Twenty-One at TREC-8: using Language Technology for Information Retrieval

*Wessel Kraaij, *Renée Pohlmann and †Djoerd Hiemstra

*TNO-TPD
P.O. Box 155, 2600 AD Delft
The Netherlands
{kraaij,pohlmann}@tpd.tno.nl

†University of Twente, CTIT
P.O. Box 217, 7500 AE Enschede
The Netherlands
hiemstra@cs.utwente.nl

Abstract

This paper describes the official runs of the Twenty-One group for TREC-8. The Twenty-One group participated in the Ad-hoc, CLIR, Adaptive Filtering and SDR tracks. The main focus of our experiments is the development and evaluation of retrieval methods that are motivated by natural language processing techniques. The following new techniques are introduced in this paper. In the Ad-Hoc and CLIR tasks we experimented with automatic sense disambiguation followed by query expansion or translation. We used a combination of thesaurial and corpus information for the disambiguation process. We continued research on CLIR techniques which exploit the target corpus for an implicit disambiguation, by importing the translation probabilities into the probabilistic term-weighting framework. In filtering we extended the the use of language models for document ranking with a relevance feedback algorithm for query term reweighting.

1 Introduction

Twenty-One[1] is a project funded by the EU Telematics programme, sector Information Engineering. The project subtitle is "Development of a Multimedia Information Transaction and Dissemination Tool". Twenty-One started early 1996 and was completed in June 1999. Because the TREC ad-hoc and CLIR tasks fitted our needs to evaluate the system on the aspects of monolingual and cross-language retrieval performance, TNO-TPD and University of Twente participated under the flag of "Twenty-One" in TREC-6 and TREC-7. Because the cooperation is continued in other projects: Olive and Druid we have decided to continue our TREC participation as "Twenty-One". For the Ad-Hoc, CLIR and SDR tasks, we used the TNO vector space engine. The engine supports several term-weighting schemes. The principal term weighting scheme we used is based on statistical language models (LM). Cf. [10] and the appendix for a more detailed description of the baseline system.

2 The Ad Hoc task

2.1 Expansion of title queries

For TREC-8 we decided to focus our experiments on title queries, because they correspond better to the average queries of current IR system users. For title queries, query expansion seems an obvious technique to improve retrieval effectiveness. We have experimented with techniques to use a lexical thesaurus for query expansion. The thesaurus is part of the VLIS lexical database of Van Dale publishers. Query expansion involves a series of steps:

1. POS tagging and lemmatization of each query word

[1]Information about Twenty-one, Olive and DRUID is available at http://dis.tpd.tno.nl/

2. Lookup of the lemma in the VLIS lexical database. The result is a “Lexical Entity” (LE), which is a reference to a concept description in the VLIS database
3. Often the previous step results in a list of concepts, especially in the case of homonyms. In these cases some form of disambiguation is needed.
4. Expansion of the concepts with related concepts, e.g. synonyms and/or hyponyms
5. Realisation of the expansion concepts in English, using the VLIS translation relations

Unfortunately the technique did not yield convincing results on the TREC-7 topic set, so we decided to base the official TREC-8 runs on our TREC-7 system, without any of the newly developed techniques. However, these techniques were used in one of the official TREC-8 CLIR runs.

We will discuss one potential reason for the failure of query expansion techniques for title queries: One of the problems in developing a system tuned for title-only runs is the fact that the judgments for test collections are based on the full topics. The title field is a two or three word *summary* of the topic which is composed after the topic has been developed. Therefore such a title will always cover only a limited set of topic aspects. If different persons would interpret the title query, they would have different interpretations of the relevance of retrieved documents. That is because title queries are necessarily under-specified. But because the judgments have been done with the full topic description in mind (including detailed constraints when a document is - or is not - relevant) it is hard to devise a query expansion method which will improve average precision of a title run, because it is hard to predict which query constraints are described in the description and narrative sections of the topic.

2.2 Experimental setup and results

For the official runs in the ad-hoc task we eventually re-used our TREC-7 system, because the experimental query expansion systems scored significantly worse on the TREC-7 test collection. The TNO vector space engine was configured to use LM weighting using an λ_i of 0.15 and standard Porter stemming. Stop-words were removed from the documents, including words that are frequent in previous TREC topics like *relevant* and *document*. Queries were generated automatically from the full topics (title, description and narrative) using the same procedure as used for indexing.

The results of our official runs in the Ad Hoc tasks are given in table 1. We submitted 3 official runs, using either only the title or both the title and description fields. We compared a baseline run with a Pseudo Relevance Feedback(PRF) run. After an initial retrieval run, the top 200 of the weighted index terms extracted from a concatenation of the top 3 documents were added to the query with a ratio of 20 : 3, i.e. the weight of the added terms was multiplied by 3/20 before adding. These parameters were determined empirically by experimenting with the English topics of the TREC-6 CLIR track.

run-name	topic fields	mode	AVP
tno8d3	t+d	PRF	0.2921
tno8d4	t+d	base	0.2778
tno8t2	t	base	0.2423
tno8t3	t	PRF	0.2755

Table 1: Ad Hoc results

3 Cross Language Information Retrieval (CLIR)

3.1 Introduction

Like in previous years, our CLIR approach is based on query translation using bilingual dictionaries. A Twenty-One cross-lingual run consists of three steps:

1. Translate the topics in the three other languages (we used the English topics as source)
2. Perform 4 parallel runs on the sub-collections, with the translated topics
3. Merge the 4 runs into a final result file

Unlike the Ad Hoc task where we used Porter stemming, we used morphological stemming based on the Xelda tools of XRCE Grenoble for all languages [2]. We have some indications that the fact that the stemmer only removes inflectional affixes, results in reduced effectiveness. Experiments with a derivational version are planned. For German we experimented with several strategies to deal with compounds, which were initially developed for Dutch [16]. We eventually used a non-optimal strategy (i.e. the strategy which replaces a compound by its parts) because the optimal strategy (just add compound parts as index terms) interfered with the merging strategy (retrieval status values (RSVs) are not compatible). All CLIR runs used the fuzzy expansion procedure as described in [10] to catch spelling variants of proper nouns and typos.

