Twenty-One at CLEF-2000: Translation resources, merging strategies and relevance feedback

[†]Djoerd Hiemstra, *Wessel Kraaij, *Renée Pohlmann and [†]Thijs Westerveld

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

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

Abstract

This paper describes the official runs of the Twenty-One group for CLEF-2000. The Twenty-One group participated in the monolingual, bilingual and multilingual tasks. The following new techniques are introduced in this paper. In the bilingual task we experimented with different methods to estimate translation probabilities. In the multilingual task we experimented with refinements on raw-score merging techniques and with a new relevance feedback algorithm that re-estimates both the model's translation probabilities and the relevance weights. Finally, we performed preliminary experiments to exploit the web to generate translation probabilities and bilingual dictionaries, notably for English-Italian and English-Dutch.

1 Introduction

Twenty-One is a project funded by the EU Telematics Applications 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 cross-language information retrieval (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 / 7 / 8. Since the cooperation is continued in other projects: Olive and Druid, we have decided to continue our participation in CLEF as "Twenty-One". [1] For all tasks, we used the TNO vector retrieval engine. The engine supports several term weighting schemes. The principal term weighting scheme we used is the "linguistically motivated probabilistic model of information retrieval" [2, 4] explained below.

2 The approach

All runs were carried out with an information retrieval system based on a simple unigram language model. The basic idea is that documents can be represented by simple statistical language models. Now, if a query is more probable given a language model based on document d_1, than given e.g. a language model based on document d_2, then we hypothesise that the document d_1 is more relevant to the query than document d_2. Thus the probability of generating a certain query given a document-based language model can serve as a score to rank documents with respect to relevance.

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

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

Formula 1 shows the basic idea of this approach to information retrieval, where the document-based language model is interpolated with a background language model to compensate for sparseness. In the formula, T_i is a random variable for the query term on position i in the query ($1 \leq i \leq n$, where n is the query length), which sample space is the set $\{t^{(0)}, t^{(1)}, \cdots, t^{(m)}\}$ of all terms in the collection. The probability measure $P(T_i)$ defines the probability of drawing a term at random from the collection, $P(T_i|D_k)$ defines the probability of drawing a term at random from the document; and λ_i defines the importance of each query term. The marginal probability of relevance $P(D_k)$ might be assumed uniformly distributed over the documents in which case it may be ignored in the above formula.

2.1 A model of cross-language information retrieval

Information retrieval models and statistical translation models can be integrated into one unifying model for cross-language information retrieval [1, 4]. Let S_i be a random variable for the source language query term on position i. Each document gets a score defined by the following formula.

$$\begin{aligned} P(S_1, S_2, \cdots, S_n|D_k)\, P(D_k) \; = \\ P(D_k) \prod_{i=1}^{n} \sum_{j=1}^{m} P(S_i|T_i\!=\!t^{(j)})((1\!-\!\lambda_i) P(T_i\!=\!t^{(j)}) + \lambda_i P(T_i\!=\!t^{(j)}|D_k)) \end{aligned} \tag{2}$$

In the formula, the probability measure $P(S_i|T_i\!=\!t^{(j)})$ defines the translation probabilities.

2.2 Translation in practice

In practice, the statistical translation model will be used as follows. The automatic query formulation process will translate the query $S_1, S_2, \cdots, S_n$ using a probabilistic dictionary. The probabilistic dictionary is a dictionary that list pairs (s, t) together with their probability of occurrence, where s is from the sample space of S_i and t is from the sample space of T_i. For each S_i there will be one or more realisations t_i of T_i for which $P(S_i|T_i = t_i) > 0$, which will be called the possible translations of S_i. The possible translations should be grouped for each i to search the document collection, resulting in a structured query.

