif
}
```

Two words are considered related when at least one sense of the first word is related with a sense of the second word, using the WordNet network. Each time a new word is added in a component, the number of interpretations for that component is updated and the weak ones (the last <no_interpretation>-<MAX_NO_INTERPRETATIONS> in the ordered list) are removed. The maximum number of allowed interpretations <MAX_NO_INTERPRETATIONS> in a component is given as a parameter.

An example of two possible interpretations associated with a component is presented in figure 3.

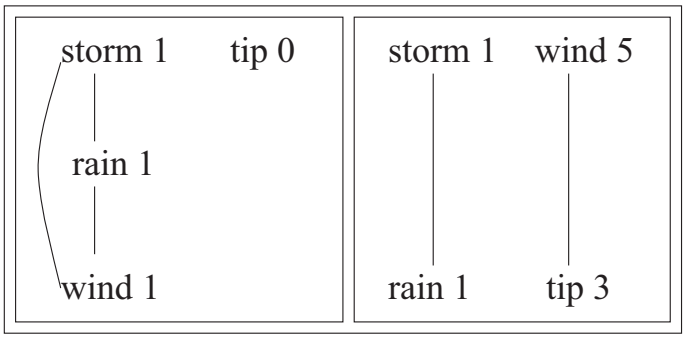


Figure 3: An example of two interpretations associated with a component

In the end, after all the words are processed, each word is assigned the sense from the most powerful interpretation from the component it belongs to.

As one can notice from the above mentioned algorithm, the disambiguation is based on all the words inside a component. The algorithm has a greedy aspect, because each word is assigned to the first component with which the word is related. It is thought not completely greedy, because for each word a number of senses (implicitly set by the parameter <MAX_NO_INTERPRETATIONS>) is kept till the end, when the strongest sense is chosen. An exhaustive comparison between each pair of words in a text is unrealistic for large texts.

2.2 Chain construction

In our implementation, the chains will consist of synonym words. From each strong interpretation of each component, the synonym words are selected in a vector.

The greedy aspect of the algorithm for disambiguation can lead sometimes to the same word in different components. For example, if one has in the text the words: *hurricane,...,wind,...,storm,...wind.* At its first apparition in the text, the word *wind* has no link with any of the existing components (*wind* and *hurricane* are at a distance of 6, so they are considered unrelated), so a new component is created for it. After, the word *storm* is inserted in the component created for *hurricane* (say component 0) and the next apparition of the word *wind* is linked to it because of the relationship with word *storm*. And all the following apparitions of *wind* will be put in component 0. So words tend to agglomerate in the first components. This effect is inversely proportional with the power of relatedness: if the relatedness criterion is very strict (just synonym words are considered as being related), then there will be less cases than when also hypernyms/hyponyms are considered as related words. In order to correct a bit this behavior, the algorithm matches in the end chains that have a word in common.

We considered two experiments:

- <MAX_DEPTH>=0 - consider just synonym words for the disambiguation. Will be further referred as exp1.
- <MAX_DEPTH>=4 - consider related words that are separated by a path of length maximum 4 in WordNet. Will be further referred as exp2.

The first experiment has the advantage of simplicity and speed, while the second one increases the accuracy of the synonym words in the chains.

2.3 Synonym replacement

Each chain of words is assigned a most representative element of the chain, i.e. the first most frequent word from that chain. All the other words from the same chain are replaced in the original text with the most representative element.

3 Evaluation

In order to evaluate the efficiency of the synonym replacement we counted the number of errors obtained after replacement of the words from a chain with the most representative word of that chain. We used news texts from TREC disks employed at DUC 2002. The articles were extracted from: Wall Street Journal 1987-1992, AP newswire 1989-1990, San Jose Mercury News 1991, Financial Times 1991-1994, LA Times from disk 5 and FBIS from disk 5.

We considered a subset of 34 documents (4 clusters of related articles). We did not count capitalization errors, wrong accord and punctuation errors, because they can be easily solved. We only looked closely to changes of meaning due to the placement in the same chain of words that were not synonyms.

There were two types of errors in the construction of the chains, both generated by the same cause (wrong disambiguation):

1. the same word (stem) appears in a chain more than one time, but with different senses. Will be further referred as error1.

2. different unrelated words are put in the same chain. Will be referred as error2.

A few examples of error 2 are presented in figure 4.

Words put in the same chain	Examples of the contexts
eye-C(c)enter	looks like the eye - National Hurricane Center
world-man	beloved by the world - Renaissance man; Rain Man - around the world
history-Story	other niches in history - West Wide Story
people-Mass.	well-known to people - Tanglewood, Mass.;
	moved so many people - Bernstein's 1971 "Mass"
Hispaniola-Haiti	moved south of Hispaniola - heavy rain on ... Haiti
air-lines	whipping paper through the air - downed power lines;
	a considerable amount air time - ads for public condolences at $15.85 a line
death-end	shot to death - the end of the checkpoint
biography-life	in a controversial biography - his personal and professional life
(H)honor-award	Legion of Honor - Peabody award
Marx-Groucho	Groucho Marx
office-roles	at the box office - fulfilled their roles

Figure 4: Errors made for different words considered as synonyms

Sometimes it was difficult to decide if the words replaced were synonyms when they were part of an expression or an idiom. For example, when *secretary of state* was replaced by *secretary of country*, we did not count it as an error. Other difficult cases were abstract terms which are synonyms, like *thing-matter*, *scene-aspect*, but which appear in unrelated parts of the text and should not be put in the same chain, because they do not improve coherence. Following the definition of the lexical chains we still considered them related and we did not count them as an error.

One can notice that we tried to be as undemanding as possible and that the number of errors we counted is actually the minimum number of errors. The number of errors are presented in figure 5.

Sometimes different parts of speech were wrongly tagged as nouns. In the brackets are the real numbers of errors, not caused by the tagger. The percentages represent the fraction of the replacements that belong to the same stem and different stems, respectively, for rows 2 and 4 and the fraction of errors in each of these two categories, for rows 3 and 5. The precision of the algorithm can be measured as 100-%errors.

	exp1	exp2
no_replacements	650(639)	504(500)
no_words_same_stem	395(394) 60.7%(61.6%)	406(405) 80.5%(81%)
error1	17(16) 4.3%(4%)	17(16) 4.1%(3.9%)
no_words_dif_stem	255(245) 39.2%(38.3%)	98(95) 19.4%(19%)
error2	142(132) 55.6%(53.8%)	36(33) 36.7%(34.7%)

Figure 5: Synonym replacement evaluation

4 Discussion of the results and possible improvements

The first type of errors is not very frequent when considering common nouns, as a study made in [8] concludes. Yarowsky counted for a few ambiguous words the percentage with which they appear with the same sense in a document and he got very high figures, which proves that usually a word appears with the same sense throughout a discourse. Notable cases in this category are the elements of proper names, like in *Mexico City* and *city* or *cities*. We did not count these cases as errors, unless the sense was obviously different, like in *National Hurricane Center* and *the center of the storm*, *New York Times* and *New York time*.

The second type of errors appears mainly because of the lack of the context. When disambiguating a word, we compare for each sense the context in which that sense occurs (the rest of the words from the same synset and the words from related synsets) with the context in the text (other nouns in the text). When there is not enough evidence for a certain sense, the word can be wrongly disambiguated. For example, if in the same text the words *side* and *English* appear and no other *side*-related words are encountered (e.g. slope, incline, position, face), then they are considered synonyms, like in the synset:

```
English, side -- ((sports) the spin given
to a ball by striking it on one side
or releasing it with a sharp twist)
```

One can notice that in the second experiment the number of errors for different stems is highly reduced, because we enlarged the context in the dictionary with all words from related synsets. Still, around 66% precision (100-34.7%) is not a very high number and one should experiment carrefully before deciding to detect synonyms based on WordNet.

In the results from figure 5 only the number of errors is computed (and implicitly the precision). We did not measure the recall of the correct synonyms detected. Usually all the words detected as synonyms in the second experiment were also detected in the first one. Sometimes, correct synonyms detected in the first experiment *style-flair*, *candidates-nominees*, *director-conductor* were not detected in the second one, which make us hypothesize that the recall is lower in the second experiment. Probably in most applications the recall is not so important as precision, but one should be aware of the trade-off between these two measures.

Proper names should not be treated in the same way as the common ones. Especially in news corpora, where they appear a lot, they are largely responsible for the first type of errors. Also the use of collocations might increase the accuracy.

It has been mentioned in the literature that WordNet is too fine-grained for being used as a thesaurus. We did not feel this as the main problem for our task, but rather the lack of the context needed for disambiguation. A learning

approach might lead to better results than a thesaurus-based one, but the need of a training corpus counter-balance the advantage of the technique.

5 Conclusions

Although synonym replacement in a text is conceptually a simple problem, there are no simple techniques to resolve it accurately. The bottleneck is due to the necessity of word sense disambiguation, which is a difficult task. In our experiment, we detected synonyms using the algorithm for lexical chains construction, with WordNet as a thesaurus. We counted the number of errors introduced by placement in the same chain of words that were not synonyms. The results showed that while in the case of the same stem, the error rate was quite low, in the case of different stems it was much higher. If one would like to use WordNet as a thesaurus for synonym replacement, one should be careful not to introduce more noise and to study the influence of wrong disambiguation on the final results.

References

[1] Barzilay R. and Elhadad M. (1997) *Using Lexical Chains for Text Summarization.* http://citeseer.nj.nec.com/324597.html (visited September, 5th, 2002).

[2] Brunn M., Chali Y. and Pinchack C.J. (2001) *Text Summarization Using Lexical Chains.* In Proceedings of Document Understanding Conference, New Orleans, USA, September 13, 2001, pp. 135-140.

[3] Halliday M.A.K. and Hasan R. (1976) *Cohesion in English*, London: Longman.

[4] Mikheev A. (1998) *Part-of-Speech Guessing Rules: Learning and Evaluation.* http://www.ltg.ed.ac.uk/software/pos/ (visited September, 5th, 2002).

[5] Stairmand M. (1996) *A Computational Analysis of Lexical Cohesion with Applications in Information Retrieval.* PhD thesis, Center for Computational Linguistics, UMIST, Manchester.

[6] St-Onge D. (1995) *Detecting and Correcting Malapropisms with Lexical Chains.* Department of Computer Science, University of Toronto, Toronto, Canada. http://www.cs.toronto.edu/compling/Publications/Abstracts/Theses/StOnge-thabs.html (visited September, 5th, 2002).

[7] Miller G.A., Beckwith R., Fellbaum C., Gross D. and Miller K.J. (1990) *Introduction to WordNet: An on-line Lexical Database.* International Journal of Lexicography (special issue), 3(4):235-312.

[8] Yarowsky D. (1995) *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods.* In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics. Cambridge, MA, pp. 189-196.

Creating Document Surrogates with Lexical Cohesion

Drs. Bas van Gils
University of Nijmegen
Bas.vanGils@cs.kun.nl

Dr. Hans Paijmans
Tilburg University
J.J.Paijmans@uvt.nl

November 18, 2002

1 Introduction

The Field of Information Retrieval (IR) traditionally deals with representation, storage, organization and finding information items in (large) collections, and has been extensively studied in e.g. (Rijsbergen, 1979; Salton and McGill, 1983; Baeza-Yates and Ribeiro-Neto, 1999). Closely related to Information Retrieval are Text Classification and Text Categorization: all three largely depend on the identification of features that can be used as keywords to be included in a query or classificator. For the purpose of our paper, we will focus on the *indexing process*, or rather on the creation of so-called Document Surrogates[1].

Deciding which parts of the text are good candidates for the indexing process is a complex task, for items in the index must not only reflect the 'aboutness' of the represented documents as well as possible, but they also can be part of classificators that divide the group of documents in two classes: relevant and not-relevant with respect to a query (which is not necessarily the same). A large part of this 'aboutness' is captured by the words used in the text (the *lexicon*). In (Morris and Hirst, 1991; Kozima, 1993) it is explained that a text is more than a random sequence of words, it has a coherent structure. Our hypothesis is that this structure can be used in the creation of a document representation. More specifically, we try to use *lexical cohesiveness* of (expository) text for this purpose (Morris and Hirst, 1991).

In text, lexical cohesion (Lcoh) is the result of chains of related words (or phrases) that contribute to the continuity of lexical meaning. A 'count' of the number of active chains is taken to be a metric for the (amount of) cohesiveness of text. A variation of this concept, in combination with the $tf.idf$ weights of the occurring words, has been exploited by e.g. Hearst (Hearst and Schütze, 1993) to detect topical shifts in text.

In other publications (e.g. (Paijmans, 1994) (Paijmans, 1997)) we have already proposed that in documents exist so-called *Gravity Wells of meaning*; passages that contain more information about the topic of the document than other passages, and that such passages are marked by certain properties of the text, such as position, or the occurrence of *cue words or - phrases* or other surface properties (see also (Paice, 1990)).

The goal of our recent experiments is to find out whether passages with a higher (or lower) degree of Lcoh are better document surrogates than randomly chosen passages.

[1] For the sake of clarity: we define the actual input to the indexing process to be the document surrogate; the result of this process is the document representation.

Preliminary experiments, in which we compared extracts created by Microsoft Word with extract based on lexical cohesion, showed a performance that was as good or better than the Word extracts, when compared with human performance[2].

2 Setup of the experiments

At this moment we are still working with a dataset of 200 LaTeX documents, collected from the arXiv-website[3]. These documents are evenly distributed over the subjects *Computer Science* (CS) and *Astrophysics* (APH). The main reason for using LaTeX documents is the fact that they are easily parsed. Furthermore, these documents are longer than the newspaper articles that are customarily used in experiments of this kind (e.g. the REUTERS corpus), and well structured (in the sense that they are divided in sections, subsections etcetera). Hence, they are good representatives for the scientific and scholarly documents in the retrieval of which we are naturally interested. Nevertheless we ran into serious problems with this dataset, which we wil describe later.

A first naive approach is to get rid of the LaTeX codes in the file and work on the remaining text. However, this leaves the problem of (lists of) mathematical equations, graphics and so on. Furthermore, since the authors of the documents differ, the internal layout of the files differ as well. For example, some authors tend to wrap their lines at 80 characters, whereas others put the entire paragraph on a single line. As the grammatical sentence is one of the units that we wanted to compute the Lcoh over, we had to write a program to normalize for these differences.

In the current version of our program, we ignore big mathematical environments and graphics, but keep inline maths. We try to normalize the text so that every sentence is on a single line, with an efficiency of approximately 80%. The errors that we still have in our normalization occur mainly with references such as "...In Fig. 3 ...", and bibliographical references; we will fix such problems as we go along. Maths are a problem in itself. Documents that did not display a coherent text after 'detexing' were skipped, so that the final database consisted of 200 documents, 100 from each class. As measure for performance we take the number of documents that were correctly classified.

2.1 The classi cator

The primary setup of the experiments is that of a classification task within the vector Space Model. Word-document weights are computed with the atc-variant of Smart. The classificator then is created automatically by computing the centroid of the positive examples in the database. Classification finally is obtained by comparing this centroid or a derivative of it with all document vectors. In our experience the centroid of the atc values of a class is not a particular good classificator, but in this database it performed very well, perhaps too well: between 90 and 98%. This happened regardless of the similarity function that was used to compute the difference between each document and this classificator; only the dice coëfficent scored lower. Superficial checks of the documents gave no obvious reasons other than the fact that the documents covered very different topics,

[2] Unpublished "onderzoeksproject" by L. Flinkenflögel, N. Konings and C. Koolen, computational linguistic students at Tilburg

[3] `http://lanl.arXiv.org`

but we will have to go into this problem again, and perhaps compile a totally dfferent database, before we draw any final conclusions.

In any case we are not interested in absolute performance, but in the differences in performance between text segments with low or high Lcoh and random segments.

2.2 Chaining

Another program that we wrote, *chains*, computes for every sentence or text window the number of active word chains, i.e. words that reoccur within a certain number of units. Such units are either sentences or windows of an equal number of tokens. The actual words that are taken into consideration can be controlled in several ways, including stopwords and lists of synonyms. Because of the many possibilities, finding the optimal algorithm for chaining is a problem in itself. For the first series of experiments we did not use the possibility of synonym lists, but concentrated on *identity of reference* or literal strings, using as units either complete sentences or word windows. As chaining with only *identity of reference* is easy to implement (but gives mediocre results), it offers a good opportunity to test our apparatus and general concepts.

So, which lexial cohesion properties we should expect in our *gravity wells* compared to the rest of the document?

- A difference with respect to non-function words. Intuitively one would assume that an author is more 'focussed' when he is describing concepts that are central to his discourse. This would lead to less but longer chains of content-bearing words and to a lower Lcoh score for such passages when the count is done with short chains.
- Also, one would expect these words to contain more information, which we can measure with e.g. the $tf.idf$.

3 Evaluation

The goal of our experiments is to find out whether document surrogates with an atypical number of active lexical chains are better document surrogates than randomly chosen passages (of the same length). Hence, we start experiments by creating three classes of document surrogates: the full text of the documents, parts selected according to Lcoh cohesion algorithms (the 'selection') and finally document surrogates that consist of random passages ('random'), but of a length equal to the selection files. The complete database is only used for reference; the real work is done with the 'selection' and the 'random' databases.

The first check of differences between the two databases gave promising results. We created a selection with low Lcoh and replaced every word in this set and in the random set by its atc value (as computed over the complete database). The average $tf.idf$ of the words in the selection was consistently a few percents lower than that in the random database. This seems to point to a situation where the author uses less different words in 'interesting' passages that are central to his discourse.

The second measure of succes would be that the document fragments selected by looking for passages with a different degree of lexical cohesion, would be easier to classify into the original classes than those of the the random database. To our chagrin, correct

classification was constantly very high and showed no actual difference between the two databases.

Former work (Apté et al., 1994), (Paijmans, 1998) had shown us that classification could be improved by using so-called *local dictionaries*. In that case only a very small number of keywords (typically between 50 and 100) is selected from the centroid and the rest is zeroed. This added one more technique to our arsenal. With this 'local dictionaries' variant of the centroid, the results changed somewhat, in that differences began to show between the selection and the random database. However, these differences were not consistent.

4 Conclusions

The first and most urgent conclusion of our work so far is that we may have selected the wrong database to experiment with. As we already mentioned, the experimental text databases that have been used by the IR community are almost exclusively short newsfeed documents, such as the Reuter collection or the documents that come with TREC. We were not able to find a publicized database of longer documents that were in a usable format and pre-classified. Creating a database of our own, and cleaning it, brings many unexpected problems and questions, not the least being the question how to handle maths and other "fremdkörper" in IR.

The experiments so far seem to suggest that lexical cohesion may be one of the surface properties of text that indicate emphasis on the topic or 'aboutness' of a document. The evidence is as yet very weak, and only expressed in differing $tf.idf$ values. We hope to get a better grip on the classification experiments after we have included synonym lists and lists of related terms, so that the reiteration by semantic relations can also be measured.

References

Apté, C., Damerau, F., and Weiss, S. M. (1994). Toward language independent automated learning of text categorization models. In *zSIGIR94*. To appear.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley. ISBN: 0-201-39829-X.

Hearst, M. A. and Schütze, H. (1993). Customizing a lexicon to better suit a computational task. In *Proc. ACL SIGLEX Work. Acquisition of Lexical Knowledge from Text*, Columbus, Ohio.

Kozima, H. (1993). Text segmentation based on similarity between words. In *Meeting of the Association for Computational Linguistics*, pages 286–288.

Morris, J. and Hirst, G. (1991). Lexical cohesions computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):1991, 21–48.

Paice, C. D. (1990). Constructing literature abstracts by computer: techniques and prospects. *Information processing and management*, 26(1):171–186.

Paijmans, J. J. (1994). Relative weights of words in documents. In Noordman, L. G. M. and de Vroomen, W. A. M., editors, *Conference proceedings of STINFON*, pages 195–208. StinfoN.

Paijmans, J. J. (1997). Gravity wells of meaning: detecting information-rich passages in scientific texts. *Journal of Documentation*, 53(5):520–536.

Paijmans, J. J. (1998). Text categorization as an information retrieval task. *South African Computer Journal*, (21):4–15.

Rijsbergen, C. v. (1979). *Information Retrieval*. Butterworths, London, United Kingdom, 2nd edition.

Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, New York.

Taming Wild Phrases

C.H.A. Koster and M. Seutter
Computing Science Institute,
University of Nijmegen,
The Netherlands,
E-mail: kees@cs.kun.nl

DIR '02 workshop

Abstract

In this paper we compare the suitability of different document representations for automatic document classification, investigating a whole range of representations between bag-of-words and bag-of-phrases. We look at some of their statistical properties, and determine for each representation the optimal choice of classification parameters and the effect of Term Selection.

Phrases are represented by an abstraction called Head/Modifier pairs. Rather than just throwing phrases and keywords together, we shall start with pure HM pairs and gradually add more keywords to the document representation. We use the classification on keywords as the baseline, which we compare with the contribution of the pure HM pairs to classification accuracy, and the incremental contributions from heads and modifiers. Finally, we measure the accuracy achieved with all words and all HM pairs combined, which turns out to be only marginally above the baseline.

We conclude that even the most careful term selection cannot overcome the differences in Document Frequency between phrases and words, and propose the use of term clustering to make phrases work.

Anyhow, you've been warned and I will not be blamed
If your Wild Strawberry cannot be tamed..
– Shel Silverstein, "A Light in the Attic", Harper & Row, 1981

1 Introduction

Over the last decades, many researchers in Information Retrieval have tried to combine keywords with phrases, extracted from documents by linguistic or statistical techniques, in order to raise the accuracy of Retrieval (for an overview see [Strzalkowski, 1999]). Little is known about the best way to combine phrases and words in one language model, but the common approach followed in query-based Information Retrieval is to *add* the phrases to the words rather than to *replace* the words by phrases.

Just adding phrases (or collocations) as terms besides keywords has led to disappointingly small improvements [Fagan, 1988, Lewis and Croft, 1990]. This is commonly attributed to the fact that phrases have a distribution over documents which is very different from that of (key)words. Moreover it is obvious that a (composed) phrase is statistically correlated with its components, which may violate assumptions of statistical independence. At any rate, the improvements gained by using more precise terms (phrases) may well be offset by a loss in recall.

In this paper, we compare the suitability of different document representations for automatic document classification, investigating a whole range of representations between bag-of-words and bag-of-phrases. Being aware of the fact that different representations may need different classification parameters, we determine the optimal tuning and Term Selection parameters for each representation.

Text categorization is a wonderful area for performing experiments in Information Retrieval: given the availability of large labeled corpora and the high performance of modern classification engines, it is a simple matter to measure the way in which the Accuracy achieved depends on the parameter settings and the choice of document representation. In traditional query-based Information Retrieval, doing such controlled experiments is much harder and costlier.

1.1 HM pairs

We are investigating the effect of using linguistically motivated terms (phrases) in Information Retrieval, particularly in Text Categorization. Following the example of many earlier authors (e.g. [Fagan, 1988, Lewis and Croft, 1990, Strzalkowski, 1992, Ruge, 1992, Evans and Lefferts, 1994, Lin, 1995]), these will be represented by *Head/Modifier pairs* (HM pairs) of the form