3.2 Translation strategies

Query translation in Twenty-One is based on the VLIS lexical database developed by Van Dale Lexicography for translations into German and French and on Systran for the translation from English into Italian. We used three different strategies for selecting translations from the VLIS database: dictionary preferred, boolean and disambiguation. The dictionary preferred and boolean strategies were also used last year, the disambiguation strategy was developed for this year's participation.

3.2.1 Dictionary preferred

In the dictionary preferred translation strategy, the selection of translations is based on the number of occurrences of a certain translation in the dictionary. Some lemmas have identical translations for different senses. If this is the case, this translation is selected. If no translation occurs more than once, the first translation is chosen by default.

3.2.2 Boolean

For the boolean strategy, translations are weighted based on the number of occurrences in the dictionary. If a translation occurs in the dictionary under three senses we assign it a weight that is three times as high. As Dutch serves as an interlingua, translation can be carried out via several Dutch pivot lemmas. This possibly generates even more occurrences of the same translation. The implicit assumption made by weighting translations is that the number of occurrences generated from the dictionary may serve as rough estimates of actual frequencies in parallel corpora. Ideally, if the domain is limited and parallel corpora on the domain are available, weights should be estimated from actual data.

3.2.3 Word sense disambiguation

This year we also experimented with a word sense disambiguation technique for cross-language retrieval. In this technique, dictionary-based word senses are disambiguated using corpus information. First, the original query is used for monolingual retrieval on the TREC ad-hoc corpus. All terms in the top N documents produced by this run are saved. Subsequently, the LEs and all lexical realisations of query terms are looked up in the VLIS database. The semantic relations defined in VLIS are used to look up synonym, hyponym and hyperonym LEs of each different sense of a query term and their lexical realisations. In this way we gather a structured group of words associated with each particular sense of a query term. These groups are further expanded using words from example sentences which are also included in the database.

The groups of words for each possible sense of a query term are subsequently compared with the terms from the monolingual retrieval run and "evidence" for each sense is computed based on the overlap between the two sets of terms. The sense for which the most evidence is found is selected. If no evidence is found at all or all senses score equally, the first sense is selected by default.

[2] In TREC-7 we relied on the Porter stemmer for Italian, developed at ETH

LEs	hyperonym relations	synonym relations	hyponym relations
bank	concern undertaking business enterprise	house institute	banker deposit mortgage loan trade
bank	rise elevation mound		sandbank shoal aground stuck
pipe	object	duct funnel nozzle tube	supply drain eustachian
pipe		tobacco	peace clay water hookah opium

Table 2: example word groups

Query translation is now fairly straightforward. The translations for the selected word senses are looked up in the VLIS database, if more than one translation is given for a particular sense the boolean weighting strategy is applied (cf. section 3.2.2 above).

We experimented with different values of N for the initial monolingual retrieval run, 20 turned out to be the best choice. We also experimented with re-weighting words associated with a particular word sense based on their semantic relation to the original term, e.g. assign hyponyms a higher weight than hyperonyms. This experimentation provided some evidence that hyponyms are very important for sense determination but synonyms should possibly be excluded from the sense groups.

3.3 Merging Strategies

The merging strategies used for TREC-7 were a major performance bottleneck, because the merged runs scored about 75-80% of the averaged average precision of the constituting runs. We compared different merging strategies: i) naive merge: this means simply merging the result files, assuming that the RSVs are "compatible" ii) Rank based merge: This technique was applied by IBM at TREC-7 [6]. The assumption is that $log(R)$ where R is the rank number has a linear relationship with the probability of relevance. The method estimates the linear model on training data (e.g. previous TREC collection) e.g. by applying regression and simply replaces each RSV in a run by the estimated probability of relevance which is a function of the rank. iii) combination of evidence: just add the RSVs of method i and ii.

The LM term weighting model is founded in probability theory, but the RSV's in the implementation in the TNO vector space engine are not equivalent to the probability of relevance. The RSVs are actually a log of the probability of relevance offset by a query dependent constant and a collection dependent constant. For the naive merge method we divided the RSVs by the query length in order to compensate for differences in query length between different language versions of a topic[3] . The IBM merge strategy has the implicit assumption that all topics have a similar probability function of R for all languages. It's obvious that this assumption is not optimal, because the distribution of relevant topics over the different collections is not equal, with the extreme case that some topics only have relevant documents in 1 or 2 sub-collections. Our combined strategy simply sums the original RSV (which is the log of the probability of relevance, normalized on query length but offset by some unknown constant) with the estimated log of the probability of relevance at rank R. The method empirically scored well, probably because the IBM probability estimates help to map to a normalized RSV. There is some theoretical justification because the probability at rank R $P(D|R)$ can be used as an estimate for the a-priori probability that a document is relevant $P(D)$.