For instance, suppose the original French query on an English collection is “déchets dangereux”, then possible translations of “déchets” might be “waste”, “litter” or “garbage”, possible translations of “dangereux” might be “dangerous” or “hazardous” and the structured query can be presented as follows.

$$((\texttt{waste} \cup \texttt{litter} \cup \texttt{garbage}),\ (\texttt{dangerous} \cup \texttt{hazardous}))$$

The product from $i = 1$ to n (in this case $n = 2$) of equation 2 is represented above by using the comma as is done in the representation of a query of length 2 as T_1, T_2. The sum from $j = 1$ to m of equation 2 is represented by displaying only the realisations of T_i for which $P(S_i|T_i) > 0$ and by separating those by ‘$\cup$’. So, in practice, translation takes place during automatic query formulation (query translation), resulting in a structured query like the one displayed above that is matched against each document in the collection. Unless stated otherwise, whenever this paper mentions ‘query terms’, it will denote the target language query terms: realisations of T_i. Realisations of S_i, the source language query terms, will usually left implicit. The combination of the structured query representation and the translation probabilities will implicitly define the sequence of the source language query terms $S_1, S_2, \cdots, S_n$, but the actual realisation of the sequence is not important to the system.

2.3 Probability estimation

The prior probability of relevance $P(D_k)$, the probability of term occurrence in the collection $P(T_i)$ and the probability of term occurrence in the relevant document $P(T_i|D_k)$ are defined by the collection that is searched. For the evaluations reported in this paper, the following definitions were used, where $tf(t, k)$ denotes the number of occurrences of the term t in the document k, and $df(t)$ denotes the number of documents in which the term t occurs. Equation 3 is the definition used for the unofficial “document length normalisation" runs reported in section 5.

$$P(D_k) = \frac{\sum_t tf(t,k)}{\sum_{t,d} tf(t,d)} \tag{3}$$

$$P(T_i = t_i | D_k) = \frac{tf(t_i,k)}{\sum_t tf(t,k)} \tag{4}$$

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

The translation probabilities $P(S_i|T_i)$ and the value of λ_i, however, are unknown. The collection that is searched was not translated, or if it was translated, the translations are not available. Translation probabilities should therefore be estimated from other data, for instance from a parallel corpus. The value of λ_i determines the importance of the source language query term. If $\lambda_i = 1$ then the system will assign zero probability to documents that do not contain any of the possible translations of the original query term on position i. In this case, a possible translation of the source language term is mandatory in the retrieved documents. If $\lambda_i = 0$ then the possible translations of the original query term on position i will not affect the final ranking. In this case, the source language query term is treated as if it were a stop word. For ad-hoc queries, it is not known which of the original query terms are important and which are not important and a constant value for each λ_i is taken. The system's default value is $\lambda_i = 0.3$.

2.4 Implementation

Equation 2 is not implemented as is, but instead it is rewritten into a weighting algorithm that assigns zero weight to terms that do not occur in the document. Filling in the definitions of equation 3, 4 and 5 in equation 2 results in the following formula. The probability measure $P(S_i|T_i = t^{(j)})$ will be replaced by the translation probability estimates $\tau_i(j)$.

$$P(D_k, S_1, S_2, \cdots, S_n) = \frac{\sum_t tf(t,k)}{\sum_{t,d} tf(t,d)} \prod_{i=1}^{n} \sum_{j=1}^{m} \tau_i(j)((1-\lambda_i)\frac{df(t^{(j)})}{\sum_t df(t)} + \lambda_i \frac{tf(t^{(j)},k)}{\sum_t tf(t,k)})$$

The translation probabilities can be moved into the inner sum. As summing is associative and commutative, it is not necessary to calculate each probability separately before adding them. Instead, respectively the document frequencies and the term frequencies of the disjuncts can be added beforehand, properly multiplied by the translation probabilities. Only λ_i in the big sum is constant for every addition and can therefore be moved outside the sum, resulting in:

$$P(D_k, S_1, S_2, \cdots, S_n) = \frac{\sum_t tf(t,k)}{\sum_{t,d} tf(t,d)} \prod_{i=1}^{n} ((1-\lambda_i)\frac{\sum_{j=1}^{m} \tau_i(j) df(t^{(j)})}{\sum_t df(t)} + \lambda_i \frac{\sum_{j=1}^{m} \tau_i(j) tf(t^{(j)},k)}{\sum_t tf(t,k)})$$

Using simple calculus (see e.g. [3]), the probability measure can now be rewritten into a term weighting algorithm that assigns zero weight to non-matching terms, resulting in equation 6. The formula ranks documents in exactly the same order as equation 2.