```
[ head, modifier]
```

where the head and the modifier are (possibly empty) strings of words, usually one word. A *pure HM pair* is one where head and modifier are not empty. There may also be HM pairs with an empty modifier (only a head), or with an empty head (only a modifier).

As an example, the phrase "new walking shoes" is first transduced to the HM tree `[[shoes > walking] > new]` and then unnested to one of the following:

- pure HM pairs

```
[ shoes, walking ]
[ shoes, new ]
```

- HM pairs including single heads

```
[ shoes ]
[ shoes, walking ]
[ shoes, new ]
```

- idem plus single modifiers

```
[ shoes ]
[ shoes, walking ]
[ shoes, new ]
[ walking ]
[ new ]
```

- pairs combined with (a well-chosen subset of) all words.

The Accuracy achieved by using these different representations will be investigated in 4.

1.2 The EP4IR parser/transducer

The EP4IR parser/transducer is freely available under the GPL/LGPL license. It is generated from the EP4IR grammar and lexicon by means of the AGFL system.

Being especially directed towards IR applications, the EP4IR grammar does not set out to give a linguistically impeccable "account" of all English sentences, but it describes mainly the Noun Phrase (NP), including its adjuncts, and the various forms of the Verb Phrase (VP), consisting of the application of a certain verbal part to certain noun phrases (NP's) which occur as its complements. These phrases are transduced into HM pairs, in the process performing certain *syntactic and morphological normalizations*: elements of the phrase are selected, reordered and grouped. Furthermore, NP's not covered by a VP are also extracted.

The transformations are purely syntactic, i.e. they take no other information into account than the grammar, the lexicon and the input. In some cases this may result in linguistically suspect interpretations.

The main practical limitations of the grammar however are caused by lexical and syntactic ambiguity: Even though no contextual or semantic information is available, the parser has to arrive at one parsing for each fragment of the input – and, as far as possible, the most plausible one. To this end, and in order to achieve robustness, a penalty mechanism is used (to be replaced in the near future by probabilistic parsing). Even so, it is easy to find examples where the parser does not succeed in finding the right interpretation.

Precision and Recall of the EP4IR version used in the experiments are barely satisfactory, between .6 and .7 according to measurements. In interpreting the following experiments it should be kept in mind that the linguistic resources are not perfect – phrases are missed or mangled. But rather than waiting for perfect resources, we started experimenting with the available ones.

1.3 About the classification engine

The classification engine LCS used in the experiment implements versions of Winnow [Dagan et al, 1997] and Rocchio [Rocchio, 1971], with a number of Term Selection algorithms and automatic Threshold Selection.

2 Statistics of phrases

According to the literature, the improvement in precision and/or recall obtained by using phrases as terms in retrieval and classification has repeatedly been found disappointing. There is a common feeling that "the statistics of phrases are wrong".

A moment's thought gives support to the idea that the statistical distribution of HM pairs (pairs of keywords) is definitely different from that of the keywords themselves: according to a well-known folklore law in corpus linguistics, in any sufficiently long text, the number of words occurring precisely once (*hapaxes*) is about 40%; therefore the expected percentage of pairs occurring precisely once is $1 - (1 - 0.4)^2 = 64\%$. In this section we shall compare the statistics of words and HM pairs in various ways.

2.1 The corpus

Our corpus, EPO1A, consists of 16000 abstracts of patent applications in English from the European Patent Office, with an average length of 143 words (see [Krier and Zaccà, 2001, Koster et al, 2001]). From this corpus we used 4 subsets of 4000 documents each, chosen at random, in a four-fold cross-validation (training on each of the subsets while using the union of the other three as test set).

The documents are only lightly preprocessed: de-capitalization and elimination of certain characters for the keywords, but no stemming. For the HM pair representation they are completely parsed and the resulting trees unnested, also without lemmatization.

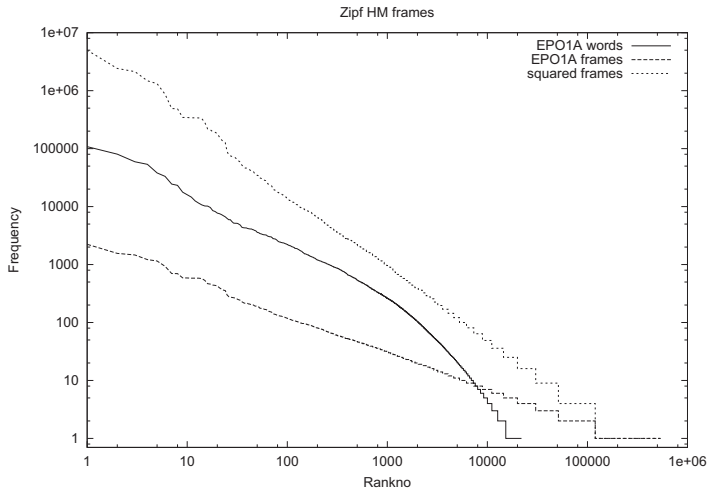
As a measure of the Accuracy, we take the micro-averaged F1-value (a kind of average between Precision and Recall). In the EPO1A corpus, all classes have the same size, so that the differences between macro- and micro-averaging are small.

2.2 Playing Zipf

Using the EP4IR parser/transducer, each of the EPO1A documents has been parsed and transduced to a bag of unnested HM pairs. We omit the phrases with an empty modifier (i.e., consisting of only one word) to avoid all overlap with the bag of words. This provides us with another representation of the same 16000 documents, with the following statistics:

corpus id	total terms	different terms	total size (bytes)
EPO1A pairs	921466	541642	20302454
EPO1A words	2004011	21921	12069192

It appears that the number of HM pairs is about half the number of words, but there are 25 times as many *different* HM pairs. The average word frequency in EPO1A is about 9, the average HM pair frequency is about 2. The statistics of words and phrases are definitely different, which may well explain the bad experiences reported in literature. This also becomes clear from a comparison of the Zipf curves (frequency of words, pairs and, for comparison, squared frequency of pairs):



According to Zipf's law, on a log-log scale the relation between the frequency of a word in a corpus and its rank (ordering the words by frequency) looks like a straight line.

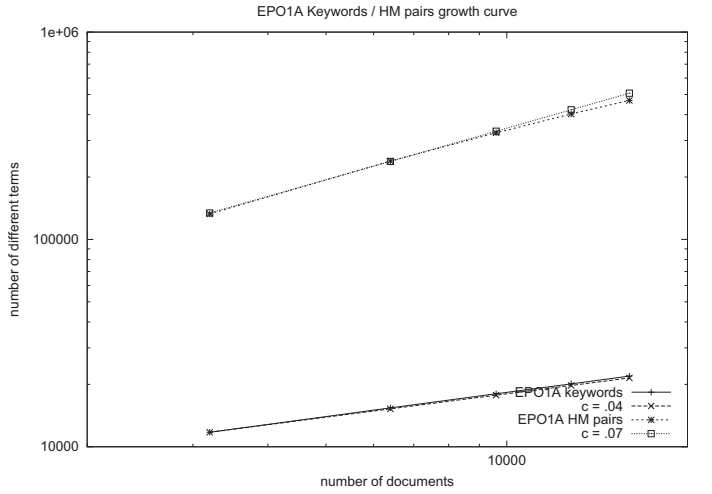
It takes some imagination to call the middle curve a straight line. Keep in mind that the hapaxes are in each case concentrated in the small line at the end of the curve.

Compared to the graph for words, the graph for pairs is ramrod straight and much less steep. Representing a document by HM pairs gives an enormous number of low-frequency terms, among which the significant terms are cunningly hidden. Intuitively, we expect the use of HM pairs as terms in place of words to be beneficial, because some of the HM pairs must be much more indicative of a particular class than the keywords out of which they are composed. But the space of HM pairs is much larger than that of keywords, and therefore much more sparse, to our chagrin.

2.3 Playing Pareto

Another well-known folklore law from IR (and from Corpus Linguistics) states that the number of *different* words in a corpus of n words is proportional to $sqrt(n)$, i.e., quadrupling the size of a corpus doubles the number of words. When the documents are all (about) the same size, the same relation holds with the number of documents. Indeed a *growth curve* showing the number of (key)words against the log of the number of documents show a straight line for the EPO1A corpus.

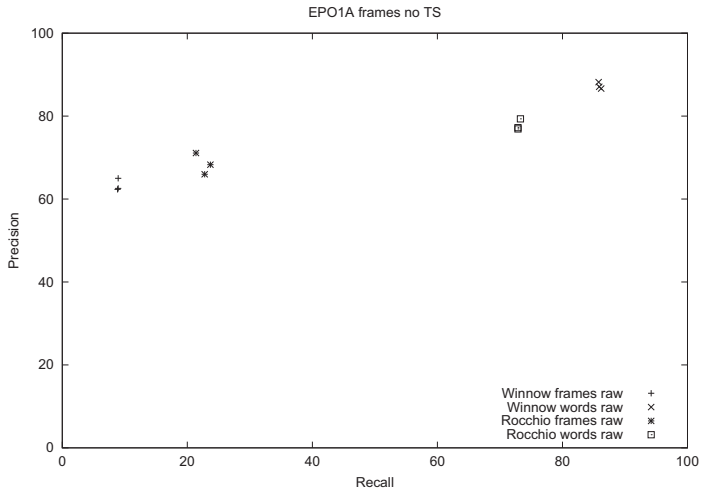
The following graph compares the growth curves for keywords and HM pairs in the EPO1A corpus:



The HM pair line starts much higher and climbs much steeper, spelling no good for the use of HM pairs as terms.

2.4 The problem with phrases

In order to show that the naive use of phrases as terms leads to a disappointing result, we first compare the effect of two different classification algorithms (Winnow and Rocchio) on phrases (represented as HM pairs) and keywords, using standard (default) parameters for the classification algorithms and without performing any term selection at all. Here we do not use four-fold cross validation, but we show individual results, three for each combination.



In comparison to the classification on phrases, words give higher Precision and much higher Recall. We must be doing something wrong in deriving or using the phrases. The fact that Rocchio achieves a higher Precision than Winnow also comes as a surprise, in the light of other experiences [Koster et al, 2001].

3 Improving the statistics

The classification algorithms are subject to noise, because their work is based on term statistics, including very many irrelevant terms, and because of the imperfect labeling of the documents. When eliminating irrelevant terms by Term Selection, we expect not only increased performance but also increased accuracy (see [Yiming and Pedersen, 1997, Peters and Koster, 2002])

In order to get the best Accuracy out of different document representations, we may also have to adapt some classification parameters to the representation. On the basis of extensive experiments, we found three parameter settings to be crucial:

- the Rocchio parameters Beta and Gamma
- the Winnow parameters for the Thick Threshold heuristic
- the choice of Term Selection and in particular the number of terms per class.

We shall first show that by an optimal choice of these parameters, the Accuracy is remarkably improved, even for the baseline (keyword representation). In the following chapter we shall do the same analysis for the other representations. It will turn out that the optimal values are in fact not strongly dependent on the representation, but that they differ quite a lot from their usual values in literature.

3.1 Common parameters

Here we discuss the parameters of the classification engine and briefly explain the algorithms.

```
MinRanks        0
MaxRanks        3
```

Although the train files are mono-labeled, we allow the classifier to assign multiple classes (between 0 and 3), in order to be prepared for multi-classification. The usual document length normalization and sub-linear frequency weighting are used:

```
Normalize       cosine
Strength        ltc
```

3.2 Winnow and its parameters

The Balanced Winnow algorithm [1, Dagan et al, 1997] is a child of the Perceptron. For every class c and for every term t two weights W_t^+ and W_t^- are kept. The score of a document d for a class c is computed as

$$SCORE(c,d) = \sum_{t \in d} (W_{t,c}^+ - W_{t,c}^-) \times s(t,d)$$

where $s(t,d)$ is the (ltc normalized) strength of the term t in d. A document d belongs to a class c if $SCORE(d,c) > \theta$, where the threshold θ is usually taken to be 1.

Winnow learns multiplicatively, driven by mistakes, one document at a time: When a train document belonging to some class c scores below θ, the weights of its terms t in W_t^+ are multiplied by a constant $\alpha > 1$ and those in W_t^- multiplied by $\beta < 1$; and conversely for documents *not* belonging to c which score above θ.

The default values for the Winnow parameters are (following [Dagan et al, 1997]):

```
[Winnow]
Alpha           1.1
Beta            0.9
MaxIters        5
```

3.3 Rocchio and its parameters

The Rocchio algorithm [Rocchio, 1971, Cohen and Singer, 1999] computes for each class c a weight for each feature (another word for term) by

$$w(t,c) = max(0, \frac{\beta}{|D_c|} \sum_{d \in D_c} s(f,d) - \frac{\gamma}{|\overline{D}_c|} \sum_{d \in \overline{D}_c} s(f,d))$$

where

- $s(f,d)$ is the normalized strength of the feature f in the document d
- D_c is the set of documents classified as c and $\overline{D}_c$ the set of non-c documents.

The score of a document for a class is the inproduct of the weights of its features times their strength, and a document is assigned to class c if its score for c exceeds a threshold which is computed from the train set (as is done for Winnow).

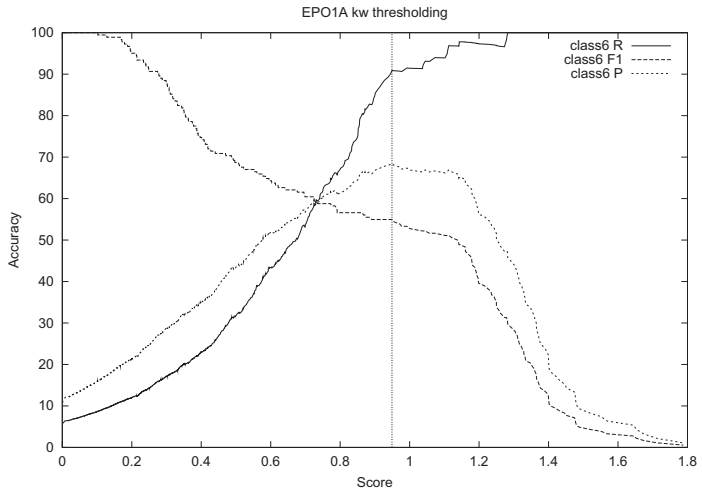
The default Rocchio parameters following [Cohen and Singer, 1999] are:

```
[Rocchio]
Beta            16
Gamma           4
```

We are left with the choice of Term Selection (TS) parameters and the threshold parameters, which may have to be adapted in order to match the different statistics of the terms.

3.4 Thresholding

For each class, after training an optimal threshold is computed, which maximizes the F1 value. Documents will be assigned to each class above whose threshold they score. The following graph shows a typical relation between Precision, Recall and F1-value as a function of score. The vertical bar represents the optimum threshold value (maximizing F1).



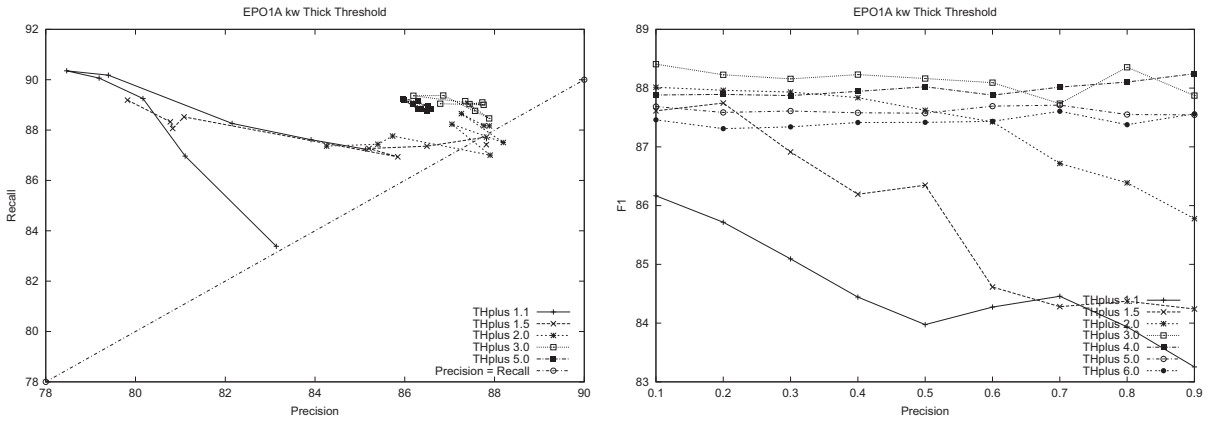
Notice that this optimum is not at the Precision = Recall point, but to its right. At the chosen threshold value, the Precision is systematically larger than the Recall. However, decreasing the threshold until Precision = Recall would not raise the F1-value.

3.5 Tuning Winnow

Phrases do not reoccur as often as words, because there are more of them; that is our main problem. The appearance of a phrase in a document may indicate that it belongs to a certain class, but its absence does not say much. Somehow, we must reward its presence stronger than we deplore its absence.

In the Winnow algorithm, this can be achieved by means of the *thick threshold* heuristic. In training, we try to force the score of relevant documents up above $\theta^+ > 1.0$ (rather than just 1) and irrelevant documents below $\theta^- < 1.0$. This resembles the "query zoning" heuristic for Rocchio, in the sense that documents on the borderline between classes get extra attention.

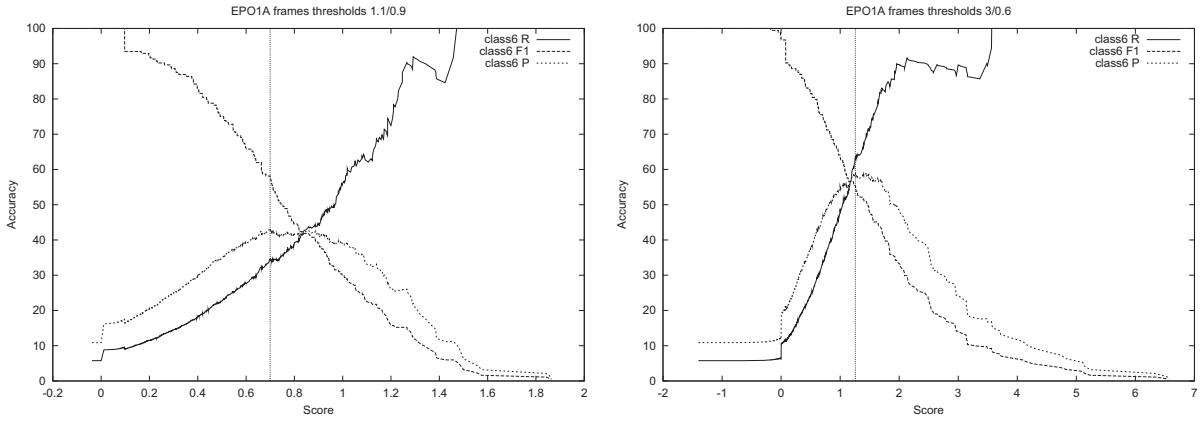
According to [Dagan et al, 1997] the optimal values for these Thick Threshold parameters are 1.1 and 0.9, respectively (just like the Winnow α and β). The following graphs show the effect of the thickness of the threshold for (key)words, our baseline.



The left graph plots Precision against Recall. Each line is a trajectory (going approximately first upwards and then to the right) which represents one value of θ^+C together with values of θ^- going from 0.9 to 0.1. The dotted line represents Precision = Recall.

The right graph is easier to read. The F1-value fluctuates wildly, but by and by an increase of θ^+ improves the Accuracy, and [3.0,0.6] raises the F1-value over [1.1,0.9] by more than 2 points.

The following graphs show the effect of the Thick Threshold heuristic in "broadening" the score distribution; please note the different horizontal scales.

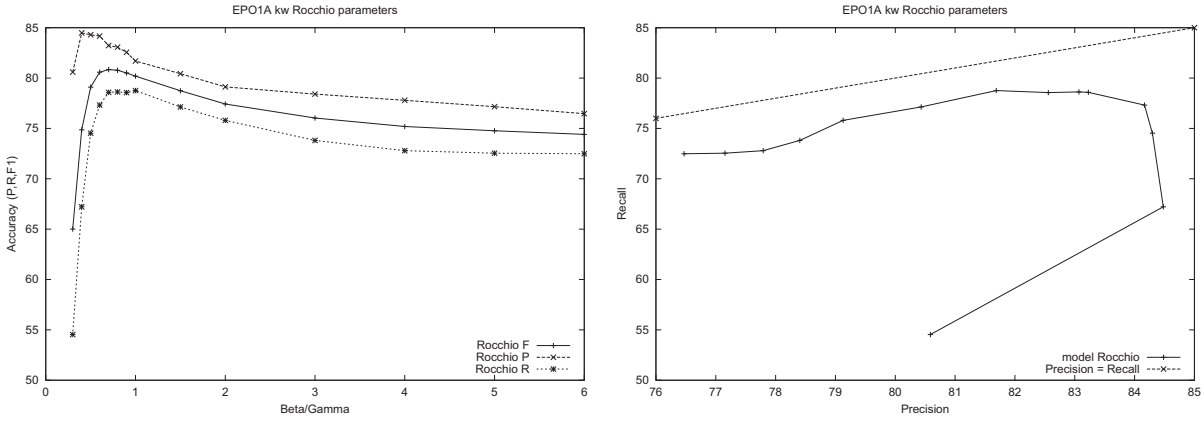


By widening the threshold from 0.9/1.1 to 3/0.6 the Accuracy for this class is raised from .41 to .60; the range where the Accuracy exceeds 70% of the maximum is widened from 0.4:1.2 to 0.5:2.5.

3.6 Tuning Rocchio

The Rocchio parameters β and γ control the relative contribution of the positive and negative examples to the weight vector; standard values in literature are $\beta = 16$ and $\gamma = 4$ [Cohen and Singer, 1999, Caropreso et al, 2000]. Of course this is very strange, because only the *ratio* between β and γ is important for the outcome. The reason for this quaint habit is that there used to be an α term also, representing some initial documents (queries).

In order to tune Rocchio to the base line (keywords), we determine experimentally its Accuracy as a function of the parameter β, keeping $\gamma = 1$ ($\beta = 4$ is the traditional choice, $\beta = \gamma = 1$ was proposed by [Arampatzis et al., 2000a]).



The left graph shows that the optimum is not at $\beta = 4$ but at $\beta = .7$, which means that negative contributions are favoured! Taking the optimal choice of β raises the F1-value from .76 to .81, which is an important improvement.

The right graph shows the *trajectory* followed by Precision and Recall as β decreases from 6 (upper left corner) to .3 (lower middle). Precision is consistently better than Recall.

As was the case for Winnow (3.5), it turns out that a non-traditional choice of parameters for Rocchio leads to a marked improvement of the Accuracy. Obviously, the textbook values are far from optimal!

In the next chapter we shall investigate in how far the optimal choice of parameters depend on the text representation.

3.7 Changing the statistics by Term Selection

Term selection is based on certain statistics of the terms, in particular their distribution over (documents belonging to) the various classes. We expect the classification process to be more accurate when two

kinds of terms are eliminated:

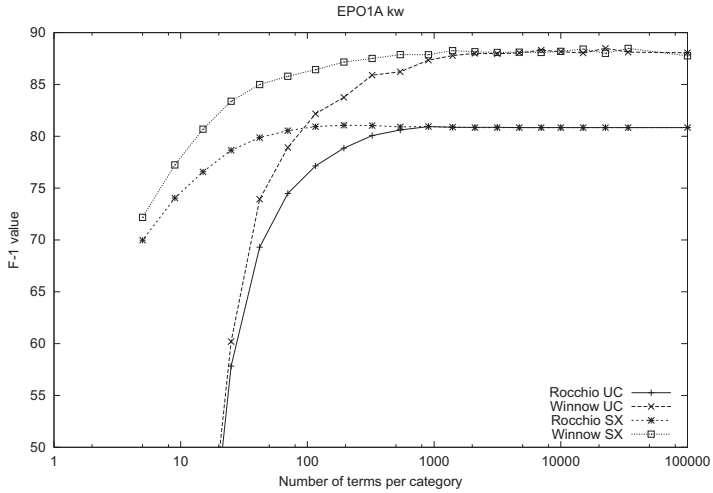
1. stiff terms – terms distributed evenly across documents of all categories, therefore occurring frequently. The traditional stop list is an attempt to eliminate on linguistic grounds the most frequent stiff terms.

2. noisy terms – terms distributed unreliably within a category and between classes. These often have a small frequency, but there are very many of them, causing *dispersion* of the document scores.

A good Term Selection criterium will remove both.

At the optimal values for the Rocchio and Winnow parameters, we apply two local (i.e. category-dependent) TS criteria to find the optimal number of terms per category (i.e. the number or rather range that maximizes Accuracy). Of the six criteria described in [Peters and Koster, 2002], we use only the Uncertainty-based (UC) and Simplified χ^2 (SX) term selection (Document Frequency (DF) is similar to UC, Information Gain (IG) and Term Frequency(TF) are similar to SX).

For our baseline, the keyword representation, Term Selection does not improve the Accuracy, because the optimal choice of parameters has apparently removed most of the unbalance. The experiments described in [Peters and Koster, 2002] were made at textbook values of the parameters.



3.8 Summarizing the baseline

Here we summarize the best results obtained in classifying EPO1A using keywords.

algorithm	method	max F1 value	parameter value
Winnow	raw	.83	$\theta^+ = 1.1, \theta^- = 0.9$
Winnow	tuned	.88	$\theta^+ = 3.0, \theta^- = 0.4$
Winnow	TSel	.88	1400-20000 terms/class
Rocchio	raw	.75	β=4, γ=1
Rocchio	tuned	.81	$\beta = .7$, γ=1
Rocchio	TSel	.81	100-1000 terms/class

At the optimal parameter values, Term Selection makes hardly any improvement to the Accuracy, but the number of terms per class can be quite low without losing Accuracy.

4 Phrases instead of words

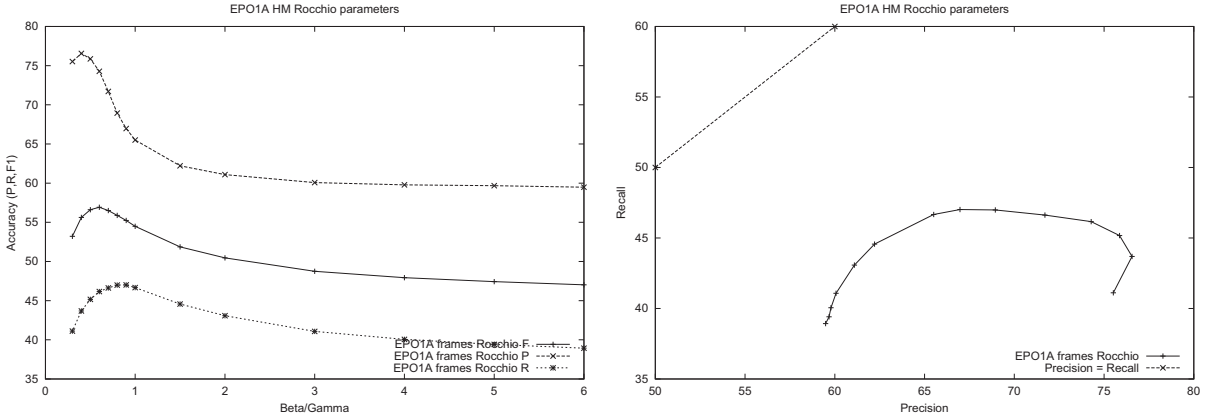
In this chapter we compare the properties of a wide spectrum of text representations, ranging from pure HM pairs (bag-of-phrases representation) to all (key)words (bag-of-words representation).

The baseline which we want to exceed is remarkably high, due to the good statistical properties of the words in the text, even without lemmatization.

4.1 Pure HM pairs

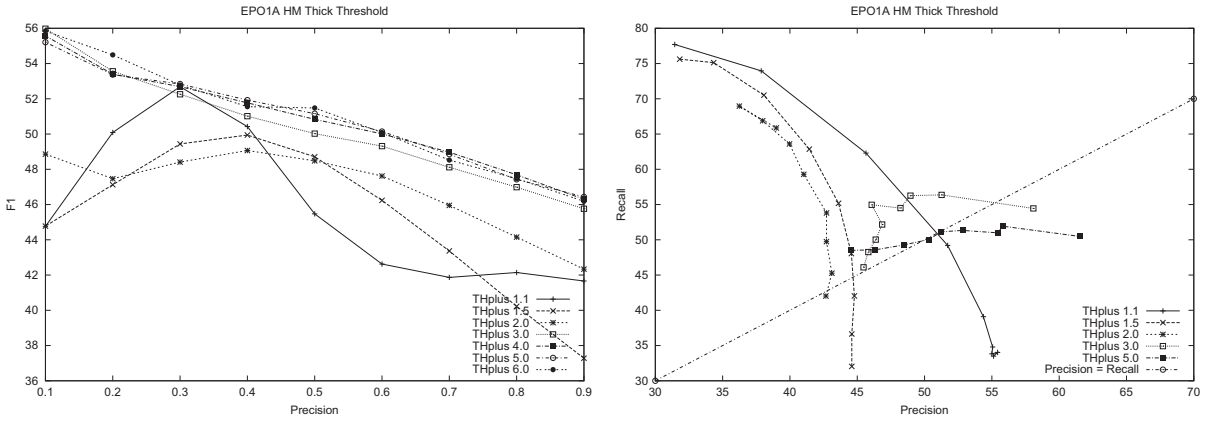
Starting at the extreme end, we investigate the effect of using only "pure" HM pairs, with a nonempty modifier, corresponding to only the composed phrases and to the traditional collocations.

4.1.1 Rocchio parameters



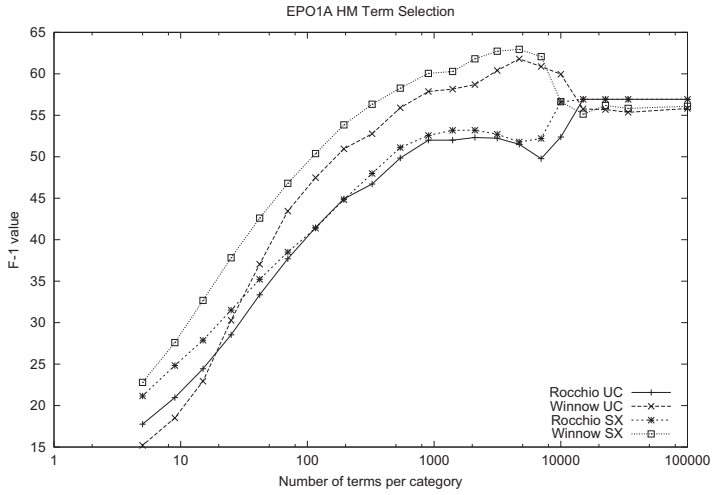
The Accuracy is much lower than for the keyword representation. In particular, the Recall is much less than the Precision, much more so than for keywords. But again the optimum is reached at $\beta = 0.6, \gamma = 1.0$.

4.1.2 Winnow parameters



Again, the Accuracy is much lower than for keywords, indicating that by themselves the "pure" keywords do not offer a good representation. Tuning the thresholds improves the Accuracy by 19% but the optimum is reached at extreme parameter values. Would the choice of a negative value for θ^- be even better?

4.1.3 Term Selection



Without Term Selection Rocchio is slightly more accurate than Winnow. Term Selection with both UC and SX has a rather localized positive effect for Winnow, for Rocchio it leads to a sudden fall which is only partially regained. We have as yet no explanation for this phenomenon (which appeared for different shuffles of the data). Rocchio is better off without Term Selection (!) but Winnow with SX manages to raise the Accuracy by another 7%.

4.1.4 Some statistics

Using all 16000 documents as both train- and test set (testing on seen documents) with Winnow with optimal parameters and Term Selection but first eliminating the hapaxes (MinTF=2), the following results were achieved:

```
Training phase...
  541537 different terms in train set
  119773 global terms in train set
  30289 final terms in train set
Testing phase...
MICRO    15878     95.47%    92.56%    93.99%   Lumped
MACRO    15878     95.46%    92.56%    93.97%   Averaged
```

It is observed that about 421000 of the terms are hapaxes. Of the remaining HM pairs about 90000 are eliminated by term selection. The macro- and microaveraged Precision, Recall and F1 agree closely. F1 is very low considering that we are testing on seen documents. 0f the 16000 train documents only 15878 were used, because the 122 others contained not enough significant terms...

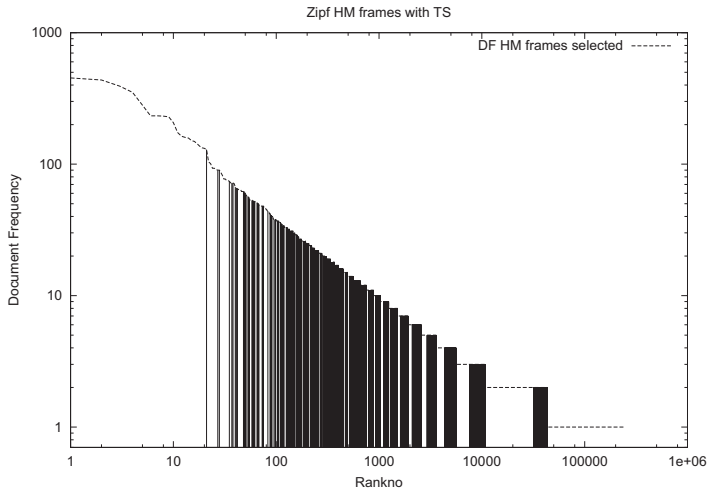
A list of the top 40 terms for one category can be found in Appendix A.1.

4.1.5 Summary HM

Using only the "pure HM pairs, the Accuracy falls far short of the baseline.

algorithm	method	max F1 value	parameter value
Winnow	raw	.37	$\theta^+ = 1.1, \theta^- = 0.9$
Winnow	tuned	.56	$\theta^+ = 3.0, \theta^- = 0.1$
Winnow	TSel	.63	3000-4000 terms/class
Rocchio	raw	.40	β=4, γ=1
Rocchio	tuned	.57	$\beta = .7$, γ=1
Rocchio	TSel	.57	no term selection

The following graph shows for a typical example the selected terms with their document frequencies (DF); both at the high end and at the low end terms have been eliminated.



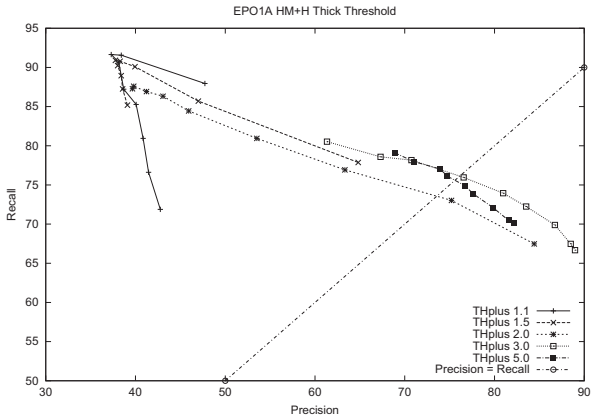
4.2 Phrases plus single heads

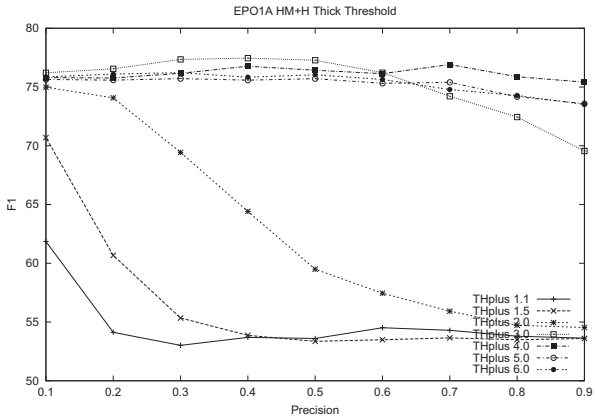
Not al phrases are composed. A non-composed phrase will have an empty modifier. In this section we therefore include the single head taken from each HP pair.