3.4 Results

Table 4 lists the results for our official runs. We discovered an error in the **tno8gr** merged run, it did contain no French documents. The tables list the results for the fixed tno8gr run. As a baseline we included **tno8mx**, a run which is based on a merge of 4 monolingual runs. We hoped to improve the pool with this run, in order to enable a better evaluation of the monolingual and bilingual intermediate runs. The best result is achieved by **tno8gr-fixed** the boolean run. The table also lists the results of the other merging alternatives. From our preliminary analysis we conclude that for the xlingual runs the naive score based merging strategy performs always better than the interleaved or rank based probability estimates strategy. The combination of evidence approach adds some extra improvement in most cases. The rank based merging strategy is based on precision at rank R estimates of the TREC7 tno7mx run. However, the TREC8 topic

[3] Only necessary for the merged monolingual run

run-name	description
tno8dpx	dictionary preferred translation of English query into 3 other languages; fuzzy expansion of each query term
tno8gr	probabilisticly interpreted boolean query of all possible translations of the English queries into 3 other languages ; fuzzy expansion
tno8dis	disambiguation and translation of English queries into 3 other languages; fuzzy expansion of each translated term
tno8mx	reference run: merged run of four monolingual searches; fuzzy expansion of each query term

Table 3: description of CLIR runs

run-name	combination of evidence official	interleave unofficial	naive unofficial	rank based unofficial
tno8dpx	0.2523	0.2049	0.245	0.2214
tno8gr-fixed	0.2789	0.2288	0.2763	0.2102
tno8dis	0.2407	0.1905	0.2355	0.1906
tno8mx	0.3226	0.3159	0.2763	0,2625

Table 4: mean average precision of CLIR runs

set has much less relevant English documents. This is probably the reason that the success of a pure rank based merging strategy is limited.

When we look at the results of the constituting runs (table 5), the results are more consistent than in TREC-7. In TREC-7 the best performing intermediate runs were the dictionary preferred runs, and the boolean run was the best merged run. In TREC-8 the boolean strategy has the best intermediate and merged average precision.

If we compare the cross-language runs with their monolingual counterparts on a per-query basis, there are a number of queries with very poor results for all three query translation strategies. We have identified some of the factors which contributed to this effect.

- The failure to recognize and translate phrases as a unit. This is especially detrimental for the English to German runs where English phrases have to be translated into German single word compounds, e.g. "World War" → "Weltkrieg", "armed forces" → "Bundeswehr" (query 61).
- Tagging errors, e.g. "arms" (weapons) was tagged as the plural of "arm" (body part) by the Xerox tagger (query 66).
- Because most words in query titles were capitalized, we decided to convert them to lower case to prevent the tagger from tagging all title words as proper nouns. This had the effect that those title words that were actually proper nouns were not tagged correctly, e.g. the proper name "Turkey" was translated as "Truthuhn" and "dindon" (bird) in German and French respectively (query 66).

Although the results with disambiguation were somewhat disappointing, we intend to continue our experiments with word sense disambiguation in the future. One possible improvement we intend to investigate would be to use the unique Lexical Entity identifiers provided by the VLIS database instead of actual words

run-name	avg.prec. english	avg.prec. french	avg.prec. german	avg.prec. italian	average over 4	merged	relat. to avg. (%)
tno8dpx	0.3130(m)	0.3319	0.2053	0.3017	0.2880	0.2523	88
tno8gr-fixed	0.3130(m)	0.3672	0.2511	0.3017	0.3080	0.2789	91
tno8dis	0.3130(m)	0.3099	0.1806	0.3017	0.2763	0.2407	87
tno8mx(m)	0.3130(m)	0.5510(m)	0.4100(m)	0.3620(m)	0.4090	0.3226	79

Table 5: per language performance and the effect of merging on 28 topics TREC-8, (m) indicates monolingual run

as a conceptual interlingua. Our current strategy has the disadvantage that after the monolingual disambiguation process, which reduces source language query terms to unique LEs, new ambiguities are introduced in the translation process when the LEs are realized as actual words in the target language.

Not surprisingly, since it was less well tested than the other two strategies which were also used last year, we also found that the disambiguation procedure contained a few omissions which resulted in the failure to translate query terms. We found that some LEs in the VLIS database did not have lexical realisations in all languages (i.e. so-called lexical gaps). In those cases the VLIS database suggests a less optimal translation. These translations were not found by the disambiguation procedure, however.

3.5 Pool validation

Judgements for the cross-language task are probably not as complete as the judgements for the other TREC tasks [10]. In this section we try to get an indication of how much of a problem the incomplete judgements actually are. For previous TREC CLIR task runs, we evaluated each run that contributed to the pool using relevance judgements both with (standard evaluation) and without the relevant documents that the run uniquely contributed to the pool.[4]. The difference between the two evaluations will give an idea of how reliable the collections are for future work.

run name	average precision		difference		unique
	unjudged	judged			rel.
98EITdes	0.1919	0.1962	0.0043	2.2 %	45
98EITful	0.2514	0.2767	0.0253	10.1 %	159
98EITtit	0.1807	0.1841	0.0034	1.9 %	27
BKYCL7AG	0.2345	0.2406	0.0061	2.6 %	44
BKYCL7AI	0.2012	0.2184	0.0172	8.6 %	120
BKYCL7ME	0.3111	0.3391	0.0280	9.0 %	164
RaliDicAPf2e	0.1405	0.1687	0.0282	20.1 %	176
TW1E2EF	0.1425	0.1569	0.0144	10.1 %	107
ceat7f2	0.1808	0.2319	0.0511	28.3 %	293
ibmcl7al	0.2939	0.3168	0.0229	7.8 %	135
lanl982	0.0296	0.0487	0.0191	64.5 %	140
tno7ddp	0.2174	0.2382	0.0208	9.6 %	152
tno7edpx	0.2551	0.2846	0.0295	11.6 %	109
umdxeof	0.1448	0.1610	0.0162	11.2 %	140
		max:	0.0511	64.5 %	293
		mean:	0.0205	14.1 %	129
		standard deviation:	0.0124	16.1 %	67

Table 6: TREC-7 pool validation

Table 6 shows the results of the pool validation experiment. On average, an unjudged run will have 0.02 higher average precision after judging. However, the difference may be much worse, up to 0.05 for ceat7f2. It might be possible to use information about the quality of the pool like the mean and standard deviation of the differences to define a confidence interval on the average precision of unjudged runs, but that goes beyond the scope of this paper. A complicating factor is how to handle the case where a pair of judged runs from the same group uniquely found a relevant document. The runs that show the maximum absolute difference (ceat7f2) and the maximum relative difference (lanl982) come from systems of which only one run was judged. For a research group that did not participate in TREC-7, the penalty for not being able to judge the run may therefore be higher than table 6 suggests.