$$P(D_k, S_1, S_2, \cdots, S_n) \propto \log(\textstyle\sum_t tf(t,k)) + \displaystyle\sum_{i=1}^{n} \log(1 + \frac{\lambda_i \, (\sum_{j=1}^{m} \tau_i(j) tf(t^{(j)},k)) \sum_t df(t)}{(1-\lambda_i)(\sum_{j=1}^{m} \tau_i(j) df(t^{(j)})) \sum_t tf(t,k)}) \tag{6}$$

Equation 6 is the algorithm implemented in the TNO retrieval engine. It contains a weighted sum of respectively the term frequencies and the document frequencies where the weights are determined by the translation probabilities $\tau_i(j)$. Unweighted summing of frequencies was used before for on-line stemming in [5] in a vector space model retrieval system.

The model does not require the translation probabilities $\tau_i(j)$ to sum up to one for each i, since they are conditioned on the target language query term and not on the source language query term. Interestingly, for the final ranking it does not matter what the actual sum of the translation probabilities is. Only the relative proportions of the translations define the final ranking of documents. This can be seen by $\tau_i(j)$ which occurs in the numerator and in the denominator of the big fraction in equation 6.

2.5 A Relevance feedback method for cross-language retrieval

This paper introduces a new relevance feedback method for cross-language information retrieval. If there were some known relevant documents, then the values of $\tau_i(j)$ and λ_i could be re-estimated from that data. The idea is the following. Suppose there are three known relevant English documents to the French query “déchets dangereux”. If two out of three documents contain the term “waste” and none contain the terms “litter” and “garbage” then this is an indication that “waste” is the correct translation and should be assigned a higher translation probability than ‘litter” and “garbage”. If only one of the three known relevant document contains one or more possible translations of “dangereux” then this is an indication that the original query term “déchets” is more important (possible translations occur in more relevant documents) than the original query term “dangereux” and the value of λ_i should be higher for “déchets” than for “dangereux”.

The actual re-estimation of $\tau_i(j)$ and λ_i was done by iteratively applying the EM-algorithm defined by the formulas in equation 7. In the algorithm, $\tau_i(j)^{(p)}$ and $\lambda_i^{(p)}$ denote the values on the pth iteration. The values are initialised with the translation probabilities from the dictionary and with $\lambda_i^{(0)} = 0.3$. The re-estimation formulas should be used simultaneously for each p until the values do not change significantly anymore.

$$\begin{aligned}
\tau_i(j)^{(p+1)} &= \frac{1}{r}\sum_{k=1}^{r} \frac{\tau_i(j)^{(p)}\,((1-\lambda_i^{(p)})P(T_i=t^{(j)}) + \lambda_i^{(p)}P(T_i=t^{(j)}|D_k))}{\sum_{l=1}^{m}\tau_i(l)^{(p)}((1-\lambda_i^{(p)})P(T_i=t^{(l)}) + \lambda_i^{(p)}P(T_i=t^{(l)}|D_k))} \\
\lambda_i^{(p+1)} &= \frac{1}{r}\sum_{k=1}^{r} \frac{\lambda_i^{(p)}\,(\sum_{l=1}^{m}\tau_i(l)^{(p)}\,P(T_i=t^{(l)}|D_k))}{\sum_{l=1}^{m}\tau_i(l)^{(p)}((1-\lambda_i^{(p)})P(T_i=t^{(l)}) + \lambda_i^{(p)}P(T_i=t^{(l)}|D_k))}
\end{aligned} \tag{7}$$

The re-estimation of $\tau_i(j)$ and λ_i was done from ‘pseudo-relevant’ documents. First the top 10 documents were retrieved using the default values of $\tau_i(j)$ and λ_i and then the feedback algorithm was used on these documents to find the new values. The actual algorithm implemented was a variation of equation 7 of the form: $(1\,/\,(r+1))\cdot$ (default value $+\sum_{k=1}^{r}\ldots$) to avoid that e.g. $\lambda_i = 1$ after re-estimation.