```
[ shoes ]
[ shoes, walking ]
[ shoes, new ]
```

Due to the prominent semantical role of heads, they are very promising classification terms, but we expect them to be less precise than pure HM pairs, and to have higher Document Frequencies so that they might overwhelm the pure HM pairs.

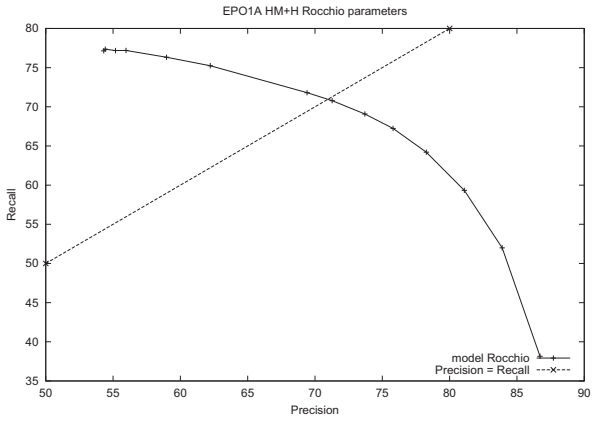
- Winnow parameters

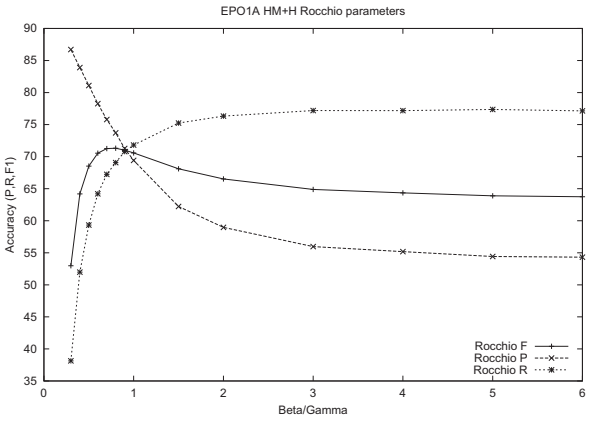




The addition of the heads causes a great improvement in behaviour. The effect of the Thick Threshold heuristic is very large but the optimal choice of the thresholds [3,0.4] is little different from that in 3.5. The trajectory Precision/Recall looks like a hybrid between that of keywords (3.5) and that of pure HM pairs (4.1.2).

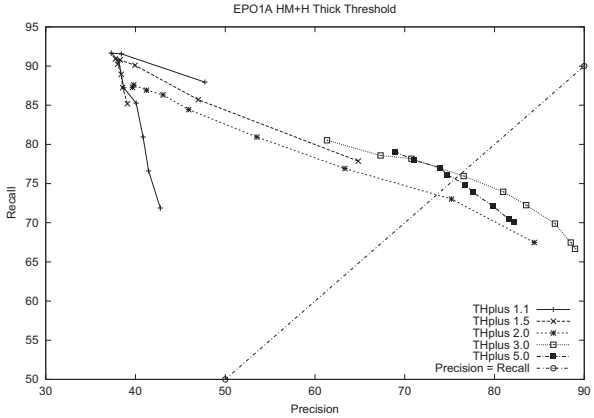
- Rocchio parameters

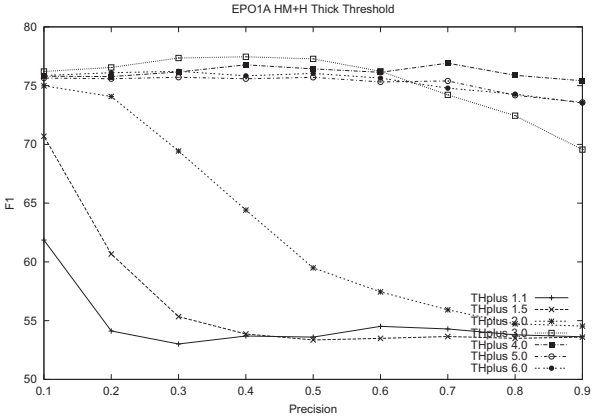




Taking $\beta = 0.7$ gives the best result, improving the Accuracy from .55 at $\beta = 4$ to .715. Notice that the Precision and Recall cross near $\beta = .9$.

- Term selection





The effect of Term Selection is small, the optimum number of terms per class is in a broad band at a high number of terms, much higher than for keywords. It appears that many infrequent terms have to be combined in order to achieve good Precision and Recall.

- Some statistics Training and testing on all documents again, we now obtain:

```
Training phase...
573222 different terms in train set
  137451 global terms in train set
  26546 final terms in train set
Testing phase...
MICRO    16000     96.73%    97.42%    97.07%   Lumped
MACRO    16000     96.74%    97.42%    97.07%   Averaged
```

There are 26000 more terms to begin with, of which 9000 are eliminated as hapaxes. Fewer terms are selected. The final Accuracy on seen documents (97%) is much better, but few HM pairs are included among the terms (see Appendix A.2), confirming our fear that the HM pairs are overwhelmed by their much more frequently occurring heads.

4.2.1 Summary HM+H

Adding the heads to the pure HM pairs greatly improves the Accuracy, provided the parameters are well-chosen. The additional effect of Term Selection is small.

algorithm	method	max F1 value	parameter value
Winnow	raw	.54	$\theta^+ = 1.1, \theta^- = 0.9$
Winnow	tuned	.77	$\theta^+ = 3.0, \theta^- = 0.4$
Winnow	TSel	.79	3000-7000 terms/class
Rocchio	raw	.55	β=4, γ=1
Rocchio	tuned	.715	$\beta = .7$, γ=1
Rocchio	TSel	.72	5000-7000 terms/class

4.3 Phrases plus heads and modifiers

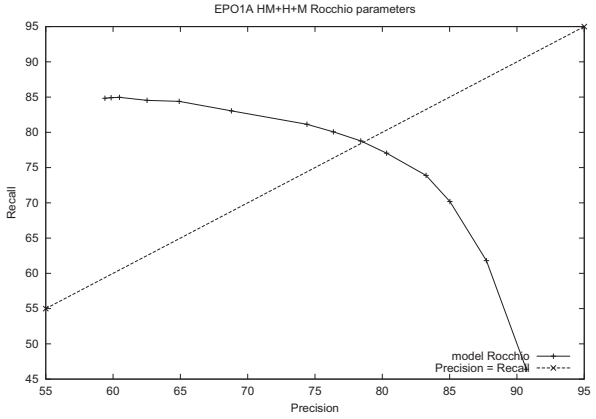
In the next representation, we include besides the pure HM pairs and their heads also their modifiers, which we also expect to be be important keywords.

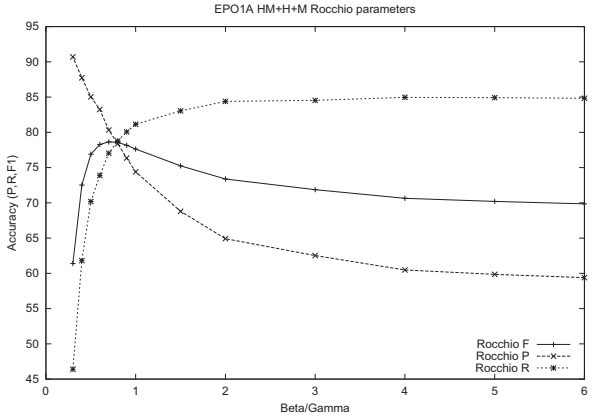
Using this representation, the example tree `[[shoes > walking] > new]` is now unnested to

```
[ shoes ]
[ shoes, walking ]
[ shoes, new ]
[ walking ]
[ new ]
```

We expect this addition to be a mixed blessing: again the number of different terms is increased, and the heads and modifiers are certainly not statistically independent from the HM pair from which they are derived.

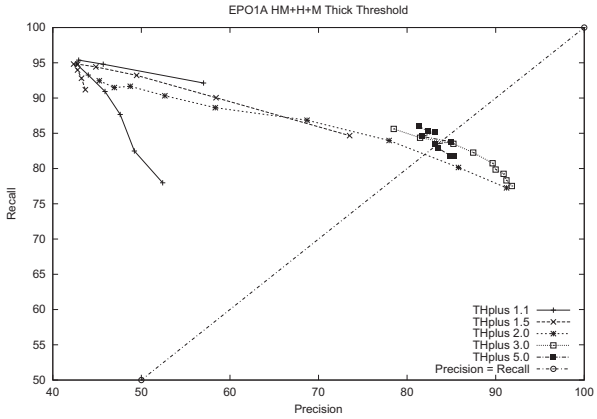
- Rocchio parameters

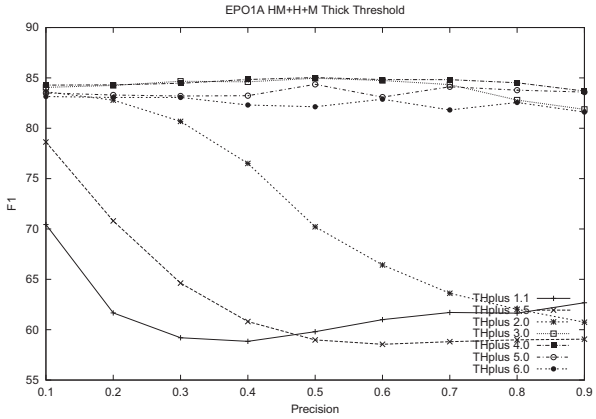




Again, the optimum is at $\beta = 0.7$. The Accuracy is improved from 0.715 to 0.78 by the addition of the modifiers. The trajectory of Precision and Recall is very similar to the previous (and rather similar to that for keywords).

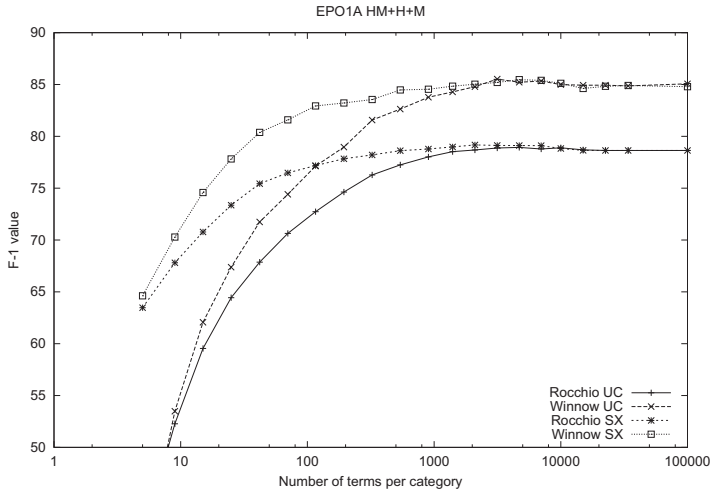
- Winnow parameters





The best threshold choice is 3.5/0.4 or 0.5. The Accuracy reached is .85, very near to that of keywords (.88).

- Term selection



A slight improvement is made by Term Selection, at a high number of terms per class.

- Some statistics

```
  633412 different terms in train set
  162758 global terms in train set
  24650 final terms in train set
Testing phase...
MICRO    16000     97.25%    97.99%    97.62%   Lumped
MACRO    16000     97.26%    97.99%    97.62%   Averaged
```

The modifiers add about 60000 modifiers new terms, of which 25000 are not eliminated as hapaxes. The number of terms after term selection is reduced by about 2000, mostly replacing a number of HM pairs by one word. Indeed, the top 40 of HM+H+M shows very few HM pairs (Appendix A.3).

What we have achieved looks more like a linguistic form of term selection than like the best way to use phrases as terms.

4.3.1 Summary HM+H+M

algorithm	method	max F1 value	parameter value
Winnow	raw	.59	$\theta^+ = 1.1, \theta^- = 0.9$
Winnow	tuned	.85	$\theta^+ = 3.0, \theta^- = 0.5$
Winnow	TSel	.855	3000-7000 terms/class
Rocchio	raw	.715	β=4, γ=1
Rocchio	tuned	.78	$\beta = .7$, γ=1
Rocchio	TSel	.79	2000-7000 terms/class

4.4 Conclusion

The experiments described here did not yield a document representation based on Head/Modifier pairs which gives better classification accuracy than the traditional keywords, but it did give many surprizes. The first surprise is that the optimal setting of the Winnow and Rocchio parameters are so far from the values given in literature. Our main result is that the choice of parameters is crucial. By themselves the optimal parameter settings for Winnow and Rocchio differ little from one representation to another, at least for the EPO1A corpus, but the parameters value quoted in literature are far from optimal.

The Winnow algorithm again clearly outperforms the Rocchio algorithm, although before tuning and Term Selection Rocchio behaves slightly better than Winnow.

The use of an appropriate Term Selection (Simplified ChiSquare rather than Uncertainty) adds some Accuracy (7% in the case of pure HM pairs and 0-2% with keywords added), which shows that Term Selection is important when using HM pairs.

Compared to the use of all keywords as a baseline, the various ways to use phrases instead of keywords give less Accuracy. Even with the best Term Selection, pure HM pairs give the lowest Accuracy. Adding the single heads improves it, and adding the modifiers even more, closely approaching the baseline but still below it (but with this last representation very few HM pairs are actually selected).

This means that at least some of the best classification terms are not heads or modifiers, as found by the syntax analysis. Maybe they are not verbs, nouns or adjectives and therefore do not appear in the HM pairs. Adding phrases to words helps very little, because their Document Frequencies are so low that they get very little weight.

The quality of the linguistic resources is another concern: the limited Precision and Recall of the HM pair extraction (reported to be between 60 and 70%) causes the system to miss about one third of them. But the problem is not just to extract the large possible number of phrases from a text, but to extract the right ones. The useless phrases throw a blanket of noise over the texts, even more so than useless words.

Grudgingly, we must agree with the authorities that we should not replace keywords by phrases but add the phrases to the keywords. However, when we do that, the Accuracy is hardly improved. Simply using all words as terms is much more efficient and still gives a better result. Our most careful Term Selection cannot change this.

4.5 Outlook

In spite of this, we see no reason to give up on the use of phrases. Notice that with the HM pairs representation many more terms are needed for an optimal result than in the case of keywords. Intuitively, there are very many highly precise phrases with a very low Document Frequency. We can try to improve term conflation by stemming and syntactical or semantical normalizations.

But many of the phrases are statistically and linguistically related, or at least not independent, as is the case for two HM pairs with the same head. The logical next step therefore is to perform some form of Term Clustering [Lewis and Croft, 1990] or fuzzy matching [Koster et al., 1999] in order to conflate terms that are not independent, raising the frequency of occurrence of the HM pairs and bringing it near to that of keywords, so that the two representations can be combined successfully.

References

[Arampatzis et al., 2000a] Avi Arampatzis, Jean Beney, C.H.A. Koster, Th.P. van der Weide, KUN on the TREC-9 Filtering Track: Incrementality, Decay, and Threshold Optimization for Adaptive Filtering Systems. The Ninth Text REtrieval Conference (TREC-9), Gaithersburg, Maryland, November 13-16, 2000.

[Caropreso et al, 2000] M.F. Caropreso, S. Matwin and F. Sebastiani (2000), Statistical Phrases in Automated Text Categorization, IEI report IEI-B4-07-2000, Pisa, Italy.
`citeseer.nj.nec.com/caropreso00statistical.html`

[Cohen and Singer, 1999] W.W. Cohen and Y. Singer (1999), Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems 13*, 1, 100-111.

[Dagan et al, 1997] I. Dagan, Y. Karov, D. Roth (1997), Mistake-Driven Learning in Text Categorization. In: *Proceedings of the Second Conference on Empirical Methods in NLP*, pp. 55-63.

[Evans and Lefferts, 1994] D. Evans and R.G. Lefferts (1994), Design and evaluation of the CLARIT-TREC-2 system. Proceedings TREC-2, NIST Special Publication 500-215, pp. 137-150.

[Fagan, 1988] J.L. Fagan (1988), *Experiments in automatic phrase indexing for document retrieval: a comparison of syntactic and non-syntactic methods*, PhD Thesis, Cornell University.

[1] A. Grove, N. Littlestone, and D. Schuurmans (2001), General convergence results for linear discriminant updates. Machine Learning 43(3), pp. 173-210.

[Koster et al., 1999] C.H.A. Koster, C. Derksen, D. van de Ende and J. Potjer, Normalization and matching in the DORO system. Proceedings of IRSG'99, 10pp.

[Koster et al, 2001] C.H.A. Koster, M. Seutter and J. Beney (2001), Classifying Patent Applications with Winnow, Proceedings Benelearn 2001, Antwerpen, 8pp.

[Krier and Zaccà, 2001] M. Krier and F. Zaccà (2001), Automatic Categorisation Applications at the European Patent Office, International CHemical Information Conference, Nimes, October 2001, 10 pp.

[Lewis and Croft, 1990] Term Clustering of Syntactic Phrases (1990), Proceedings SIGIR 90, pp. 385-404.

[Lin, 1995] D. Lin (1995), A dependency-based method for evaluating broad-coverage parsers. *Proceedings IJCAI-95*, pp. 1420-1425.

[Peters and Koster, 2002] C. Peters and C.H.A. Koster (2002), Uncertainty-based Noise Reduction and Term Selection, Proceedings ECIR 2002, Springer LNCS 2291, pp 248-267.

[Rocchio, 1971] J.J. Rocchio (1971), Relevance feedback in Information Retrieval, In: Salton, G. (ed.), *The Smart Retrieval system - experiments in automatic document processing*, Prentice - Hall, Englewood Cliffs, NJ, pp 313-323.

[Ruge, 1992] G. Ruge (1992), Experiments on Linguistically Based Term Associations, Information Processing & management, 28(3), pp. 317-332.

[Strzalkowski, 1992] T. Strzalkowski (1992), TTP: A Fast and Robust Parser for Natural Language, In: Proceedings COLING '92, pp 198-204.

[Strzalkowski, 1999] T. Strzalkowski, editor (1999), *Natural Language Information Retrieval*, Kluwer Academic Publishers, ISBN 0-7923-5685-3.

[Yiming and Pedersen, 1997] Y. Yiming and J.P. Pedersen (1997), A Comparative Study on Feature Selection in Text Categorization. In: ICML 97, pp. 412-420.

A Most important terms

For one of the 16 classes (class04) we show the 40 most positive and negative terms, using the Winnow weight $W_t^+ - W_t^-$ as the measure of importance. The most important terms of a classifier, and in particular the negative terms, can be very surprising index terms when their semantics is considered, because their origin is purely statistical.

A.1 Pure HM pairs

In the left column the positive terms are shown with their Winnow weight, in the right column the negative ones.

```
file: data/class04.cpf
[    insert,   cutting] 0.301065 [        it, connected] -0.018143
[       bar,    boring] 0.100540 [        it,   mounted] -0.014295
[      tool,  drilling] 0.046827 [        it,  attached] -0.010098
[     chuck,       has] 0.033779 [   machine,  grinding] -0.006916
[ spindle,machine tool] 0.033027 [ apparatus,  includes] -0.006624
[    insert, indexable] 0.030669 [        it,   applied] -0.006030
[      jaws,  clamping] 0.029042 [        it,associated] -0.005645
[  teeth, cutting] 0.027096 [engine,internal combustion] -0.005623
[   element,   cutting] 0.026082 [        it,   coupled] -0.005092
[      head,      tool] 0.025630 [        it,  arranged] -0.004032
[    insert,       has] 0.024524 [        it,    placed] -0.003870
[   machine,  drilling] 0.023802 [       end,     lower] -0.003753
[      tool,    boring] 0.022418 [        it,  supplied] -0.003486
[      head,    boring] 0.022194 [      side,       one] -0.003102
[   machine,    boring] 0.022044 [        it,  disposed] -0.003059
[      tool, deburring] 0.019345 [        it,   located] -0.002987
```

```
[      axis,     chuck] 0.018748 [    system,  includes] -0.002915
[     chuck,    having] 0.017917 [        it,   secured] -0.002828
[   holding,      tool] 0.017817 [    device,  includes] -0.002776
[tool holder,   having] 0.017510 [  position,    second] -0.002776
[       end,   cutting] 0.016763 [ connected, pivotally] -0.002686
[       bit,      tool] 0.016444 [    having,   surface] -0.002665
[tool holder,      has] 0.016344 [   surface,     inner] -0.002266
[adjustment,    radial] 0.016045 [    having,      pair] -0.002215
[      jaws,     chuck] 0.015645 [        it,    driven] -0.002055
[       tip,   cutting] 0.015202 [       end,     other] -0.001993
[ operation, machining] 0.014888 [        it,  required] -0.001958
[        it,   drilled] 0.014640 [        it, disclosed] -0.001919
[       jaw,       top] 0.014249 [        it, installed] -0.001849
[cutting tool,  having] 0.014063 [       end,     upper] -0.001811
[     stock,       bar] 0.013563 [        it,     fixed] -0.001783
[   inserts,   cutting] 0.013489 [        it,controlled] -0.001683
[     lathe, automatic] 0.013434 [        it,    opened] -0.001667
[ operation,  drilling] 0.013097 [      side,     other] -0.001665
[  clamping,      tool] 0.013010 [     sides,  opposite] -0.001585
[  movement,    radial] 0.012946 [        it,  selected] -0.001569
[     force,  clamping] 0.012633 [        it,  employed] -0.001556
[  movement,     axial] 0.012323 [    center,      dead] -0.001550
[   surface,      pipe] 0.012291 [      side,        on] -0.001550
[      head,   cutting] 0.011868 [    having,    member] -0.001525
```

The number of 3rd person verb-forms with the filler-subject "it" is remarkable. There are numerous examples where morphological conflation (lemmatization) should be helpful.

A.2 HM pairs + heads

```
file: data/class04.cpf
                  chuck 0.333122               grinding -0.040691
           cutting tool 0.213392                 ground -0.033674
[    insert,   cutting] 0.137481              apparatus -0.025367
            tool holder 0.119683               material -0.023279
                  lathe 0.096758                    one -0.023059
               drilling 0.090993              connected -0.021734
                 collet 0.088814                 method -0.019450
                   jaws 0.084564                    air -0.018548
             chuck body 0.081346                  frame -0.018490
           cutting edge 0.076380                   side -0.015939
                    bit 0.062549                 engine -0.014716
                  shank 0.059412 [        it, connected] -0.013755
                 insert 0.058292              including -0.013123
[       bar,    boring] 0.056996                    has -0.012622
                cutting 0.055640                 system -0.012056
          cutting edges 0.055377             supporting -0.011722
                spindle 0.055203                mounted -0.011490
                   tool 0.054769                machine -0.010531
                  drill 0.045572                   used -0.010500
             drill bit 0.044876                  panel -0.010153
             workpiece 0.042357                   path -0.009524
                   head 0.038875                 signal -0.009124
                  holes 0.037894                control -0.009074
                 holder 0.035553             compressor -0.008530
                    jaw 0.034878              polishing -0.008127
           machine tool 0.033788                   wire -0.008091
[      tool,  drilling] 0.032236                 liquid -0.007921
                 boring 0.031047                 driven -0.007851
                   hole 0.030339                devices -0.007004
                   bore 0.029082             embodiment -0.006455
                mandrel 0.027642                  cover -0.006232
                 cutter 0.027624                   each -0.006221
```