Table 7 shows that the total number of judged documents is more or less the same as last year. However the average number of relevant documents per topic is lower than 100. This probably means that the quality of the pool has improved, which makes the collection more useful for per language comparisons.

[4]Thanks to Chris Buckley for proposing the pool validation experiment

collection	total docs.	judged docs.	relevant docs.	no hits in topic	judged fraction	judged docs.	relevant docs.	no hits in	judged fraction
english	242,866	8,973	956	59,63,66,75	0.0013	9,810	1,689	26,46	0.0014
french	141,637	5,751	578	76	0.0014	6,130	991	-	0.0015
german	185,099	4,098	717	60,75,76	0.0008	4,558	917	26	0.0009
italian	62,359	4,334	170	60,63,75,80	0.0024	3,062	501	26,44,51	0.0018
total	631,961	23,156	2,421	average:	0.0013	23,560	4,098	average:	0.0013

Table 7: CLIR task statistics (a) 28 topics TREC-8, (b) 28 topics TREC-7

4 Adaptive filtering

In the TREC-7 filtering task three important issues turned up [5]: 1) the initial threshold, 2) threshold adaption and 3) query reweighting. Setting the thresholds probably has the greater impact on perceived performance in terms of utility [21]. Once the threshold performs satisfactory, it is hard to improve upon the performance by query reweighting. Although we put a considerable amount of work in the threshold algorithms, the main objective of the Twenty-One participation was the development of relevance weighting algorithms for the linguistically motivated probabilistic model. Details of the probabilistic retrieval model can be found in the appendix of this paper.

4.1 Evaluation setup

For the filter track we used the experimental linguistically motivated probabilistic retrieval engine developed at the University of Twente. Initial document frequencies for term weighting were collected from the '87 to '91 editions of the Wall Street Journal (TREC CDs 1 and 2). We did not use the '92 editions because this data would not have been available in a real world application. The topics and The Financial Times documents were stemmed using the Porter stemmer and stopped using the Smart stop-list which was augmented with some domain-specific stopwords like 'document' and 'relevant'. We used title, narrative and description of the topics to build the initial profile. The controlled language fields of the Financial Times test collection were not used. We did not process the incoming documents in chunks. That is, document frequencies were updated for each incoming document; a binary decision was made directly for each incoming document; selected documents were immediately checked for relevance; thresholds and profiles were immediately updated after the relevance assessments. Unjudged documents were assumed to be not relevant. All selected documents were saved for future updating of thresholds and query profiles.

4.2 Setting the initial threshold

The linguistically motivated model ranks documents by the probability that the language model of the document generates the query (see the appendix). For ranking this is sufficient, but for binary selection of a document we need to answer the question "when is the probability high enough?". One way to answer this question is to relate the probability of sampling the query from a document to the probability that the query is the result of a random sample from the entire collection. Queries that have a high probability of being sampled from the collection (i.e. queries with common words), should receive a higher initial threshold than queries with a low probability of being sampled from the collection (i.e. queries with uncommon words). We might approximate the probability that the query $T_1, T_2, \cdots, T_n$ of length n is sampled from the collection as follows.

$$P(T_1 = t_1, T_2 = t_2, \cdots, T_n = t_n) = \prod_{i=1}^{n} \frac{df(t_i)}{\sum_t df(t)} \tag{1}$$

Initially only documents that generate the query with a much higher probability than equation 1 should be selected. The initial threshold might be set to select documents with probabilities that are more than 100.000 times higher than the probability of random selection. This does not result in a very high threshold, because words that appear only once in the Wallstreet Journal receive a probability smaller than 1 in 2 million according to equation 1 and the probabilities $P(T = t | D = d)$ of a term t given a document-id d are much higher for matching terms.

After rewriting the probability measures to their corresponding vector product weighting algorithms (see the appendix), the document frequencies in the initial threshold disappear. The vector product threshold that corresponds with the decision above is $\mathit{threshold} = n \log(1/(1-\lambda_i)) + c$, where $c = \log(100.000)$. This shows an interesting feature of the initial threshold. In its vector product form, the threshold is related to the relevance weights λ_i. High initial relevance weights result in a high initial threshold. Relevance weights were initialised as $\lambda_i = 0.5$ and were re-estimated after feedback.

4.3 Threshold adaption

The threshold adaption algorithm is the part of the system that uses the utility functions to optimise its performance. We simply decided to aim just below the optimum utility given the similarities of the documents that were selected be the system. Updating was done as follows.

1. recompute the similarities of all selected documents (because of changed document frequencies and changed relevance weights);
2. recompute the initial threshold (because of changed relevance weights) and add it to the selected documents like it was a non-relevant document;
3. rank the selected documents by their similarities and find the maximum utility max;
4. the new threshold will be the similarity of the lowest ranked document that has a utility of $\mathit{max} - 3$ when optimising for LF1 and $\mathit{max} - 1$ when optimising for LF2.

As long as the system does not find any relevant document, it will increase its threshold quite fast. In general, it will never lower its threshold again, although this might happen in practice because changed document frequencies and relevance weights sometimes change the ranking of selected documents.