3 Translation resources

As in previous years we applied a dictionary based query translation approach. The translations were based on the VLIS lexical database of Van Dale publishers [2]. Because VLIS currently lacks translations into Italian, we used two other resources: i) the Systran web based MT engine ii) a probabilistic lexicon based a parallel web corpus. The next section will describe the construction of this new resource in more detail.

3.1 Parallel web corpora

We developed three parallel corpora based on web pages in close cooperation with RALI, Université de Montréal. RALI already had developed an English-French parallel corpus of web pages, so it seemed interesting to investigate the feasibility of a full multilingual system based on web derived lexical resources only. We used the PTMiner tool [7] to find web pages which have a high probability to be translations of each other. The mining process consists of the following steps:

1. Query a web search engine for web pages with a hyperlink anchor text “English version” and respective variants.

2. (For each web site) Query a web search engine for all web pages on a particular site.

3. (For each web site) Try to find pairs of path names that match certain patterns, e.g.: `/department/tt/english/home.html` and `/department/tt/italian.html`.

4. (For each pair) download web pages, perform a language check using a probabilistic language classifier, remove pages which are not positively identified as being written in a particular language.

The mining process was run for three language pairs and resulted in three modest size parallel corpora. Table 1 lists sizes of the corpus during intermediate steps. Due to the dynamic nature of the web, a lot of pages that have been indexed, do not exist anymore. Sometimes a site is down for maintenance. Finally a lot of pages are simply place holders for images and are discarded by the language identification step.

language	nr of web sites	nr of candidate pages	nr of candidate pairs	retrieved + cleaned pairs
EN-IT	3651	1053649	23447	4768
EN-DE	3817	1828906	33577	5743
EN-NL	3004	1170082	24738	2907

Table 1: Intermediate sizes during corpus construction

These parallel corpora have been used in different ways: i) to refine the estimates of translation probabilities of a dictionary based translation system (corpus based probability estimation) ii) to construct simple statistical translation models (IBM model 1) [7] . The former application will be described in more detail in Section 5.2 the latter in Section 5.3. The translation models for English-Italian and English-German, complemented with an already existing model for English-French formed also the basis for a full corpus based translation multilingual run which is described elsewhere in this volume [6].

4 Merging intermediate runs

Our strategy to multilingual retrieval is to translate the query into the document languages, perform separate language specific runs and merge the results into a single result file. In previous CLIR evaluations, we compared different merging strategies:

round robin Here the idea is that document scores are not comparable across collections, because we are basically ignorant about the distribution of relevant documents in the retrieved lists, round robin assumes that these distributions are similar across languages.

raw score This type of merging assumes that document scores are comparable across collections.

rank based It has been observed that the relationship between probability of relevance and the log of the rank of a document can be approximated by a linear function, at least for a certain class of IR systems. If a training collection is available, one can estimate the parameters of this relationship by applying regression. Merging can subsequently be based on the estimated probability of relevance. Note that the actual score of a document is only used to rank documents, but that merging is based on the rank, not on the score.

The new CLEF multilingual task is based on a new document collection which makes it hard to compute reliable estimates for the linear parameters; a training set is not available. A second disadvantage of the rank based merging strategy is that the linear function generalises across topics. Unfortunately in the multilingual task, the distribution of relevant documents over the subcollections is quite skewed. All collections have several (differing) topics without relevant documents, so applying a rank based merging strategy would hurt the performance for these topics, because the proportion of retrieved documents in every collection is the same for every topic.

The raw score merging strategy (which proved succesful last year) does not need training data and also does not suffer from the equal proportions strategy. Unfortunately, usually scores are not totally compatible across collections. We have tried to identify factors which cause these differences. We have applied two normalization techniques. First of all we treat term translations as a weighted concept vector (cf. section 2). That means that we can normalise scores across topics by dividing the score by the query length. This amounts to computing the geometric avarage of probabilities per query concept. Secondly, we have observed that collection size has a large influence on the occurence probability estimates $P(T_i|C)$ because the probability of rare terms is inversely proportional to the collection size.