```
                screw 0.026520            containing -0.006133
                chips 0.024529                 layer -0.006106
            machining 0.022874                 cable -0.006044
           engagement 0.022696               surface -0.005992
           toolholder 0.021805               circuit -0.005967
                teeth 0.020370             attached -0.005725
             clamping 0.020105                 base -0.005693
                  bar 0.017285             disposed -0.005545
```

This top 40 includes many composed terms which were recognized as collocations rather than as HM pairs (`cutting edge, machine tool`). The presence of collocations in the lexicon improves the precision of the parser but it also reduces the conflation between terms, acerbating the dimensionality problem. A possible solution is to translate also all collocations in the lexicon into HM form. Another possibility is to trust the parser (rather than the lexicon) to find collocations. Notice also that some conflation could be gained by morphological normalization (`cutting edge(s)`).

A.3 HM pairs + heads + modifiers

```
file: data/class04.cpf
                  chuck 0.553001              grinding -0.112165
                boring 0.390707                having -0.056260
                  tool 0.337589                ground -0.046682
              drilling 0.337181                 frame -0.046236
               cutting 0.288783                 lower -0.034253
                 lathe 0.193259              abrasive -0.033839
            tool holder 0.192296             connected -0.031916
                 collet 0.170691          substantially -0.031097
           cutting tool 0.163741                circuit -0.030698
           cutting edge 0.160208                such as -0.029326
              workpiece 0.116131              polishing -0.028434
                   jaws 0.110567               includes -0.025306
                spindle 0.106145              apparatus -0.024551
[    insert,   cutting] 0.104987                    air -0.024005
             drill bit 0.102570               material -0.023445
              radially 0.101235               attached -0.023374
                insert 0.093745                  motor -0.022296
                cutter 0.092428             supporting -0.021492
          machine tool 0.092343             preferably -0.020910
             machining 0.084084                  while -0.019181
           chuck body 0.083999                 signal -0.018727
                  jaw 0.077567                   used -0.017818
              clamping 0.072417                 engine -0.017674
                  chip 0.070480               flexible -0.017233
                 shank 0.064043                   lens -0.016784
                   bit 0.058559                 manner -0.015212
        cutting edges 0.054391                 system -0.014883
                 drill 0.051551                 liquid -0.014798
              drilled 0.046899                 bottom -0.014781
           toolholder 0.046808                  panel -0.014634
                holes 0.044175              pivotally -0.014195
                 bore 0.044049                plastic -0.014173
                  bar 0.042413           therebetween -0.014116
                axial 0.036344                   type -0.013931
                screw 0.036101              operating -0.013575
           engagement 0.034555                   flow -0.013132
              locking 0.034294                   path -0.013027
            hole saw 0.033908                control -0.011893
              holder 0.030622              compressor -0.011414
              radial 0.029467                   then -0.011304
```

Translation Events in Dutch Cross-Language Information Retrieval

Anne R. Diekema
Center for Natural Language Processing
School of Information Studies, Syracuse University
4-206 Center for Science and Technology
Syracuse, NY 13244, U.S.A
diekemar@syr.edu

Abstract

The paper describes an analysis of the translation events encountered when queries cross the language barrier in cross-language information retrieval. A study of a set of query source and target triples resulted in the creation of a translation taxonomy. The taxonomy was used to code 750 English target queries. The 750 coded queries are currently being used in retrieval experiments to assess the impact of the different translation problems on retrieval performance.

1 Introduction

Cross-Language Information Retrieval (CLIR) is a special type of Information Retrieval. In CLIR, retrieval is not restricted to the query language; rather queries in one language retrieve documents in multiple languages. Cross-language retrieval systems allow users to state their queries in their native language, and retrieve documents in all the languages supported by the system. (Oard and Diekema, 1998).

Because queries and documents in CLIR do not always share the same language, translation is needed before matching can take place. The literature explores four different translation options: translating queries (e.g. Ballesteros and Croft 1997), translating documents (Oard and Hackett 1998), translating both queries and documents (Dumais et al. 1997), and cognate matching, which matches words with close to or identical spelling (Buckley et al. 1998). The general trend however, as is the case in this study, is to use query translation.

Automatic translation of queries or documents requires lexical resources such as machine-readable dictionaries, ontologies, corpora, or machine translation systems. One of the problems facing CLIR is that lexical resources are not widely available, especially for less common languages. Most resources require linguistic processing to create useable translation tools. Even ready-made resources such as machine-readable dictionaries need processing to filter out extraneous information; for example, the definition of the lexical item that is provided in dictionaries for human users tends to confuse computer systems (Hull and Grefenstette 1996). Multilingual ontologies with their rich internal knowledge structures are extremely time-consuming to create and are only used by a limited group of CLIR researchers (Gilarranz et al. 1997). The largest resource for translation data is formed by multilingual corpora, which are used to mine translation data. Previous to the extraction of translation equivalents, alignment (e.g. at the sentence level) of a corpus is required (Dumais et al. 1997; Nie et al. 1999; Sheridan et al. 1998). The most obvious solution to the translation problem is to apply machine translation (Gey et al. 1999; Oard and Hackett 1998). Machine translation performs reasonably well but is limited to only those language pairs that are available. In addition, machine translation is often not sufficient for short query translation because short queries lack context, which is used by machine translation systems to determine the sense of the word.

Translation problems from the translation literature	
lexical ambiguity	words having multiple meanings
lexical mismatches	differing conceptual structures between language communities
lexical holes	unlexicalized concepts across languages
figures of speech	words that should not be taken literally or words that are used to create a certain literary effect
multiword lexemes	idioms, phrasal verbs, and collocations
specialized terminology and proper names	words used by certain discourse communities and names of people, places and organizations often do not appear in dictionaries
false cognates	words that seem to be the same across languages but are not

Table 1: Specific translation problems from the translation literature.

It is often thought that by acquiring the right translation resources the translation problem, and thus CLIR, is solved. Although a relationship between dictionary quality and retrieval performance has been observed (Kraaij 2001, McNamee and Mayfield 2002), natural language is too complex for this to be true. Even in the monolingual case, lexical ambiguity and synonymy cause serious problems. When an additional language is added to the mix, problems increase. In the ideal situation, a word or phrase has only one sense and hence only one translation. However, many words have multiple senses, each of which may have single or multiple translations. Although phrases exhibit less translation ambiguity than single words, non-compositional phrases are problematic since they cannot be reconstructed from translations of the constituent terms. In the worst situation, a word or phrase does not have a foreign language equivalent. The CLIR literature recognizes three main factors causing translation error in dictionary-based translations: lack of translations for technical terms and acronyms; erroneously breaking up non-compositional phrases in translation; and the addition of multiple translation senses of a word to the translation (Hull and Grefenstette 1996).

The translation literature lists similar and additional translation problems: lexical ambiguity, lexical mismatches, lexical holes, figures of speech, multiword lexemes, specialized terminology, and false cognates (see Table 1). According to Gophinathan (1993), problems of meaning result from words having 1) suggestive meaning as well as literal meaning; 2) socio-cultural meaning, such as culturally specific lexical items, idioms, and folk images; and 3) false cognates.

CLIR performance is commonly expressed as a percentage of monolingual effectiveness. Reported values typically range from around 50% for unconstrained dictionary-based query translation, to 98% for more sophisticated techniques. On some test collections CLIR systems can even be more effective than monolingual systems due to careful exploitation of translation resources and combinations of query expansion techniques.

The generally reduced performance of CLIR is caused by translation, which adds additional noise to a query. A large part of CLIR research is therefore devoted to finding reliable methodologies to reduce translation ambiguity. Solutions so far have included using part-of-speech taggers to restrict the translation options (Davis 1996), applying query expansion techniques (McNamee and Mayfield, 2002), using corpora for term translation disambiguation (Ballesteros and Croft, 1998), and weighted Boolean models which tend to have a self-disambiguating quality (Hull 1997; Hiemstra and Kraaij 1999).

A careful examination of the nature of translation problems, and the relation of these problems to cross-language retrieval performance, has received little attention in the literature. This ongoing research outlines the translation problems facing the CLIR query translation process, and is in the process of analyzing how these translation problems impact retrieval

English source query
Ireland, peace talks
Dutch source query
Ierland, vredesbesprekingen
English target query
Ireland, island of Ireland, Erin, The Emerald Isle, Hibernia, The Republic of Ireland, Eire, peace talks, peace negotiations

Table 2: Query triple for title query 404.

performance to provide a better understanding of the theoretical and practical implications of translation in CLIR.

2 Methodology

The study followed a two-phase multi-method approach. In phase one, a taxonomy of translation events was created through content analysis of queries and their translations, in combination with an examination of the literature. In the second and final phase, the test queries were coded using the taxonomy resulting from phase one. These queries were then used in information retrieval experimentation to assess the impact of translation problems on retrieval performance. The analysis of the retrieval results using multiple regression analysis has not been completed at this time.

2.1 Queries

The majority of TREC topics consist of 3 parts: a title, description, and a narrative. The title is the shortest version of a topic and describes it in a few key terms. The description is a longer version of a topic and describes the topic in a full sentence. The narrative lists explicitly what constitutes a relevant document and a non-relevant document. The narrative is intended to aid TREC judges in their relevance assessments, but is often used as part of the topic for document retrieval. This study only used the title and description since these better resemble a user query than the relatively long narrative (Cutting et all., 1997). Using both the title and the description provided two versions of the same query, varying in query length. Partitioning the 400 TREC topics (051-450) into title and description resulted in 350 title queries, and 400 description queries.

The English source queries were manually translated into Dutch by an independent translation bureau to create a high quality equivalent Dutch query set. These manual Dutch translations can be considered Dutch equivalents of the English originals, and both sets are viewed as source queries as is common in CLIR evaluations (CLEF). The Dutch source queries were automatically translated back into English using a machine-readable dictionary (Van Dale groot woordenboek), creating the English target queries. Thus, data collection resulted in 750 source and target language query triples, where each query has an English source query, a Dutch source query, and an English target query (see Table 2).

2.2 Query translation process

The electronic version of the Van Dale Groot Woordenboek Nederlands-Engels (1997) was used to translate the Dutch source queries back into English. To accomplish this automatic translation, a term list first needed to be extracted from the dictionary itself. The set of unique TREC query terms was used for dictionary lookup. All dictionary entries for each term were collected in a single file, which was processed to create a straight term translation list. This term list was used for the automatic translation.

All Dutch source queries were tokenized (based on white space) and each content term was looked up in the term list described above. If a term was on the list, it was replaced with all the translations listed in the entry. If the term was not on the list, it was kept untranslated. The original terms are kept in this case because these terms are often proper names that transcend the language barrier.

A randomly chosen subset of the 750 query triples was used to establish which translation problems were encountered when crossing the language barrier. Content analysis in combination with a literature review, resulted in a translation problem taxonomy. This taxonomy was then used to code all 750 queries. In addition, the English source queries and the (coded) target queries were used in the CLIR retrieval experiment.

3. Results

3.1 Taxonomy

An analysis of a subset of the query triples was carried out to study the effect of translation on the queries. Content analysis showed that there are three kinds of translation problems: 1) problems with the lexical resource, 2) problems with the source word, and 3) problems with terms that remain untranslated. Problems concerning the lexical resource are discovered at the initial term lookup. The term might simply not be listed, or have errors in its lexical entry as a result of the automatic conversion from machine readable dictionary to translation term list. Problems with the source word capture possible linguistic issues that might hinder a translation. For example, the meaning of a non-compositional noun phrase might get lost in a word-by-word translation. Problems with untranslated words concern terms that did not have an entry in the translation resource. Problems occur when an untranslated word exists in the target language but has a different meaning entirely.

It is the result of a particular translation problem that impacts information retrieval, not the actual problem itself. It is important therefore that in categorizing the query using the translation taxonomy, we also get an indication of what happened in the translation building stage. For example, lexical ambiguity sometimes results in an erroneous translation. Other times the ambiguity results in both an erroneous translation as well as a correct one. Examination of the word-by-word translation building process revealed four basic events in a translation: 1) the term is translated correctly and is identical to the term used in the English source query, 2) the term is translated correctly but is not identical to the term used in the English source query translation, 3) the term is translated incorrectly, 4) the term could not be translated. In short, either the translation was correct, incorrect, or had no translation at all. This is of course a simplification of an actual translation where one will find several possible combinations of these four events (i.e. both correct and incorrect translations for a single entry), and variations in the magnitude of these events (i.e. depending on the number of translations found in a dictionary entry).

When we combine the three translation classifiers with the translation builder we get a query categorization schema (see Figure 1) that combines all aspects of the translation: the translation problems, the result of the problems, and the magnitude of the problems.

The taxonomy was tested for reliability in a test where two judges coded 16 queries. The test resulted in a Kappa of 0.87. A reliability coefficient of this magnitude is considered to be excellent. The Kappa coefficient measures the proportion of inter-rater agreement after chance agreement has been removed. To calculate this coefficient you measure the observed proportion of agreement between coders, and the proportion of agreement expected by chance. The more the observed proportion of agreement exceeds that of agreement expected by chance, the higher the coefficient.

3.2 Query codes

All 750 queries were coded using the taxonomy developed (see section 4.1). Each Dutch source query term – English target translation term pair was given an individual code vector. The vectors represent the query categorization schema. The 31,024 vectors traveled through the schema in 17 different ways (see Table 3). Out of the 17 possible vectors, a number vectors appear less than 10 times, and others well over a thousand. The vector distribution is thus heavily skewed. Even though the translation problem taxonomy is very expressive, not many different problems seem to occur in this sizeable query set.

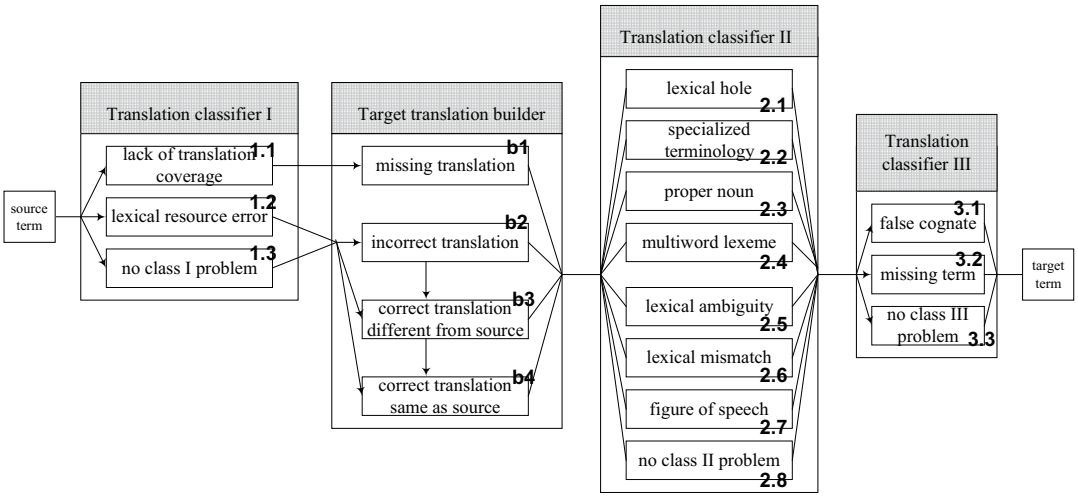


Figure 1: Translation taxonomy.

The four most common code vectors were: ambiguity, correct but different, wrong translation due to multi word expression, and correct. More than half the terms were translated incorrectly due to ambiguity (52%). In this case the Dutch source terms had multiple senses and all their translations were added to the query, some of them incorrect. Just under a quarter of all terms (23%) was translated correctly but with a synonym of the English source term. The distinction between "correct" and "correct but different" was made because this could potentially cause a retrieval performance difference between source and target queries. About 12% of the terms were parts of phrases that could not be translated word-by-word but had been. Just over 9% of the terms were translated correctly with the same terms used in the English source query. The remaining 4% of translation events is spread out over 13 different code vectors.

3.3 Information retrieval experiment

The English source and target queries were used in an information retrieval experiment. The results are currently being analyzed. To address the problem of queries possibly having multiple translation problems, multiple regression will be used to assess the impact of translation problems on retrieval performance. Multiple regression allows two or more independent variables to predict scores on a dependent variable. Instead of using one independent variable (one of the many possible translation problems) at a time, multiple regression allows the combination of numerous translation problems to influence the dependent variable (retrieval performance).

3.4 Limitations

The study findings are dependent on the following: languages under study, lexical translation resource, and the test queries. There are several thousand languages in the world and this study only covered two of them. The two languages, English and Dutch, not only both belong to the world's largest language group of Indo-European languages, but also share the same language sub-group of Germanic languages (Katzner, 1995). It could be argued that such a narrow selection limits the findings of this study. Naturally, there are likely to be translation problems specific to the unidirectional Dutch-English language pair. Although using the translation literature and CLIR literature enabled the researcher to find additional translation problems that transcend this single language pair, generalization to other language pairs might still be problematic. The translation resource used is a relatively standard one for Dutch. However, only replication of this study with other translation resources can show whether the results will be the same. Although a large number of queries were used in this study the number of translation events found was relatively small. It would be interesting to see whether this is true for other collections of queries.

code vector name	Vector	Frequency
reverse lexical hole	1.1|b1|2.1|3.3	4
The Dutch language has no term for this phenomenon so it is forced to use the English terminology. (196ETT voucher in school vouchers)		
missing specialized terminology	1.1|b1|2.2|3.2	122
The source term is very specific and does not occur in a regular dictionary. The term is lost in the translation. (170ETD borstimplantaten siliconengel – silicone gell breast implants)		
specialized terminology	1.1|b1|2.2|3.3	36
The source term is very specific but Dutch uses the English term. (066ETT natural language processing)		
missing Proper Name	1.1|b1|2.3|3.2	3
The Proper Name in the source language does not appear in the dictionary. The term is lost in the translation. (163D Zuid-Vietnam – South Vietnam)		
Proper Name	1.1|b1|2.3|3.3	171
Again, the Proper Name in the source language does not appear in the dictionary but is a cognate. (109D Minnesota Mining and Manufacturing)		
missing MWL	1.1|b1|2.4|3.2	7
The source term is part of a multi-word lexeme and is lost in the word-by-word translation. (091D met name genoemde ("name" not in dictionary)		
missing general term	1.1|b1|2.8|3.2	98
Term not in dictionary (no obvious reason). The term is lost in the translation. (258D bevoegdheid)		
general term or number	1.1|b1|2.8|3.3	12
Term (number) not in the dictionary but is a cognate. (053D "200", or 167D sex)		
lexical creation error	1.2|b2|2.8|3.3	133
Error in creation of the lexicon, term translated incorrectly. (449D verklaren translated as "zich verklaren")		
wrong translation due to lexical hole	1.3|b2|2.1|3.3	8
Term cannot be translated back to the original English source term since the Dutch language does not have an equivalent term. Term translated incorrectly. (169D local, state, federal – Dutch source has regional instead of state).		
wrong translation of Proper Name	1.3|b2|2.3|3.3	393
The English Proper Name in the Dutch source query is translated (false cognate). Erroneous terms added to translation and possibly (parts of) the Proper Name get(s) lost in translation. (320 Fiber Optic Link around the Globe (FLAG) - link is translated as sly.)		
wrong translation due to MWL	1.3|b2|2.4|3.3	3761
Dutch source query has a MWL that needs to be translated as a whole but gets lost in the word-by-word translation. Erroneous terms are added to the translation and the MWL might be lost in the translation as well. (416D stand van zaken – should be translated simply as status)		