4.4 Relevance weighting of query terms

Initially, when no information on relevant documents is available, each query term will get the same relevance weight $\lambda_i = 0.5$. So, initially we assume that the query profile is best explained if on average half of the query terms is sampled from relevant documents and the other half is sampled from the updated Wall Street Journal data. If a relevant document is available, we might be able to explain the query profile better. Query terms that occur often in the relevant document(s) are more likely to be sampled from the relevant document. They should get a higher relevance weight. Query terms that do not occur (often) in the relevant document(s) are more likely to be sampled from the Wall Street Journal data.

Notice that we cannot simply use the proportions of relevant and non-relevant documents that contain a query term to directly estimate the new relevance weight as is done in classical probabilistic models [19]. When searching for the best relevance weights, we have to take into account the term frequencies of terms in the relevant documents. A possible approach to relevance weighting is the EM-algorithm (expectation

E-step: $$m_i = \sum_{j=1}^{r} \frac{\lambda_i^{(p)} \cdot P(T_i = t_i | D_j = d_j)}{(1-\lambda_i^{(p)}) P(T_i = t_i) + \lambda_i^{(p)} P(T_i = t_i | D_j = d_j)}$$

M-step: $$\lambda_i^{(p+1)} = \frac{m_i + 1.5}{r + 3}$$

Figure 1: relevance weighting of query terms: EM-algorithm

maximisation algorithm [4]) of figure 1. The algorithm iteratively maximises the probability of the query $t_1, t_2, \cdots, t_n$ given r relevant documents $d_1, d_2, \cdots, d_r$. Before the iteration process starts, the relevance weights are initialised to their default values $\lambda_i^{(0)} = 0.5$, where i is the position in the query. Each iteration p estimates a new relevance weight $\lambda_i^{(p+1)}$ by first doing the E-step and then the M-step until the value of the relevance weight does not change significantly anymore. The M-step should be a maximum likelihood estimate according to its definition [4], but we used a Baysian update because a small number of relevant documents should not radically change the initial relevance weights.

4.5 Experimental results

Six official runs were submitted: three optimised for LF1 and three optimised for LF2. For both utility functions we did the same three experiments.

1. a baseline run that only uses the initial threshold setting and threshold adaption routines;
2. the same run as 1, but with relevance weighting of query terms;
3. the same run as 1, but using a very high initial threshold.

The high initial threshold experiments were done using the TNO vector engine under slightly different conditions. These two runs use the AP Newswire data for the initial estimation of document frequencies and a somewhat different stop list. We do not think that these slightly different conditions change the big picture of our evaluation results.

run name	description	LF1	LF2	prec.	recall
uttno8lf1	optimised for LF1	-9.30	4.86	0.242	0.240
uttno8lf1f	optimised for LF1; query reweighting	-7.28	7.10	0.243	0.251
uttno8lf1p	optimised for LF1; high initial threshold	-1.20	2.46	0.216	0.105
uttno8lf2	optimised for LF2	-12.96	4.80	0.232	0.254
uttno8lf2f	optimised for LF2; query reweighting	-9.12	6.60	0.237	0.254
uttno8lf2p	optimised for LF2; high initial threshold	-5.54	1.34	0.199	0.127

Table 8: adaptive filtering, official results averaged over topics

Table 8 lists the evaluation results of the official runs using four evaluation meaures: LF1, LF2, precision and recall averaged over topics. Recall and precision were calculated by assigning 0 % recall to topics with no relevant documents and assigning 0 % precision to topics with emtpy retrieved sets. Both baseline runs show a consistent improvement in the average utility and the average precision/recall after relevance weighting of query terms.

The high initial threshold run shows different behaviour. When optimising for LF1 (uttno8lf1p), the performance in terms of average LF1 utility improves considerably. At the same time, the performance in terms of precision and recall goes down. When optimising for LF2, a high initial threshold results in a system with lower performance than the baseline in terms of average utility, precision and recall.

4.6 Some thoughts on the evaluation

The problem with the LF1 utility is that it is plain too hard to build a system that does not perform below zero utility on average. Scoring negatively on utility means that the user would prefer to use no system at all. We found ourselves deliberately worsening our filtering system (that is lowering its precision and recall) to improve the utility score up to a point where we came pretty close to no system at all. The uttno8lf1p run did not select any document for 22 out of 50 topics.

Average utility and average precision/recall both have their disadvantages if used for the evaluation of adaptive filtering runs. In short, precision causes problems with topics for which the system selected no relevant document at all, and the problem with average utility is that it will be dominated by topics with large retrieved sets [11]. We feel that utility and precision/recall are both valuable measures for the evaluation of adaptive filtering systems. In future evaluations, situations in which both measures contradict each other, like for the LF1 experiments mentioned above, should be avoided. An obvious solution would be to aim a little bit lower. The LF2 utility function seems to be a reasonable measure for future evaluations.

5 Spoken Document Retrieval

5.1 Word recognition vs. word spotting

In TREC-7, TNO [5] investigated whether effective retrieval algorithms based on phoneme recognition and a word spotter could be built. The absence of a Language Model (which is a key component in a word based recognizer) was found to be a serious drawback. For TREC-8, LIMSI kindly provided the word recognition transcripts. For details on the speech recognition algorithms we refer to LIMSI's paper in this volume.

5.2 Olive

TNO-TPD , University of Twente and LIMSI participate in the EU project Olive. This project is a direct descendant of the Twenty-One project. Olive uses Twenty-One retrieval technology to retrieve video fragments from a video database. In order to enable an automatic indexing step of the video material, we employ automatic speech recognition on the soundtrack of the video. The recognition transcripts contain detailed timecode information which ensures a precise coupling of the transcripts with the video. For video retrieval, a user must type a query, the query is matched against an index of noun phrases extracted from the recognition transcripts. The resulting hitlist is visualized by marking hits on a bar which represents the timeline of a video. Clicking on one of these marks will start the video at the corresponding offset through a streaming server. The video material that is used in Olive is in German and French. The speech recognition for these languages is developed at LIMSI. Because the TREC SDR task is higly relevant for Olive, we decided to cooperate with LIMSI for the TREC-8 task. LIMSI provided us with transcripts of both the TREC-7 and TREC-8 SDR data, and we tuned our retrieval on the TREC-7 SDR test collection.