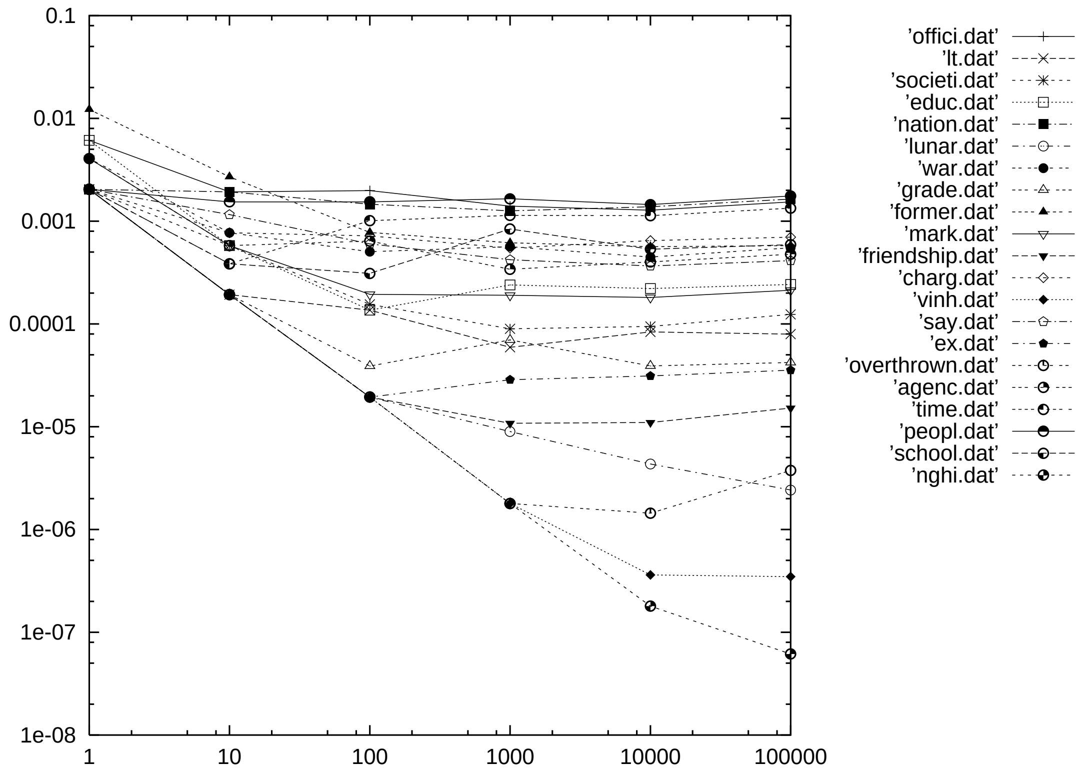

Figure 1: Probability estimates vs collection size

Figure 4 shows the probability estimates of a sample of words of 1 document when we add more documents to the collection. The occurrence probability of common words stabilises fast when the collection size increases. The more rare a word is however, the higher is the degree of overestimation of its occurrence probability. This effect is a consequence of the sparse data problem. In fact, a small collection will never yield correct term occurrence probability estimates.

The collection-size dependency of collection-frequency (or global term frequency) estimates has a direct influence on the distribution of document scores for a particular query. When the collection is small, the scores will be lower than the scores on a large collection. This is due to the fact that the score we study is based on the maximum likelihood ratio. So the median of the distribution of document scores for a particular topic (set) is inversely related with the collection size. Thus when we use the raw scores of different subcollections as a basis for merging, large collections will be favoured.

We hypothesised that we could improve the merging process, if we could correct the estimates for their dependence on the collection size. Suppose we have just two collections with a different size (and different language): C_1,C_2 with vocabulary size V_1,V_2 and number of tokens T_1, T_2 respectively, with $T_1 << T_2$. Now we could try to either extrapolate the term occurrence probability estimates on collection C_1 to a hypothetical collection with T_2 tokens or try to 'downscale' the term occurrence probability estimates of a term from C_2 to vocabulary size V_1.