wrong translation due to ambiguity	1.3|b2|2.5|3.3	16128
Dutch source term has multiple meanings. Erroneous terms are added to the translation. (130T betrekkingen – should be translated as relations but also gets job, and position.)		
wrong translation due to lexical mismatch	1.3|b2|2.6|3.3	4
The way of thinking between the two languages differs, term cannot be translated back to the English source term. (063 vertaalprogramma – translation programs but should be machine translation.)		
wrong translation due to figure of speech	1.3|b2|2.7|3.3	166
Figure of speech gets translated literally. Erroneous terms added to the translation and the meaning is lost. (069 nieuw leven inblazen - literally: blow new life into, but should be translated as revive)		
correct translation but different	1.3|b3|2.8|3.3	7094
Correct translation but translation is a synonym of the term used in the English source query. (195 fluctuaties – in English source as "shifts", translated as : fluctuation, drift, change, instability, swing.)		
correct identical translation	1.3|b4|2.8|3.3	2884
Correct translation and identical term as the term used in the English source query. (405D heelal – translated as cosmos)		
		Total number of code vectors: 31,024

Table 3: The seventeen translation code vectors.

4. Conclusions

When examining the translation events in cross-language query translation in more detail, it appears that more than half of the erroneous term translations in a large query set are caused by ambiguity. The other main cause of translation error is the word-by-word translation of terms that are part of multiple-word expressions. About one third of the terms are translated correctly, some with terms identical to the English source queries, and others with synonyms. To study the impact of these and other translation events on query performance a multiple regression analysis of retrieval results using these queries is underway.

References

Ballesteros, L. and Croft, B. (1996). *Dictionary Methods for Cross-Lingual Information Retrieval*. In: "Proceedings of the 7th International DEXA Conference on Database and Expert Systems", 1996 September 9-13; Zürich, Switzerland. New York, NY: Springer, 1996. 791-801

Ballesteros, L. and Croft, B. (1997). *Phrasal Translation and Query Expansion Techniques for Cross-Language Information Retrieval*. In: "Proceedings of the Association for Computing Machinery Special Interest Group on Information Retrieval (ACM/SIGIR) 20th International Conference on Research and Development in Information Retrieval"; 1997 July 25-31; Philadelphia, PA. New York, NY: ACM, 1997. 84-91.

Ballesteros, L. and Croft, B. (1998). *Resolving Ambiguity for Cross-language Retrieval*. In: "Proceedings of the Association for Computing Machinery Special Interest Group on Information Retrieval (ACM/SIGIR) 21st International Conference on Research and Development in Information Retrieval"; 1998 August 24-28; Melbourne, Australia. New York, NY: ACM, 1998. 64-71.

Buckley, C.; Mitra, M.; Walz, J.; and Cardie, C. (1998). *Using Clustering and Super Concepts within SMART: TREC 6*. In: "Proceedings of the 6th Text REtrieval Conference (TREC-6)"; 1997 November 19-21; National Institute of Standards and Technology (NIST), Gaithersburg, MD. 107-124.

CLEF – Cross-Language Evaluation Forum: http://www.clef-campagin.org.

Davis, M. (1997). "*New Experiments in Cross-Language Text Retrieval at NMSU's Computing Research Lab.*" In: "The Fifth Text Retrieval Conference (TREC-5)". 1996, November. National Institute of Standards and Technology (NIST), Gaithersburg, MD.

Dumais, S. T.; Letsche, T. A.; Littman, M. L.; and Landauer, T. K. (1997). *Automatic Cross-Language Retrieval Using Latent Semantic Indexing*. In: American Association for Artificial Intelligence (AAAI) Symposium on Cross-Language Text and Speech Retrieval; 1997 March 24-26; Palo Alto, CA 1997. 15

Gey, F., Jiang, H., and Chen, A. (1999). *Manual Queries and Machine Translation in Cross-Language Retrieval and Interactive Retrieval with Cheshire II at TREC-7*. In: "The Seventh Text REtrieval Conference (TREC-7)". 1998, November 9-11; National Institute of Standards and Technology (NIST), Gaithersburg, MD. 527-540.

Gilarranz, J.; Gonzalo, J. and Verdejo, F. (1997). *An approach to Conceptual Text Retrieval Using the EuroWordNet Multilingual Semantic Database*. In: "American Association for Artificial Intelligence (AAAI) Symposium on Cross-Language Text and Speech Retrieval"; 1997 March 24-26; Palo Alto, CA 1997. 49-55.

Gopinathan, G. (1993). *The Nature and Problems of Translation*. In: "Problems of Translation". G. Gopinathan and Kandaswamy, S. (Eds.). Allahabad, India: Lokbharati Prakashan. 37-50.

Hiemstra, D. and Kraaij, W. (1999). *Twenty-One at TREC-7: Ad-hoc and Cross-language Track*. In: E.M. Voorhees and D.K. Harman (Eds.) "The Seventh Text REtrieval Conference (TREC-7)". 1998, November 9-11; National Institute of Standards and Technology (NIST), Gaithersburg, MD. 227-238.

Hull, D. A. (1997). *Using Structured Queries for Disambiguation in Cross-Language Information Retrieval*. In: "American Association for Artificial Intelligence (AAAI) Symposium on Cross-Language Text and Speech Retrieval"; 1997 March 24-26; Palo Alto, CA 1997. 84-98.

Hull, D. A. and Grefenstette, G. (1996). *Querying across Languages: A Dictionary-Based Approach to Multilingual Information Retrieval*. In: "Proceedings of the Association for Computing Machinery Special Interest Group on Information Retrieval (ACM/SIGIR) 19th International Conference on Research and Development in Information Retrieval"; 1996 August 18-22; Zürich, Switzerland. New York, NY: ACM, 1997. 49-57.

Katzner, K. (1995). *The Languages of the World*. New Edition. New York, NY: Routledge. 378p

Kraaij, Wessel (2001) *TNO at CLEF-2001: Comparing Translation Resources*. In: "CLEF-2001 Working Notes". (http://www.clef-campaign.org).

McNamee, Paul and Mayfield, James. (2002) *Comparing Cross-Language Query Expansion Techniques by Degrading Translation Resources*. In: "Proceedings of the Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval"; 2002 August 11-15. Tampere, Finland. 159-166.

Nie, J.; Simard, M.; Isabelle, P.; and Durand, R. (1999). *Cross-Language Information Retrieval based on Parallel Texts and Automatic Mining of Parallel Texts from the Web*. In: "Proceedings of the 22nd International Conference on Research and Development in Information Retrieval". Berkeley, California: ACM/SIGIR. 74-81.

Oard, D. and Diekema, A. (1998). *Cross-Language Information Retrieval*. In "Annual Review of Information Science (ARIST)", Vol. 33, Martha Williams (Ed.), Information Today Inc., Medford, NJ, 1998.

Oard, D. and Hackett, P. (1998). *Document Translation for Cross-Language Text Retrieval at the University of Maryland*. In: "Proceedings of the 6th Text REtrieval Conference (TREC-6)"; 1997 November 19-21; National Institute of Standards and Technology (NIST), Gaithersburg, MD. 687-696.

Sheridan, P., Ballerini, J.P. and Schäuble, P. (1998) *Building a Large Multilingual Test Collection from Comparable News Documents*. In: "Cross Language Information Retrieval". G. Grefenstette, (Ed.): Boston, MA: Kluwer Academic.

Van Dale Groot Woordenboek Nederlands-Engels (1997). Van Dale Lexicografie. Utrecht, The Netherlands. (based on the 2nd edition paper version published in 1991).

URUK, a platform for causal text retrieval

Dirk VERVENNE
Hogeschool Gent,
Cel voor Onderzoek en Dienstverlening
Voskenslaan 270 B-9000 Gent, CTL-building
Gent, Belgium, 9000
Dirk.Vervenne@hogent.be

R. Godijns (**), S. Vandepitte (**),
H. Verplaetse (**), C. De Groote (**),
K. Denturck (**) P. Kaczmarski (***),
S. Gierts (***), F.Vandamme (***)
(**) Hogeschool Gent, Departement Vertaalkunde
(***) Labo Toegep. Epistemology, University Gent

Abstract

In this paper we present the Uruk-platform: a system that indexes multilingual documents on the basis of causal patterns. The basic idea is twofold: (1) to develop a search engine that is specialised in causal retrieval of documents and (2) to connect documents automatically when their causal indexes have common causes or results, taking into account the 3 standard relation types of a thesaurus. The paper presents the global architecture, identifying the basic components. The multilingual approach is explained and we end the paper by describing the test environment. The project is currently funded by IWT through the HOBU-funding programme under contract 10136.

Introduction

Text indexing platforms that incorporate the semantic richness of a thesaurus are usually imbedding the latter as a manual support during the indexing and/or the search process (Callan, 1993; Church, 1988; Croft, 1991; Crouch, 1990). The automatic thesaurus-based text indexing have been introduced in, e.g., Khan et al, 1997, Ginsberg A. 1993, Gao et al, 1995 and Vervenne, 1999. The advantage is this approach is the automatic extension of the index with (1) broader terms through the hierarchic BT/NT-thesaurus relations, with (2) equivalent terms based on the USE/USE FOR-thesaurus relation and (3) related terms through the RT-thesaurus-associations.

However, all these approaches are limited to the true nature of a thesaurus, being the exclusive focus on nouns and their semantic interrelations. Since the standardisation rules that are contained in ISO-5964, 1985 and ISO-2788, 1986, both express the requirement not to use verbs while constructing and maintaining thesauri, it means that action-related dependencies in the indexed documents, such as causal relations, are not captured during the thesaurus-based indexing process.

Since "*causality is one of the most intriguing concepts which human being acquires in the earliest stages of life*" (Vandepitte, 2000), it is astonishing that little research effort has been established in the area of automatic causal text indexing. Furthermore, scientific research in nearly every domain is becoming more and more highly specialised; this evolution therefore requires tools which link automatically the small pieces of causal reasoning chains that might be expressed in scientific texts. We therefore decided to combine the strength of our thesaurus-based indexing algorithms (IKEM^{pat}) with the complexity of the English and French causal text patterns: this combination provides the cornerstone of the URUK-architecture.

1 The URUK-architecture

The Uruk-architecture, illustrated in figure 1, is basically developed on top of a set of a pattern-matching algorithms. The work by, e.g., Hearst (1998) indicated that this approach is a powerful paradigm for knowledge-based document indexing.

The Uruk-platform is developed as a compatible module to the existing IKEM^{pat}-system, which uses a thesaurus to generate automatically keywords and concepts as meta-information (Vervenne, 1999). The latter is then used in the Uruk-platform as values for variables that are

expressed in the causal patterns. These patterns are stored in the Uruk-pattern database, for which a dedicated editor is provided to support the maintenance of the language-sensitive causal expressions.

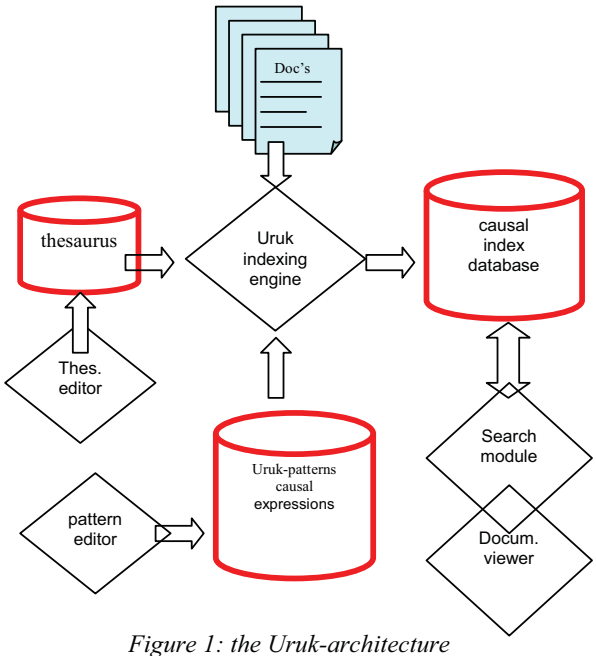


Figure 1: the Uruk-architecture

The Uruk-indexing engine includes a fuzzy algorithm for scanning all sentences in order to map the causal patterns. When a causal expression is found, the thesaurus is scanned for relevant descriptors. Such descriptors are then recognized as 'causes' and 'results', which are stored in the Uruk-index.

2. Causal document linking

The causal index of our Uruk-platform can be queried through the dedicated search module which both provides the causal query-facility and following advanced search options that are currently developed:

(1) the 'multi-doclink' option gives the user the possibility to search for linked documents on the basis of common causal indexes. If, e.g.. a document D1 contains the causal link

'A → B'

and an other document D2 is indexed by

'B → C',

then the query

'*Is C influenced by A ?*'

will automatically yield the link between the documents D1 and D2, through the 'multi-doclink' search option. This multi-doclink option could function also in case

'Bbis → C'

is given, where Bbis is, e.g., a synonym for B.

(2) the 'concept-link' option gives the user the facility to search causal links on the basis of the broader-term descriptors of the indexed causes and results stored in the index-records. If, e.g.. a document D1 contains the causal link

'AA → BB'

and if A and B are thesaurus-descriptors that are not present in D1 and if AA is a narrow term of A and BB is a narrow term of B in the thesaurus, then the query

'*Is B influenced by A ?*'

will deliver document D1; we call this the 'concept-link' option since the IKEM^{pat}-platform considers A and B as concepts for document D1.

(3) The combination of the two mentioned query-options leads to the facility of identifying 'knowledge gaps' within a collection of documents that have been indexed with the Uruk-platform. . If, e.g.. a document D1 contains the causal link

'AA → BB'

and an other document D2 is indexed by

'B → DD',

and if term BB is narrow term of B then the Uruk-platform will propose the hypothesis that a causal relation might exist between AA and DD.

3. Multilingual causal patterns

Causal patterns have been edited manually by the research members of the project: as experts in English and French language, they developed a classification of causal expressions and stored

the results in a formalised way within a pattern lexicon (Vandepitte, S. 1993 and Godijns, et al., 2002).

Causes and results in such patterns are represented as variables that can be searched in the respective domain thesaurus. Two example-patterns are included in figure 2 (for English) and figure 3 (for French).

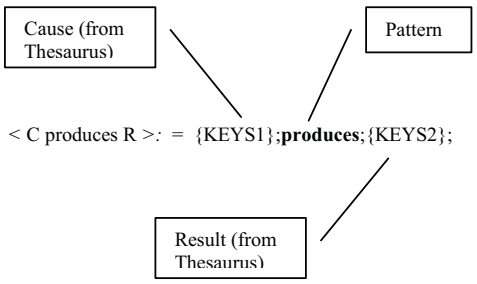


Figure 2 : En-pattern exemple

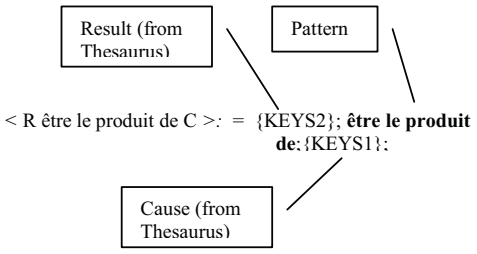


Figure 3 :Fr- pattern exemple

The variables KEYS1 and KEYS2 represent cause en result-terms which are captured from the pre-installed thesaurus. In Kaczmarski P., Gierts S., 2002, a prototype called PatViews, is presented. A screendump of the PatViews-viewer is given in figure 4.

As a side effect of the project, causal pattern structures are compared between the English and the French languages. Also, the developed thesauri in the field of the stock exchange will be studied: e.g., the way the hierarchic broader/narrow relations between terms are expressed in both languages and how a merging process could be supported by the ThesEdit-software.

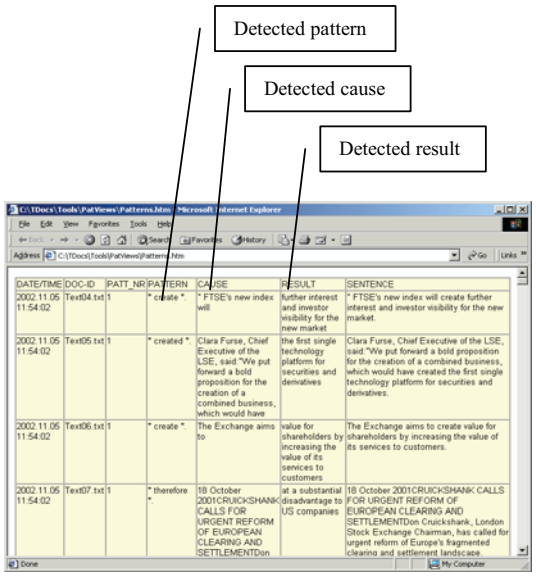


Figure 4 : screendump of PatViews which presents the detected causal patterns for a given set of documents

4. Evaluation tests

The Uruk platform is currently tested within the domain of stock exchanges: the hypothesis is that in this field, many causal relations within socio-economic processes can be detected (Hoover, K. D. 2001). A collection of documents in that domain, both in French and in English were selected. These documents have been used to construct the two domain thesauri.

The tests for the automatic recognition of the causal patterns, are planned in winter 2002. A final prototype will be demonstrated mid 2003. Current tests are limited to 'unambiguous causal patterns' that operate within one single sentence. Since many patterns operate on a paragraph-level, we plan to tackle unambiguous anaphoric expressions in the near future.

Tests have already revealed that some causal relations only hold under certain conditions. The latter should be mentioned when query results are presented to the enduser.

Conclusion

The Uruk-platform combines automatic thesaurus-based indexing algorithms with causal pattern analyses. This platform aims at indexing automatically multilingual texts from a causal point-of-view. The search facilities combine thesaurus-based concept-search as well as document-linking features.

Current tests are running within the domain of stock exchanges for unambiguous patterns that operate on a single sentence level. Since our Uruk-project is half way, more complex patterns will be treated in the next phase of our project.

Acknowledgements

Our thanks go to IWT for providing financial support to the project HOBU-10136.

References

Callan, J.P. and Croft, W.B., 1993, *An Evaluation of Query Processing Strategies Using the TIPSTER Collection*, SIGIR'93 347-355, 1993.

Church, K., 1988; *A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text,* In Proc. of the 2nd Conf. on Applied Natural Language Processing, 136-143,. 1988.

Croft, W.B., Turtle H.R., Lewis D.D., 1991, *The Use of Phrases and Structured Queries in Information Retrieval*. SIGIR'91, 1991.

Crouch, C.J., 1990, *An approach to the automatic construction of global thesauri,* Information Processing & Management, 26(5): 629-40, 1990.

Gao et al, 1995, *Fuzzy multi-linkage thesaurus builder in multimedia systems*, Document Analysis and recognition, 1995, pages 142-145.

Ginsberg A.,1993, *A Unified Approach to Automatic Indexing and Information Retrieval,* IEEE Expert, Oct, vol 8, issue 5, pp 46 – 56.

Godijns R., Vandepitte S., Verplaetse H., Vervenne D., De Groote C., Denturck K., 2002 *URUK, een methode voor geïntegreerd kennisbeheer gebaseerd op een meertalige kennispatroon-editor voor automatische documentontsluiting;* Techn. Rapport: Causale relaties: gebruikers-perspectief, internal report R1.

Hearst M., 1998 "Automated discovery of wordnet relations." In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Databas*e, pages 131–151. MIT Press, Cambridge, MA, 1998.

Hoover, K. D. 2001. *Causality in macroeconomics.*

ISO-2788, 1986,: International Organisation for Standardization. *Guidelines for the Establishment and Development of Monolingual Thesauri.* 2nd ed. Geneva: ISO, 1986.

ISO-5964, 1985, International Organisation for Standardization. *Guidelines for the Establishment and Development of Multilingual Thesauri*. 2nd ed. - Geneva: ISO, 1985.

Kaczmarski P., Gierts S., 2002 *PatViews : Causal pattern matching within the Uruk-framework,* Internal Uruk-Report

Khan et al, 1997, *Personal Adaptive Web Agent: a Tool For Information Filtering*, Electrical and Computer Engineering, , IEEE 1997, Canadian Conference, pages 305-308, vol 1

Vandepitte, S. 1993, *A pragmatic study of the expression and the interpretation of causality Conjuncts and conjunctions in modern spoken British English*. Verhandelingen van de Koninklijke Academie voor Wetenschappen, Letteren en Schone Kunsten van Belgie. Klasse de Letteren 55: 146, Brussel.

Vandepitte, S. 2000, *Causality,* in "Handbook of Pragmatics", J. Verscheuren, J. Ostman, J. Blommaert, C. Bulcaen, eds, John Benjamins Pub Co, Amsterdan/Philadelphia, 2000

Vervenne D., 1999, *Advanced Document Management through Thesaurus-based Indexing: the IKEM^{pp} Platform,* CWI Quarterly, Volume 12, N°2, June 1999, special issue on Digital Libraries

.

A Systematic Evaluation of Concept-based Cross-Lingual Information Retrieval in the Medical Domain

Martin Volk
Eurospider Information Technology AG
Schaffhauserstrasse 18
CH-8006 Zürich, Switzerland
volk@eurospider.com

Paul Buitelaar
DFKI GmbH
Stuhlsatzenhausweg 3
D-66123 Saarbrücken, Germany
paulb@dfki.de

1 Introduction

The paper describes experiments and results of the MuchMore project[1], which is concerned with a systematic comparison of concept-based and corpus-based methods in cross-language information retrieval (CLIR) in the medical domain. Primary goals of the project are to develop and evaluate methods for the effective use of multilingual thesauri in the semantic annotation of English and German medical texts and subsequently to evaluate and compare the impact of such semantic information for the purpose of CLIR. In particular we describe work on semantic annotation with both domain-specific (UMLS, the Unified Medical Language system[2]) and general language semantic resources (EuroWordNet (Vossen, 1997)). Central to the approach is the use of linguistic processing (part-of-speech tagging, morphological analysis, phrase recognition and grammatical function analysis) for an accurate semantic annotation of relevant terms and relations in both the queries and the documents (Vintar et al., 2002). Especially for morphologically rich languages such as German it is important to extend linguistic processing beyond primitive stemming. All linguistic information was added to the documents into an XML format. The document collection used in the project is a parallel English-German corpus of approximately 9000 scientific medical abstracts with a total of one million tokens for each language.

2 Related Work

Many authors have experimented with machine translation or dictionary look-up for CLIR (see e.g. (Kraaij and Hiemstra, 1998)). In a comparison of such methods in both query and document translation, (Oard, 1998) found that dictionary-based query translation seems to work best for short queries while for long queries machine translation of the queries performs better than dictionary look-up. However, machine translation of the documents outperforms all other methods with long queries. An important problem in the translation of short queries is the lack of context for disambiguation of words that have more than one meaning and therefore may correspond to more than one translation (Sanderson, 1994). Therefore, in the case of short queries all translations are considered instead of trying to disambiguate between them.

Ambiguity is also important for interlingua approaches to CLIR that use multilingual thesauri as resources for a language-independent (semantic) representation of both queries and documents. Domain-specific multilingual thesauri have been used for English-German CLIR within social science (Gey and Jiang, 1999), while (Eichmann et al., 1998) describe the use of the UMLS MetaThesaurus for French and Spanish queries on the OHSUMED text collection, a subset of MEDLINE. Both of these approaches use the thesaurus as a source for compiling a bilingual lexicon, which is then used for query translation. A different use of multilingual thesauri is in combination with document classification techniques, such as Latent Semantic Indexing and the Generalized Vector Space Model (Carbonell et al., 1997), both of which depend on parallel corpora. Finally, next to domain-specific thesauri also more general semantic resources such as EuroWordNet have been used in both monolingual and cross-language information retrieval.

In the MuchMore project we assign seman-

[1] http://muchmore.dfki.de
[2] http://umls.nlm.nih.gov

tic codes (MeSH, UMLS and EuroWordNet) to terms on the basis of a linguistic analysis. MeSH codes are assigned to terms in documents as well as in the queries. Annotation with UMLS codes is used for recognition and annotation of semantic relations. Finally, EuroWordNet senses are assigned to all (simple or complex) terms that are represented in this resource.

3 Annotation

The essential part of any concept-based CLIR system is the identification of terms and their mapping to a language-independent conceptual level. Our basic resource for semantic annotation is UMLS, which is organized in three parts.

The **Specialist Lexicon** provides lexical information for medical terms: a listing of word forms and their lemmas, part-of-speech and morphological information.

Second, the **Metathesaurus** is the core vocabulary component, which unites several medical thesauri and classifications into a complex database of concepts covering terms from 9 languages. Each term is assigned a unique string identifier, which is then mapped to a unique concept identifier (CUI). For example, the entry for *HIV pneumonia* in the Metathesaurus main termbank (MRCON) contains (among others) the concept identifier, the language of the term and the string:

```
C0744975 | ENG | HIV pneumonia
```

In addition to the mapping of terms to concepts, the Metathesaurus organizes concepts into a hierarchy by specifying relations between concepts. These are generic relations like *broader_than, narrower_than, parent, sibling* etc. Another component of the Metathesaurus provides information about the sources and contexts of the concepts. The UMLS 2001 version includes 1.7 million terms mapped to 797,359 concepts, of which 1.4 million entries are English and only 66,381 German. Only the MeSH (Medical Subject Heading) part of the Metathesaurus covers both German and English, therefore we only use MeSH terms for corpus annotation.

The third part is the **Semantic Network**, which provides a grouping of concepts according to their meaning into 134 semantic types. The concept above would be assigned to the class *T047, Disease or Syndrome*. The Semantic Network then specifies potential relations between those semantic types. There are 54 hierarchically organized domain-specific relations, such as *affects, causes, location_of* etc.

In the MuchMore project we assigned semantic codes to each sentence based on the linguistic information. MeSH codes were assigned to documents and to queries. UMLS concept identifiers were used as the basis for finding semantic relations. Appropriate EuroWordNet synset codes were assigned if a word or an expression belonged to a EuroWordNet synset (Buitelaar and Sacaleanu, 2001).

4 Evaluation

In order to evaluate whether semantic annotation results in a performance gain in information retrieval, several experiments were carried out. We used our corpus as document collection (the set of medical abstracts described above) in combination with a set of 25 queries and relevance assessments defined by medical experts that are partners in the project. MuchMore aims primarily at cross-language retrieval, but in order to assess CLIR performance, monolingual experiments in German and English were conducted as baselines for the cross-language experiments.

Below we present retrieval results in four columns. The first column contains the overall performance, measured as mean average precision (mAvP) as has become customary in the Text Retrieval Conference (TREC) experiments[3]. This figure is computed as the mean of the precision scores after each relevant document retrieved. The value for the complete evaluation run (i.e. the set of all queries) is the mean over all the individual mean average precision scores. This value integrates both precision and recall and is the most commonly used summary measure. In the second column we present the absolute number of relevant documents retrieved, a pure recall measure. Third, we present the average precision at 0.1 recall (AvP01). According to (Eichmann et al., 1998), the effectiveness within the high precision area is measured assuming that users are most interested in getting relevant documents ranked topmost in the result list. Because this number can

[3] http://trec.nist.gov/

vary substantially for different queries, we consider also the precision figure for the topmost documents retrieved (in column four). There we focus on the precision after the top 10 documents (P10).

4.1 Monolingual Evaluation Runs

For the retrieval experiments we used the commercial *relevancy* information retrieval system from Eurospider Information Technology AG. It is a vector-based retrieval system that can handle large document collections. In regular deployment this system extracts word tokens from documents and queries and indexes them using a straight *lnu.ltn* weighting scheme (for the theoretical background of this scheme see (Schäuble, 1997)).

For the MuchMore evaluation runs we adapted the *relevancy* system so that it indexes the information provided by the XML annotated documents and queries: word forms (tokens) and their base forms (lemmas) for all indexable parts-of-speech both for German and English. The indexable parts-of-speech encompass all content words, i.e. nouns (including proper names and foreign expressions), adjectives, and verbs (excluding auxiliary verbs). All semantic information was indexed in separate categories each.

For each language, we produced a baseline performance by indexing only the tokens in both the documents and the queries. We call the German baseline DE-token. In addition an evaluation run based on linguistic stemming was produced which we termed DE-lemma. In table 1 we present the results of the monolingual German retrieval experiments.

In the baseline experiment for German (DE-token) the system finds only 322 relevant documents (out of 956). The mean average precision is thus low (mAvP = 0.16), but the average precision in the top ranks is acceptable (AvP = 0.56). So, the few documents that are found are often ranked at the top of the list. On average there are 4.16 relevant documents among the 10 top ranked documents (P10).

The importance of good linguistic stemming and decompounding is shown by the second experiment (DE-lemma), which achieves a recall gain of 70% compared to DE-token. In parallel, the precision figures have improved substantially. Lemmatization was done in two steps. First we used a general-purpose (i.e. general vocabulary) morphological analyzer. It turned out that many medical terms were not lemmatized since they were not in the analyzer's lexicon. Therefore we developed heuristics for treating words that were unknown to the analyzer. Through these heuristics unknown adjectives were lemmatized by suffix truncation (*arthroskopischen* → *arthroskopisch*), and unknown nouns were decompounded if both compound parts were found as separate words in the corpus (*Nociceptinspiegel* → *Nociceptin Spiegel*). In this way the corpus itself was used as domain specific lexicon for decompounding.

We also experimented with a combination of token and lemmas. Both were combined as indexing terms of equal weights in the queries and the documents. This combination leads to a decrease in precision (see DE-token-lemma) and therefore the tokens were discarded in the subsequent runs.

The impact of the different types of semantic information was determined one by one, but always in combination with lemmas. We wanted to support the hypothesis that semantic information will improve the precision over pure lemma information. The results show that the MeSH codes are the most useful indexing features whereas the EuroWordNet terms (EWN), without disambiguation in our current experiments (!), are the worst. Using MeSH codes increases recall (from 591 to 601) and also average precision (from 0.2809 to 0.2873). As was to be expected the very specific semantic relations (Semrel) have hardly any impact. Using the EuroWordNet terms in combination with the lemmas degrades the overall performance.

4.2 Cross-language Evaluation Runs

The easiest approach to CLIR is monolingual retrieval over a parallel corpus. This means that we would search German documents with a German query and simply display those English documents that are known to be correspondences of the found German documents. Our approach however is different. Instead, we assume that we have a document collection (i.e. a corpus) in one language and a query in another language. For the cross-language evaluation runs we used German queries to retrieve English documents.

A rough baseline for the cross-language task is

	mAvP	Rel. Docs Retr.	AvP 0.1	P10
DE-token	0.1600	322	0.5622	0.4160
DE-lemma	0.2809	591	0.6759	0.5320
DE-token-lemma	0.2547	594	0.6744	0.5120
DE-lemma-EWN	0.2414	584	0.6140	0.4880
DE-lemma-MeSH	0.2873	601	0.6647	0.5280
DE-lemma-Semrel	0.2795	591	0.6474	0.5200

Table 1: Results of the monolingual German runs

to use the tokens of the German queries directly for retrieval of the English documents. The idea is that the overlap in technical vocabulary between these languages will lead to relevant documents. And indeed, this approach finds 66 relevant documents (cf. DE2EN-DE-token in table 2).

It might be surprising that the overlap in technical vocabulary does not carry further than merely 66 documents. But one must consider that often the roots of the words are identical but the forms do not match because of differences in spelling and inflection (e.g. *arthroskopische* vs. *arthroscopic*). Stemming combined with some letter normalization (e.g. $k = c = z$) would lead to an increased recall, but has not been explored here.

As a second baseline we investigated the use of Machine Translation (MT) for translating the queries. We employed the latest version of the PC-based system PersonalTranslator (PT2002; linguatec, Munich) to automatically translate all queries from German to English. PersonalTranslator allows to restrict the subject domain of the translation, and we selected the domains medicine and chemistry. This restriction helps the system to choose the subject-specific interpretation if multiple interpretations for a given lexical entry are available. Still many translated queries are incomplete or incorrect but they scored surprisingly well with regard to recall. In table 2, line DE2EN-MT-PT2002, we see that the translated queries lead to 440 relevant documents at a rather low mean average precision of 0.1381.

Now we compare these results with the semantic codes annotated in our corpus and queries. This means we are using the semantic annotation of the German queries to match the semantic annotation of the English documents. One could say that we are now using the semantic annotation as an interlingua or intermediate representation to bridge the gap between German and English. The third block in table 2 has all the results. Again MeSH terms lead to the best results with respect to recall and precision. EuroWordNet leads to the worst precision and the semantic relations have only a minor impact due to their specificity. If we combine all semantic information, we reach 404 relevant documents and a mean average precision of 0.1774. This precision clearly exceeds machine translation.

For the last two experiments we built a similarity thesaurus (SimThes) over the parallel corpus (Qui, 1995). Our similarity thesaurus contains German words (adjectives, nouns, verbs) from our corpus, each accompanied by a set of 10 English words that appear in similar contexts and are thus similar in meaning. The building of the similarity thesaurus can be understood as exchanging the roles of documents and terms in document retrieval. The documents now represent the indexing features and the terms are the retrievable items. (Schäuble, 1997) contains the technical details. In building a bilingual similarity thesaurus over a parallel corpus the term sets of two parallel documents are exchanged. Given a term from the source language we then compute similar terms from the target language.

If a similarity thesaurus is built over a monolingual corpus, it may serve for query expansion in monolingual retrieval. In our case we built the similarity thesaurus over the parallel corpus. We were interested in German words and their similar counterparts in English. Each German word from the queries was then substituted by the English words of its similarity set. This resulted in the retrieval of 409 relevant documents and a relatively good mean av-

	mAvP	Rel. Docs Retr.	AvP 0.1	P10
DE2EN-DE-token	0.0512	66	0.1530	0.1160
DE2EN-MT-PT2002	0.1381	440	0.3747	0.2920
DE2EN-EWN	0.0090	111	0.0311	0.0160
DE2EN-MeSH	0.1699	304	0.3888	0.2600
DE2EN-Semrel	0.0229	23	0.0657	0.0480
DE2EN-all-combined	0.1774	404	0.3872	0.2720
DE2EN-SimThes	0.2290	409	0.4492	0.3640
DE2EN-SimThes+all-comb.	0.2955	518	0.5761	0.4600

Table 2: Results of the cross-language runs: German queries and English documents

erage precision of 0.2290 (see DE2EN-SimThes in table 2). Finally we checked the combination of all semantic annotations with the similarity thesaurus. Each query is now represented by its EuroWordNet, MeSH and semantic relations codes as well as by the words from the similarity thesaurus. This combination leads to the best results for CLIR. We retrieved 518 relevant documents with a mean average precision of 0.2955 (cf. the last line DE2EN-SimThes+all-combined in table 1). And the figures for the high precision area (AvP and P10) are also outstanding. This result is approximating the results of monolingual retrieval with tokens, lemmas and semantic annotation.

5 Conclusions

We have explored the use of different kinds of semantic annotation for both monolingual and cross-language retrieval.

In monolingual retrieval (for both English and German) semantic information from the MeSH codes (Medical Subject Headings) were most reliable and resulted in an increase in recall and precision over token and lemma indexing. Moreover, the monolingual experiments show that high-quality linguistic analysis is crucial for a good retrieval performance.

In cross-language retrieval the combination of all semantic information outperformed machine translation with respect to precision. It was only superseded by the use of a similarity thesaurus built over the parallel corpus where we used 10 similar words of the target language for each source language word. This means we included query expansion in combination with translation.

The highest overall performance resulted from a combination of the similarity thesaurus with the semantic information. This result is comparable to the German monolingual retrieval results in terms of precision but still 14% lower in the number of relevant retrieved items.

References

P. Buitelaar and B. Sacaleanu. 2001. Ranking and selecting synsets by domain relevance. In *Proc. Of NAACL WordNet Workshop*, Pittsburgh.

J. Carbonell, Y. Yang, R. Frederking, R. D. Brown, Y. Geng, and D. Lee. 1997. Translingual information retrieval: A comparative evaluation. In *Proc. Of the Fifteenth International Joint Conference on Artificial Intelligence*.

D. Eichmann, M. Ruiz, and P. Srinivasan. 1998. Cross-language information retrieval with the UMLS metathesaurus. In *Proc. Of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia.

F.C. Gey and H. Jiang. 1999. English-german cross-language retrieval for the GIRT collection - exploiting a multilingual thesaurus. In *Proc. of the Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, MD. National Institute of Standards Technology (NIST).

W. Kraaij and D. Hiemstra. 1998. TREC6 working notes: Baseline tests for cross language retrieval with the twenty-one system. In *TREC6 Working Notes*, Gaithersburg, MD. National Institute of Standards and Technology (NIST).

D. Oard. 1998. A comparative study of query and document translation for cross-lingual

information retrieval. In *Proc. of AMTA*, Philadelphia, PA.

Y. Qui. 1995. *Automatic Query Expansion Based on a Similarity Thesaurus*. Phd thesis, ETH Zurich.

M. Sanderson. 1994. Word sense disambiguation and information retrieval. In B. Croft and K. Van Rijsbergen, editors, *Proc. of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 142–151. Springer-Verlag.

Peter Schäuble. 1997. *Multimedia Information Retrieval. Content-based Information Retrieval from Large Text and Audio Databases*. Kluwer Academic Publishers, Boston.

S. Vintar, P. Buitelaar, B. Ripplinger, B. Sacaleanu, D. Raileanu, and D. Prescher. 2002. An efficient and flexible format for linguistic and semantic annotation. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Canary Islands, Spain, May 29-31.

Piek Vossen. 1997. EuroWordNet: A multilingual database for information retrieval. In *Proc. Of the DELOS Workshop on Cross-Language Information Retrieval*. Zurich, Switzerland, March, 5-7.

Information Retrieval from Historical Corpora

Loes Braun and **Floris Wiesman** and **Ida Sprinkhuizen-Kuyper**
IKAT, Universiteit Maastricht, P.O.Box 616, 6200 MD Maastricht

With the increasing number of documents that are available in digital form, also the number of digital historical documents is increasing (Berkvens, 2001). It cannot be assumed that standard IR systems perform well on historical documents: historical texts differ from modern texts in three ways (Hüning, 1996; Van Der Horst and Marschall, 1989): (a) vocabularies have changed, (b) spelling has changed (sometimes to the extent that two variants of the same word are not even recognizable as such), and (c) spelling used to be highly inconsistent (in the Netherlands until the 19th century). The goals of this research were to identify the bottlenecks of information retrieval from historical corpora and to find solutions for these bottlenecks.

Section 1 describes our efforts to further identify the bottlenecks, section 2 examines potential solutions, and section 3 sketches our approach to alleviate the bottlenecks. Finally section 4 draws conclusions and provides directions for future research.

1 Experimental identification of bottlenecks

An experiment was conducted to establish the influence on IR performance of the above-listed differences between historical and moderns texts.

The corpus used was a collection of Dutch and Belgian law texts known as the *Antwerpse Compilatae* and the *Gelders Land- en Stadsrecht*, dating from the 16th and 17th century. The texts were split up into 393 documents, with a total of 371862 words. They were made available in electronic form partly by optical character recognition (OCR) and partly by manual data entry.

Six experts in the field of historical law, acquainted with the documents used, were asked to formulate five information needs each. The information needs were manually translated to queries and run on a basic vector-model IR system (with normalized tf.idf weights and a cosine similarity function). For each query, the relevance was judged — by the expert who formulated the corresponding information need — of each document with a similarity score ≥ 0.02. Furthermore, the experts were asked to point out, for each of their information needs, several documents which they considered relevant, but were not retrieved by the system.

Inspection of the relevant documents that were not retrieved by the system showed that these documents have at least one of two specific characteristics: (a) the documents contain one or more of the query terms but in a different spelling and (b) the documents do not contain the specific query terms but synonyms of them. The first characteristic is caused by the fact that the vector model may only consider a document relevant if it contains one or more of the exact query terms of the query under consideration. If, however, a document only contains *spelling variants* of one or more query terms, it is not considered relevant. We consider two words A and B spelling variants of each other, if they satisfy the following requirements: (a) they have the same meaning, (b) they have a different but allowed spelling, (c) they can be recognized as variants of the same word, and (d) they have the same form (e.g. nominative plural form). This means that already a one-character difference in a term can cause the vector model to label a document as non-relevant incorrectly. This is a bottleneck, since we wish to find all relevant documents, regardless of spelling inconsistencies. Although this bottleneck occasionally presents itself also in information retrieval with modern texts, in that case it is rarely caused by the use of different spelling variants of words, since modern words normally have less spelling variants. Spelling inconsistencies in modern texts are mostly due to different causes, for example OCR errors, which introduce spelling errors. In general however, the problem of spelling inconsistencies does not manifest itself regularly in information retrieval from modern texts. From the documents inspected, it is clear that in historical texts spelling inconsistencies occur on a considerably larger scale than they do in modern texts (see table 1), presumably because no spelling rules existed during the time the documents were written. Therefore the bottleneck can be considered as specific to information retrieval from historical texts.

Table 1: Word variants (var) and synonyms (syn) found in historical texts (i.e., the *Antwerpse Compilatae* and the *Gelders Land- en Stadsrecht*) and in modern texts.

	Historical texts		Modern texts	
Word	var	syn	var	syn
advocaat	4	9	0	4
arbitrages	2	1	0	0
borgtocht	9	0	0	4
dadingen	4	0	0	3
gehuwd	3	2	0	1
notaris	2	4	0	0
panding	1	0	0	1
persoon	0	2	0	0
procederen	2	1	0	1
procureur	1	1	0	0
rechter	5	0	0	1
rente	4	5	0	1
secretaris	2	5	0	0
termijn	3	0	0	0
verbintenissen	6	0	0	0
vonnis	7	2	0	2
vrouw	1	2	0	10

The second characteristic is caused by the inability of the vector model to recognize synonyms of a query term. We consider two words A and B synonyms, if they satisfy the following requirements: (a) they have the same meaning, (b) they have a different but allowed spelling, (c) they cannot be recognized as variants of the same word, and (d) they have the same form (e.g. nominative plural form). Therefore documents containing one or more synonyms of a query term, but not the exact query term, are incorrectly considered non-relevant. This problem can obviously also be observed in information retrieval from modern texts, because a modern vocabulary also contains synonyms. However, from the documents inspected it seems reasonable to state that in a modern vocabulary, words have less synonyms than in a historical vocabulary. (see table 1). For only two words from table 1 ('vrouw' and 'dadingen'), the modern vocabulary contains more synonyms than the historical vocabulary. Therefore, this problem manifests itself on a considerably larger scale in historical texts, and might consequently be considered a problem specific to information retrieval from historical texts.