5.3 Relevance Feedback

We studied pseudo relevance feedback techniques that were successfully applied by other groups in TREC-7. After some testing on TREC-7 we found that a technique introduced as "Blind Relevance Feedback" [15] performed best. The relevance feedback was applied on a larger secondary corpus: the TREC Ad Hoc corpus. Even though the corpus covers a different time-span, results with the secondary corpus were better than BRF on the SDR corpus. The following BRF parameters were used:

- select top 20 documents
- compute 60 best terms based on BRF algorithm
- add new terms down-weighted with factor 20/6

5.4 Unknown story boundaries

We reviewed the literature on story segmentation but because of time pressure we were only able to implement a baseline system for unknown story boundaries, based on fixed windows. So we did not attempt to detect story boundaries at all, we simply wanted to know how a baseline system would perform. In [18] a default section window size of 250 words was recommended, this was estimated as a 15kbyte length fragment of the transcript files, because the average number of bytes per recognized word (including timecode mark-up) is about 60. The segments have an overlap of 50 bytes to avoid missing words that occur right at a window boundary.

5.5 Experimental setup and results

We tested two term weighting algorithms (LM and BM25) in combination with two automatic query expansion techniques (PRF and BRF) on the TREC-7 SDR test collection. A combination of BM25 and Blind relevance feedback (as implemented by Cambridge University[15]) yielded the best results. For TREC8 we found that LM weighting performed consistently better than BM25 in combination with BRF (see table 9). The somewhat poorer performance of LM on TREC-7 SDR can probably attributed to the rather small size of the TREC-7 collection, the TREC-8 results are probably much more reliable.

The results[5] for the known story boundary conditions are good, though they could be improved. It was only after the submission deadline that we discovered that quite a few topics contain proper noun abbreviations in a format which is idiosyncratic for recognizer output, e.g, *U. S.* which would normally be

[5] We list the uninterpolated average precision over 49 topics

run-name	transcript source	term weighting	mode	AVP
tno8b-r1-limsi	manual	BM25	BRF	0.4806
tno8b-b1-limsi	NIST	BM25	BRF	0.4650
tno8b-s1-limsi	LIMSI	BM25	BRF	0.4826
tno8c-r1-limsi	manual	LM	BRF	0.5169
tno8c-b1-limsi	NIST	LM	BRF	0.4898
tno8c-s1-limsi	LIMSI	LM	BRF	0.4969

Table 9: SDR results: Known Story Boundaries

spelled as *US*. Our tokenizer will remove the abbreviation dots, and the single letters will be stopped as well. What we need is a special tokenizer which recognizes these special cases.

run-name	transcript source	mode	AVP
tno8b-b1u-limsi	NIST	BRF	0.0238
tno8b-s1u-limsi	LIMSI	BRF	0.0325

Table 10: SDR results: Unknown Story Boundaries

The unknown story boundary condition yielded very poor results. This is probably due to the fact that no effort was done to merge clusters of hits into single documents. Multiple hits in the same story were quite heavily penalized in the scoring algorithm. Further analysis is needed to check this assumption.

6 Conclusions

The probabilistic retrieval model based on statistical language models performs consistently well in all tracks. The results of the experiments with sense disambiguation are slightly disappointing, although a real evaluation is only possible when the techniques are more mature. It is a question however whether the disambiguation step can be effective because documents are indexed on terms, not on senses. We improved upon our CLIR results of last year, due to a better merging technique, unfortunately our best official xlingual run (tno8gr) suffered from a merging error. Our best xlingual run uses the corpus for implicit disambiguation and interpolates between a rank and score based merging strategy. In Adaptive Filtering we showed that LM weighting can be extended with a relevance feedback algorithm. Finally, in the SDR track we showed that a word error rate of 26.3% does not harm retrieval effectiveness in a significant way when standard retrieval techniques are used.

Acknowledgements

The work reported in this paper is funded in part by the Dutch Telematics Institute project Druid and the EU project Olive (LE 8364). We would like to thank Chris Buckley for suggesting the evaluation of the cross-language pool. Furthermore we would like to thank Rudie Ekkelenkamp and Hap Kolb of TNO-TPD for their help with the SDR and filtering tracks, and LIMSI for providing their SR transcripts to us.

Appendix: using language models for document ranking

The Twenty-One TREC-8 evaluations are based on the use of statistical language models for information retrieval [8, 9, 10]. This appendix gives an overview of the model and of its application to cross-language information retrieval and adaptive filtering. Similar models were developed and evaluated by other groups participating in TREC [3, 12, 14, 17].

A.1 An informal description of the underlying ideas

When using statistical language models for information retrieval, one builds a simple language model for each document in the collection. The term "language model" refers to statistical models similar to language models used in e.g. speech recognition. Given a query, a document is assigned the probability that the language model of that document generated the query.

The metaphor of "urn models" [13] might give more insight. Instead of drawing balls at random with replacement from an urn, we will consider the process of drawing words at random with replacement from a document. Suppose someone selects one document in the document collection; draws at random, one at a time, with replacement, ten words from this document and hands those ten words (the query terms) over to the system. The system now can make an educated guess as from which document the words came from, by calculating for each document the probability that the ten words were sampled from it and by ranking the documents accordingly. The intuition behind it is that users have a reasonable idea of which terms are likely to occur in documents of interest and will choose query terms accordingly [17]. In practice, some query terms do not occur in any of the relevant documents. This can be modeled by a slightly more complicated urn model. In this case the person who draws at random the ten words, first decides for each draw if he will draw randomly from a relevant document or randomly from the entire collection. The yes/no decision of drawing from a relevant document or not, will also be assigned a probability. This probability will be called the relevance weight of a term, because it defines the distribution of the term over relevant and non-relevant documents. For ad-hoc retrieval all non stop words in the query will be assigned the same relevance weight. For adaptive filtering, the user's feedback will be used to re-estimate the relevance weight for each query term.

The model evaluates Boolean queries by treating the sampling process as an AND-query and allowing that each draw is specified by a disjunction of more than one term. For example, the probability of first drawing the term *information* **and** then drawing either the term *retrieval* **or** the term *filtering* from a document can be calculated by the model introduced in this paper without any additional modeling assumptions. Boolean queries were used to model more than one possible translation per query term in cross-language information retrieval.

Furthermore, the model can be extended with additional statistical processes to model differences between the vocabulary of the query and the vocabulary of the documents. Statistical translation can be added to the process of sampling terms from a document by assuming that the translation of a term does not depend on the document it was sampled from. Cross-language retrieval using e.g. English queries on a French document collection uses the sampling metaphor as follows: first an French word is sampled from the document, and then this word is translated to English with some probability that can be estimated from a parallel corpus.

A.2 Definition of the corresponding probability measures

Based on the ideas mentioned above, probability measures can be defined to rank the documents given a query. The probability that a query $T_1, T_2, \cdots, T_n$ of length n is sampled from a document with document identifier D is defined by equation 2.

$$P(T_1, T_2, \cdots, T_n|D) = \prod_{i=1}^{n}((1-\lambda_i)P(T_i) + \lambda_i P(T_i|D)) \tag{2}$$

In the formula, $P(T)$ is the probability of drawing a term randomly from the collection, $P(T|D)$ is the probability of drawing a term randomly from a document and λ_i is the relevance weight of the term. If a query term is assigned a relevance weight of $\lambda_i = 1$, then the term is treated as in exact matching: the system will assign zero probability to documents in which the term does *not* occur. If a query term is assigned a relevance weight of 0, then the term is treated like a stop word: the term does not have any influence on the final ranking. In section A.4 it is shown that this probability measure can be rewritten to a tf×idf term weighting algorithm. A similar probability function was used by Miller, Leek and Schwartz [12]. They showed that it can be interpreted as a two-state hidden Markov model in which λ and $(1-\lambda)$ define the state transition probabilities and $P(T)$ and $P(T|D)$ define the emission probabilities.

The evaluation of Boolean queries for cross-language retrieval is straightforward. For each draw, different terms are mutually exclusive. That is, if one term is drawn from a document, the probability of drawing e.g. both the term *information* and the term *retrieval* is 0. Following the axioms of probability theory (see e.g. Mood [13]) the probability of a disjunction of terms in one draw is the sum of the probabilities of drawing the single terms. Disjunction of m possible translations T_{ij} $(1 \leq j \leq m)$ of the source language query term on position i is defined as follows.

$$P(T_{i1} \cup T_{i2} \cup \cdots \cup T_{im}|D) = \sum_{j=1}^{m}((1-\lambda_i)P(T_{ij}) + \lambda_i P(T_{ij}|D)) \tag{3}$$

Following this line of reasoning, AND queries are interpreted similar as unstructured queries defined by equation 2. Or, to put it differently, unstructured queries are implicitly assumed to be AND queries. If a relevance weight of $\lambda_i = 1$ is assigned to each query term, then the system will behave like the traditional Boolean model of IR. Statistical translation is added to these probability measures by assuming that the translation of a term does not depend on the document it was drawn from [9]. If $N_1, N_2, \cdots, N_n$ is a English query of length n and a English term on position i has m_i possible French translations T_{ij} $(1 \leq j \leq m_i)$, then the ranking as structured queries would be done according to equation 4

$$P(N_1, N_2, \cdots, N_n|D) = \prod_{i=1}^{n}\sum_{j=1}^{m_i} P(N_i|T_{ij})((1-\lambda_i)P(T_{ij}) + \lambda_i P(T_{ij}|D)) \tag{4}$$

The translation probabilities $P(N_i|T_{ij})$ can be estimated from parallel corpora, or alternatively by using occurrences in a machine readable dictionary . A very similar model that also combines document ranking and statistical translation was introduced by Berger and Lafferty [2, 3]. Their model differs from equation 4 only by a different smoothing method, using global information on N_i instead of global information on each T_{ij}.

A.3 Parameter estimation

In information retrieval it is good practice to use the term frequency and document frequency as the main components of term weighting algorithms. Our probabilistic model does not make an exception. The term frequency $tf(t,d)$ is defined by the number of times the term t occurs in the document d. The document frequency $df(t)$ is defined by the number of documents in which the term t occurs. Estimation of $P(T)$ and $P(T|D)$ in equation 2, 3 and 4 was done as follows:

$$P(T_i = t_i) = \frac{df(t_i)}{\sum_t df(t)} \tag{5}$$

$$P(T_i = t_i|D = d) = \frac{tf(t_i, d)}{\sum_t tf(t, d)} \tag{6}$$

The value of the relevance weights λ_i might change for different applications. High relevance weights result in tf×idf rankings that obey the conditions of coordination level ranking [10], that is, documents containing n query terms are always ranked above documents containing $n-1$ query terms. High relevance weights are a good choice for applications that aim at high precision or applications in which very short queries are used, like web search engines. Documents that are judged as relevant by the user can be used to re-estimate the relevance weights. An algorithm for relevance weighting was developed for the adaptive filtering task (see section 4).