The results of the inspection of the documents are shown in table 1. In this table, for 17 terms from the queries used to determine the bottlenecks, the number of spelling variants found in both historical and modern texts, and the number of synonyms found in both historical and modern texts are shown. The numbers of spelling variants and synonyms found in historical documents are based on the inspection of the documents in the historical collections under consideration. The numbers of spelling variants and synonyms found in modern texts are based on data from a thesaurus of modern Dutch (Sterkenburg, 1996). It is important to note that, in table 1, there is a difference between the origin of the results for modern texts and historical texts. The numbers of spelling variants and synonyms found in historical texts are only *minima*, since we based these results on only a limited collection of documents. However, the numbers of spelling variants and synonyms found in modern texts are *maxima*, since these results are based on a thesaurus, containing all spelling variants and synonyms. Therefore, the differences between the number of spelling variants and synonyms in historical texts and modern texts are probably even greater than in table 1.

From the results of the inspection of the documents, we conclude that there are two main bottlenecks in information retrieval with historical texts:

- Varying and inconsistent spelling: the *spelling* bottleneck
- Use of a different vocabulary: the *vocabulary* bottleneck

2 Potential solutions

The bottleneck of the changed vocabularies has a straightforward solution: the use of a thesaurus. Nevertheless, this solution was not implemented, because no thesaurus suitable for the text corpus used in this research existed and constructing one was not feasible because of the limited availability of the experts. Consequently, it was decided to concentrate on the changed and inconsistent spelling. The use of *conflation procedures* (Robertson and Willett, 1992) seemed to be be the most promising approach. Such a procedure can match different forms of the same word. With conflation, documents can be retrieved even if the exact query term does not occur in the document but an equivalent word form does. Various conflation procedures exist, such as n-gram matching, stemming, and dynamic programming. Stemmers are highly language specific; fortunately a stemming algorithm for modern Dutch is available (Kraaij and Pohlmann, 1994). To make it usable for historic Dutch we devised several heuristics for transforming old word forms into modern ones. Subsection 2.1 explains the transformation heuristics and subsection 2.2 assesses the various conflation procedures.

2.1 Transformation heuristics

The purpose of the application of the heuristics was *not* to transform the old word forms completely into modern ones. Instead, the main goal was to fine tune the word forms to the Dutch stemming algorithm, which would possibly be applied after the ap-

plication of the heuristics. Consequently, the heuristics were strongly based on the rules applied by the Dutch stemming algorithm. This algorithm transforms words into their stems based on their suffixes, prefixes, and infixes. The old word form 'uuytrichtinghe' has, for example, the suffix 'inghe', the prefix 'uuyt' and the infix 'richt'. However, the suffixes, prefixes and infixes of old word forms might not exactly match those required by the stemming algorithm. Therefore, the algorithm will not work on old word forms. By using heuristics, however, old word forms can be transformed in such a way that their suffixes, prefixes, and infixes *can* be matched and transformed by the stemming algorithm. The old word form 'uuytrichtinghe' can, for example, be transformed into 'uitrichtingen'.

We developed three kinds of heuristics: (a) heuristics for suffix transformation, (b) heuristics for prefix transformation, and (c) heuristics for infix transformation. The reason for the development of three specific sets of heuristics was that the transformation of a specific combination of characters is dependent on whether this combination of characters is a suffix, a prefix, or an infix. An example is the heuristic for suffix transformation 'ff' → 'f', which indicates that the characters 'ff' have to be transformed into 'f' when they form the suffix of a word. If they form the prefix of an infix of a word, however, they cannot always be transformed.

The heuristics for suffix and prefix transformation were furthermore divided into two subsets of rules. The reason for this division is that after its transformation, a suffix (or a prefix) might require even further transformation. Therefore there are two sets of rules which were applied one at a time in a specific order to ensure that the steps in which a suffix (or a prefix) has to be transformed were executed correctly. Consider, for example the application of the two heuristics for suffix transformation (a) 'liijck' → 'lijk' and (b) 'ck' → 'k' to the old word form 'mogelijck'. If heuristic (b) is applied before heuristic (a), the word 'mogeliijck' will first be transformed into 'mogeliijk'. Then, however, application of heuristic (a) is no longer possible, since the suffix 'liijck' can not be matched. Consequently, the word remains 'mogeliijk'. Because the suffix 'liijk' is, however, not common in modern Dutch this word unfortunately cannot be stemmed by the Dutch stemming algorithm. If the heuristic (a) is applied first, however, the word 'mogeliijck' is first transformed into 'mogelijk'. Then, application of heuristic (b) is no longer possible for the same reason the application of heuristic (a) was no longer possible after the application of heuristic (b). In this case, however, application of heuristic (b) is no longer *needed*, since the word 'mogelijk' has a suffix that *is* common in modern Dutch, and consequently *can* be matched by the Dutch stemming algorithm.

Each of the subsets of the heuristics for suffix and prefix transformation was developed in such a way that there is always only one rule which can be applied. Consider for example the heuristics for prefix transformation 'uuy' → 'ui' and 'uu' → 'ui'. Since the prefix 'uu' is itself a prefix of the prefix 'uuy', both heuristics are applicable to every word with the prefix 'uuy', for example the word 'uuytspraeck'. Since every subset of heuristics can only contain one applicable heuristic, these two heuristics were put into different subsets.

Unlike the heuristics for suffix and prefix transformation, the heuristics for infix transformation were gathered into *one* set of rules. The reason for this difference is that a word can have multiple infixes which all need to be transformed. Consequently, more than one rule from this set might be applied. Consider, for example, the old word form 'teeckeninghe'. This word form contains the infixes 'eeck' and 'gh', which both need transformation. Therefore both the heuristics 'eeck' → 'eck' and 'gh' → 'g' have to be applied. As with suffix and prefix transformation however, the order in which the transformation process is executed is highly important. Consider, for example, the heuristics 'lick' → 'lijk' and 'ck' → 'k'. If the second heuristic is applied before the first, the old word form 'mogelickheyt' will be transformed into 'mogelikheyt' instead of 'mogelijkheyt'. Therefore the rules are applied in a specific order, which is always the same.

2.2 Assessment of conflation procedures

To be able to choose the best of various conflation procedures (n-gram matching, stemming, dynamic programming, and transformation heuristics), or a combination thereof, an experiment was conducted. Based on the results of Robertson and Willett (1992) we chose $n = 3$ for n-gram matching and as a particular dynamic programming approach the Wagner-Fischer algorithm (Wagner and Fischer, 1974).

The setup of this experiment was as follows. All query terms were extracted from the queries of each expert (see section 1). For each of these query terms, variants of the term were retrieved from the complete list of document terms extracted from the documents in the collection under consideration. The retrieval of the variants of each query term was done once by every method to be tested. For each method, this resulted in a list of document terms for each query term. If a document term actually was a variant of the query term, it was labeled as relevant. If it was not, it was denoted as non-relevant.

The experiment described above was run on several, but not all combinations of the four potential solutions. Trigram matching and the Wagner-Fischer algorithm were not combined, because this technique would transform the old word forms to

an extent that they cannot be effectively matched with modern word forms. Trigram matching and the Wagner-Fischer algorithm can, however, be effectively combined with both stemming and the use of heuristics. Consequently, we tested eight combinations of solutions: (a) trigram matching only, (b) trigram matching combined with stemming, (c), trigram matching combined with preprocessing by applying heuristics, (d) trigram matching combined with both stemming and preprocessing, (e) the Wagner-Fischer algorithm only, (f) the Wagner-Fischer algorithm combined with stemming, (g), the Wagner-Fischer algorithm combined with preprocessing by applying heuristics, and (h) the Wagner-Fischer algorithm combined with both stemming and preprocessing. For the results of each of these eight methods, the comparative recall and the precision were computed (see table 2).

From the results of the experiment, shown in table 2, it is clear that the retrieval methods involving the Wagner-Fischer algorithm reach only a low precision. Therefore we chose not to use one of these methods, despite their reasonably high recall. The retrieval methods involving trigram matching, on the contrary, do reach a high precision in combination with a high recall. We should note that the low precision values of the Wagner-Fischer algorithm are primarily due to the large number of terms retrieved by this method. The number of terms retrieved can be decreased by introducing a similarity threshold. However, decreasing of the number of retrieved terms, does not result in higher precision values, since the *relevant* terms found by the Wagner-Fischer algorithm are not the terms *most similar* to the query term. Therefore, when decreasing the number of retrieved terms, also the number of retrieved *relevant* terms is decreased and consequently, the precision stays at a low level.

Although table 2 indicates the superiority of trigram matching over the Wagner-Fischer algorithm, it is not instantly clear which of the four trigram matching methods performs best. We therefore calculated the average precision of each of these four methods at 11 standard recall levels (0% to 100%). To do this, the resulting terms of each query are inspected separately, in order of decreasing relevance. For each new term inspected the recall and the precision are calculated. Suppose, for example, that a query q has ten relevant terms. Furthermore, suppose that we have inspected the four most relevant terms resulting from the query, two of which are relevant and that the fifth term inspected is relevant too. The recall for the fifth term is then 30%, since three of the ten relevant documents have been retrieved. The precision for the fifth term is 60%, since three out of the five documents inspected are relevant. The precision at a recall level of 30% is therefore 60%. When proceeding as described above, recall and precision can only be calculated with respect to one particular query. To determine the average precision at each recall level over all queries ran on the system, equation 1 should be used.

$$\overline{P}(r) = \sum_{i=1}^{N_q} \frac{P_i(r)}{N_q} \tag{1}$$

$\overline{P}(r)$ is the average precision at recall level r, $P_i(r)$ is the precision at recall level r for query i, and N_q is the number of queries. The results of these calculations are shown in figure 1. This figure shows that the term matching performance of each of the four trigram matching methods is better than the term matching performance of the vector model.

To determine which of these four trigram matching methods performs best over the vector model, we performed eleven *Mann-Whitney tests* to compare the data from which the average precision at recall level r is computed with the data from which the average precision at the same recall level r is computed in the vector model. The test statistics (z–values) of these tests are shown in the first eleven rows of table 3. The p–values in these tables, which are derived from the test statistics, denote the probability that the average precision at recall level r in a specific trigram matching method equals the average precision at the same recall level r in the vector model. For example, the probability that the average precision at a recall level of 0% for trigram matching combined with stemming and preprocessing does not differ from the average precision of the vector model at a recall level of 0% is 32.74%. Furthermore, for each of the trigram matching methods, we performed a Mann-Whitney test to compare the average precision values for *all* recall levels r of the trigram matching method with the average precision values for *all* recall levels r of the vector model. The results of these tests are shown in the last row of table 3.

Table 2: Results of the performance assessment.

Retrieval method	Comp. recall	Precision
Tri	70.4	57.9
Tri – stem	74.0	62.5
Tri – prepro	74.8	53.7
Tri – stem – prepro	82.1	57.8
Wag	67.2	12.1
Wag – stem	70.6	13.2
Wag – prepro	73.8	11.9
Wag – stem – prepro	77.4	12.1

Legend: Tri: Trigram matching, Wag: Wagner-Fischer, Stem: stemming, Prepro: preprocessing

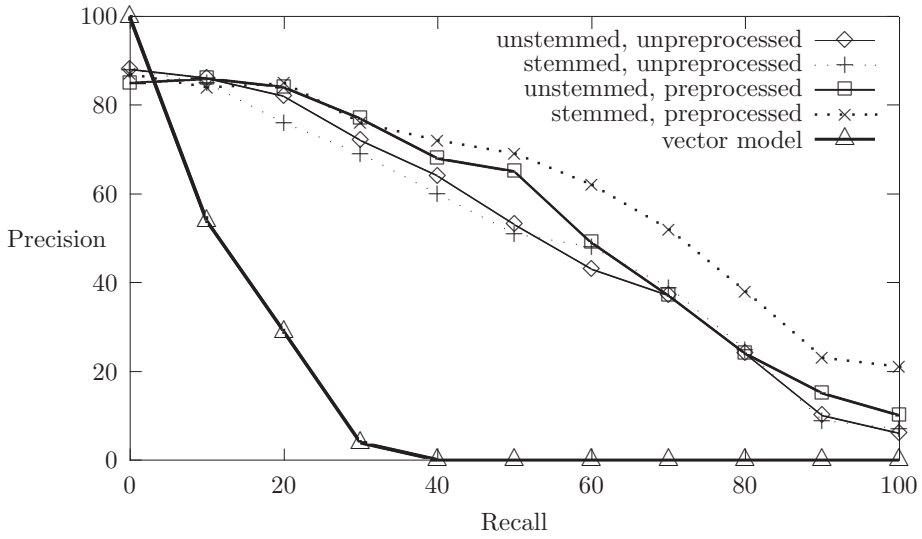


Figure 1: Average recall vs. average precision of various trigram matching methods and the vector model.

Table 3: Results of Mann-Whitney tests of various trigram matching methods vs. the vector model.

Recall level in %	Stemmed, preprocessed		Stemmed, unpreprocessed		Unstemmed, preprocessed		Unstemmed, unpreprocessed	
	p	z	p	z	p	z	p	z
0	0.327	0.979	0.464	0.732	0.464	0.732	0.327	0.979
10	0.057	1.907	0.054	1.928	0.070	1.815	0.054	1.928
20	0.000	3.588	0.004	2.846	0.002	3.093	0.001	3.413
30	0.000	4.866	0.000	3.918	0.000	4.588	0.000	5.083
40	0.000	4.938	0.000	3.701	0.000	4.196	0.000	4.938
50	0.000	4.691	0.001	3.206	0.001	3.454	0.000	4.691
60	0.000	4.196	0.001	3.206	0.003	2.959	0.001	3.454
70	0.000	3.701	0.027	2.217	0.007	2.712	0.007	2.712
80	0.003	2.959	0.140	1.474	0.049	1.969	0.049	1.969
90	0.014	2.464	0.327	0.979	0.327	0.979	0.085	1.722
100	0.027	2.217	0.464	0.732	0.464	0.732	0.220	1.227
Total	0.005	2.824	0.009	2.627	0.009	2.627	0.007	2.692

The statistical results in the first eleven rows of table 3 show that at nine of the eleven recall levels, the trigram matching method combined with stemming and preprocessing has a test statistic which is *at least as high* as the test statistics of the other trigram matching methods. At eight of these nine recall levels, this test statistic is higher than 1.960, which means that, at these levels, the probability that the results of the vector model equal the results of trigram matching method combined with stemming and preprocessing is less than 5%. Furthermore, the results in the last row show that the p–values of *all* trigram matching methods are higher than 1.960, meaning that the results of these methods are significantly better than those of the vector model when the average precision values at all recall levels are compared. The trigram matching method combined with stemming and preprocessing, however, has the *highest* test statistic (of 2.824) of all four trigram matching methods. This result means that the probability that the results of the vector model equal those of the trigram matching method combined with stemming and preprocessing, is less than 0.5%.

We also performed Mann-Whitney tests to compare the different trigram matching methods to each other. However, these test showed no significant difference between the performance of the trigram matching methods. Because trigram matching combined with stemming and preprocessing nevertheless

performs best over the vector model, we chose to use this method as our final solution.

3 Assessment of the new system

To determine the performance of our new system (i.e., stemmed preprocessed trigram matching) we compared its document retrieval performance with that of the standard vector-based system. The experts of section 1 formulated 25 information needs, which were transformed into queries. The procedure used for document assessment for the newly developed system was performed in almost the same way as in section 2.2. The experts were asked to assess all retrieved documents and to point out documents they deemed relevant, but which were not retrieved by the system.

Figure 2 shows that at nine of the eleven standard recall levels, the average precision of our new system is at least as high as the average precision of the standard system. Consequently, from figure 2 the performance of our new system seems to be better than the performance of the standard system.

We compared the average precision values for *all* recall levels r of our new system with the average precision values for *all* recall levels r of the standard system by performing a Mann-Whitney test. With a test statistic of 2.1013 and $p = 0.0356$, the difference was significant.

4 Conclusions

We have shown that there are two main bottlenecks in information retrieval from historical corpora: (a) the spelling bottleneck and (b) the vocabulary bottleneck.

In our experiments we focused on alleviating the spelling bottleneck, in particular by using conflation procedures. The ability of matching word forms was tested for various conflation procedures: trigrams, stemming, preprocessing, and dynamic programming. Preprocessing consisted of applying heuristics to map old word forms to modern ones so as to make the words suitable for stemming with a modern stemmer. The most promising combination turned out to be trigrams with preprocessing and stemming. Subsequently, this new system was compared to a standard vector-based system in an information retrieval task. The performance of the new system was significantly better.

The two main directions for future research concern language specificity and domain specificity. First, the document collections used in our research contain only Dutch texts. Therefore, our new system uses heuristics specifically designed for Dutch texts, which will probably cause the system to perform worse on texts written in other languages. There should be no obstacle, however, to formulate similar heuristics for other languages. It should be investigated to what extent other languages can benefit from our approach.

Second, the documents used in our research are from fairly specific and small document collections. As the language specificities, these domain specificities are also implicitly implemented in the system. Hence, research should be performd with other document collections, preferably from a period other than the 16th and 17th century.

References

A.M.J.A. Berkvens. 2001. Digitalisering: Een virtuele toekomst voor oude rechtsbronnen. *Pro Memorie. Bijdragen tot de Rechtsgeschiedenis der Nederlanden*, 3(1):128–140.

Matthias Hüning. 1996. Geschiedenis van het Nederlands. Vakgroep Nederlands, Universiteit Wenen. http://www.ned.univie.ac.at/publicaties/-taalgeschiedenis/nl/.

W. Kraaij and R. Pohlmann. 1994. Porter's stemming algorithm for dutch. In Noordman L.G.M. and W.A.M. de Vroomen, editors, *Informatiewetenschap 1994: Wetenschappelijke bijdragen aan de derde STINFON Conferentie*, pages 167–180, Tilburg.

Alexander M. Robertson and Peter Willett. 1992. Searching for historical word-forms in a database of 17th-century english text using spelling-correction methods. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, IR Applications*, pages 256–265. ACM Press.

P.G.J. Sterkenburg. 1996. *Groot woordenboek van synoniemen en andere betekenisverwante woorden*. Van Dale Lexicografie, Utrecht, The Netherlands.

J. Van Der Horst and F. Marschall. 1989. *Korte Geschiedenis van de Nederlandse Taal*. Sdu Uitgevers, Den Haag, The Netherlands.

R.A. Wagner and M.J. Fischer. 1974. The string-to-string correction problem. *Journal of the ACM*, 21:168–173.

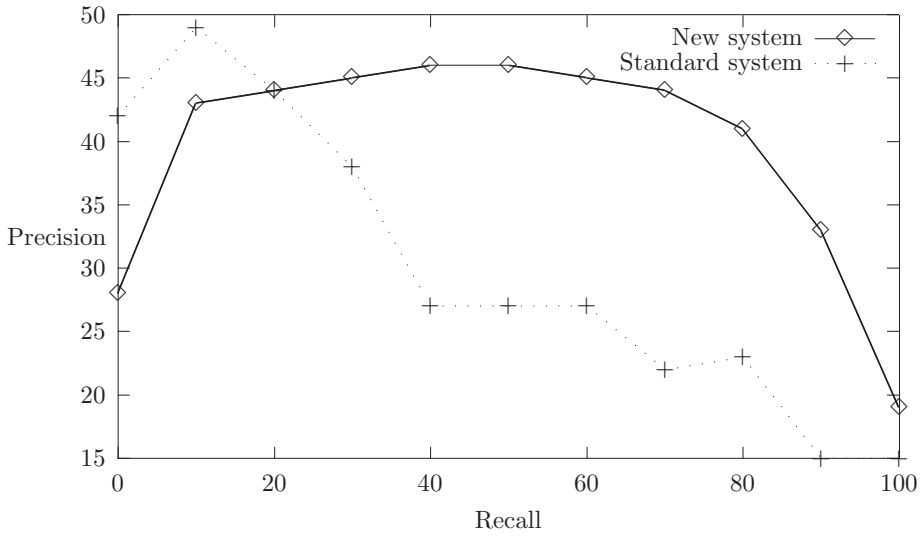


Figure 2: Average recall vs. average precision of the standard system and the new system.