A.4 Some notes on the implementation

Similar to traditional probabilistic models of information retrieval [19] probability measures for ranking documents can be rewritten into a format that is easy to implement. A presence weighting scheme (as opposed to a presence/absence weighting scheme) assigns a zero weight to terms that are not present in a document. Presence weighting schemes can be implemented using the vector product formula. This section presents the resulting algorithms. Rewriting equation 2 results in the formula displayed in figure 2 [10]. It can be interpreted as a tf×idf weighting algorithm with document length normalisation as defined by Salton and Buckley [20].

vector product formula:	$\text{score}(d,q) \;=\; \sum_{\substack{k\,\in\,\text{match-}\\ \text{ing terms}}} w_{qk} \cdot w_{dk}$
query term weight:	$w_{qk} = tf(k,q)$
document term weight:	$w_{dk} = \log(1 + \frac{tf(k,d)\sum_t df(t)}{df(k)\sum_t tf(t,d)} \cdot \frac{\lambda_k}{1-\lambda_k})$

Figure 2: tf×idf term weighting algorithm

If a structured query is used, the disjunction of possible translations as defined by equation 3 should be calculated first. As addition is associative and commutative, we do not have to calculate each linear interpolation of equation 3 separately before summing them. Instead, the document frequencies and the term frequencies of the disjuncts respectively, can be added beforehand. The added frequencies can be used to replace $df(k)$ and $tf(k,d)$ in the weighting formula of table 2. A similar ranking algorithm for Boolean queries was introduced earlier by Harman [7] for on-line stemming. Harman did not present her algorithm as an extension of Boolean searching, but instead called it 'grouping'. Instead of adding the document frequencies, the TNO vector engine calculates the actual document frequencies of the disjuncts, by merging their postings at run time. A similar approach for cross-language information retrieval was adopted by Ballesteros and Croft [1] by using a 'synonym operator' on possible translations.

If translation probabilities are used following equation 4, the adding of respectively the document frequencies and the term frequencies of the disjuncts should be done as a weighted sum with the translation probabilities as weights.

References

[1] L. Ballesteros and W.B. Croft. Resolving ambiguity for cross-language retrieval. In *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 64–71, 1998.

[2] A. Berger and J. Lafferty. The Weaver system for document retrieval. In *In* Proceedings of the 22nd ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR'99, pages 222–229, 1999.

[3] A. Berger and J. Lafferty. The Weaver system for document retrieval. In *Proceedings of the seventh Text Retrieval Conference TREC-8*, 2000. elsewhere in this volume.

[4] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em-algorithm plus discussions on the paper. *Journal of the Royal Statistical Society*, 39(B):1–38, 1977.

[5] R. Ekkelenkamp, W. Kraaij, and D. van Leeuwen. TNO TREC-7 site report: SDR and Filtering. In *Proceedings of the seventh Text Retrieval Conference TREC-7*, pages 519–526, 1999. NIST Special Publication 500-242.

[6] M. Franz, J.s. McCarley, and S. Roukos. Ad hoc and multilingual information retrieval at IBM. In *Proceedings of the seventh Text Retrieval Conference TREC-7*, pages 157–168, 1999. NIST Special Publication 500-242.

[7] D. Harman. How effective is suffixing? *Journal of the American Society for Information Science*, 42(1):7–15, 1991.

[8] D. Hiemstra. A probabilistic justification for using tf.idf term weighting in information retrieval. *International Journal on Digital Libraries*, 1999. (to appear).

[9] D. Hiemstra and F.M.G. de Jong. Disambiguation strategies for cross-language information retrieval. In *Proceedings of the third European Conference on Research and Advanced Technology for Digital Libraries*, pages 274–293, 1999.

[10] D. Hiemstra and W. Kraaij. Twenty-One at TREC-7: Ad-hoc and cross-language track. In *Proceedings of the seventh Text Retrieval Conference TREC-7*, pages 227–238, 1999. NIST Special Publication 500-242.

[11] D. Hull. The TREC-7 filter track: Description and analysis. In *Proceedings of the seventh Text Retrieval Conference TREC-7*, pages 33–36, 1999. NIST Special Publication 500-242.

[12] D.R.H. Miller, T. Leek, , and R.M. Schwartz. BBN at TREC-7: using hidden markov models for information retrieval. In *Proceedings of the seventh Text Retrieval Conference, TREC-7*, 1999. NIST Special Publication 500-242.

[13] Mood and F.A. Graybill, editors. *Introduction to the Theory of Statistics, Second edition*. McGraw-Hill, 1963.

[14] K. Ng. A maximum likelihood ratio information retrieval model. In *Proceedings of the seventh Text Retrieval Conference TREC-8*, 2000. elsewhere in this volume.

[15] Karen Sparck Jones Pierre Jourlin, Sue E. Johnson and Philip C. Woodland. General query expansion techniques for spoken document retrieval. In *Proceedings of the ESCA ETRW Workshop: Accessing Information in Spoken Audio*, 1999.

[16] Renée Pohlmann and Wessel Kraaij. The effect of syntactic phrase indexing on retrieval performance for Dutch texts. In L. Devroye and C. Chrisment, editors, *Proceedings of RIAO'97*, pages 176–187, 1997.

[17] J.M. Ponte and W.B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, 1998.

[18] Jeffrey C. Reynar. *Topic Segmentation: Algorithms and Applications*. PhD thesis, University of Pennsylvania, 1998.

[19] S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

[20] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In *Information Processing & Management*, volume 24, pages 513–523, 1988.

[21] C. Zhai, P. Jansen, E. Stoica, N. Grot, and D.A. Evans. Threshold calibration in CLARIT adaptive filtering. In *Proceedings of the seventh Text Retrieval Conference TREC-7*, pages 149–156, 1999. NIST Special Publication 500-242.