The first option seems cumbersome, because we have hardly information to guide the extrapolation process. The second option, trying to adapt the estimates of the large collection to the small collection, seems more viable. The idea is to adapt the probability estimates of rare terms in such a way, that they will become ‘compatible’ with the estimates on the small collection. As shown in figure 4 the estimates of frequent terms stabilise soon. Our idea is to construct a mapping function which maps the probability estimates to the small collection domain. The mapping function has the following requirements: a probability $1/T_2$ has to be mapped to $1/T_1$. So the probability is multiplied by the factor T_2/T_1 and probabilities p larger than $1/T_2$ will be multiplied by a factor which decreases for larger p. In fact we only want very small changes for $p > 10^{-3}$. A function which meets these properties is the polynomial $f(x) = x - ax^2$ (where $x = log(p)$ and $a = \frac{T_2 - T_1}{T_2^2}$). Because we have re-estimated the probabilities, one would expect that the probabilities have to be re-normalised ($p\prime(t_i) = p(t_i) / \sum^{V_2} p(t_i)$). However, this has the result that all global probabilities (also those of relatively frequent words) are increased, which will increase the score of all documents, i.e. will have the opposite effect of what we want. So we decide not to re-normalise, because a smaller corpus would also have a smaller vocabulary, which would compensate for the increase in probability mass which is a result of the transformation.

5 Results

5.1 Monolingual runs

We indexed the collections in the 4 languages separately. All documents were lemmatised using the Xelda morphological toolkit from Xerox XRCE and stopped with language specific stoplists. For German, we splitted compounds and added both the full compound and its parts to the index. This strategy is motivated by our experience with a Dutch corpus (Dutch is also a compounding language) [8] and tests on the TREC CLIR test collection. Table 2 shows the results of the monolingual runs, runs in bold are judged runs, runs in italic font are unofficial runs (mostly post-hoc). The table also lists the proportion of documents which has been judged. The standard runs include fuzzy lookup of unknown words. The expand option adds close orthographical variants for every query term. The official runs were done without document length normalisation defined by equation 3.

run name	avp	above median	description	% j@1000	%j@100	%j@10
tnoutdd1	0.3760	-	standard	18.64	79.05	100
tnoutdd2	0.3961	28/37	+expand	18.72	81.22	100
tnoutdd2l	0.3968	-	+length normalisation	18.58	78.22	97.50
tnoutff1	0.4551	-	standard	16.13	79.42	100
tnoutff2	0.4471	18/34	+expand	16.21	80.88	100
tnoutff2l	0.4529	-	+length normalisation	16.00	77.88	97.50
tnoutii1	0.4677	-	standard	16.59	78.92	100
tnoutii2	0.4709	18/34	+expand	16.67	80.33	100
tnoutii2l	0.4808	-	+length normalisation	16.66	77.25	98
tnoutee01i	0.4200	-	standard	17.81	71.10	100
tnoutee01	0.4169	-	+expand	17.84	70.75	99.75
tnoutee01l	0.4273	-	+length normalisation	17.82	69.30	98.00

Table 2: Results of the monolingual runs

The first thing that strikes us, is that the pool depth is 50, contrary to what has been practice in TREC in which the top 100 documents are judged for relevance. Section 5.4 analyses the CLEF collection further. Length normalisation usually gives a modest improvement in average precision. The ‘expand’ option was especially effective for German. The reason is probably that compound parts are not always properly lemmatised by the German morphology. Especially the German run performs well with 28 out of 37 topics above average. This relatively good performance is probably due to the morphology, which includes compound splitting.

5.2 Bilingual runs

Table 3 lists the results of the bilingual runs. All runs use Dutch as a query language. The base run of 0.3069 can be improved by several techniques: a higher lambda, document length normalisation or Porter stemming instead of dictionary based stemming. The latter can be explained by the fact that Porter's algorithm is an aggressive stemmer that also removes most of the derivational affixes. This is usually beneficial to retrieval performance. The experiment with corpus based frequencies yielded disappointing results. We first generated topic translations in a standard fashion based on VLIS. Subsequently we replaced the translation probabilities $P(w_{NL}|w_{EN})$ by rough corpus based estimates. We simply looked up all English sentences which contained the translation and determined the proportion of the corresponding (aligned) Dutch sentences that contained the original Dutch query word. If the pair was not found, the original probability was left unchanged. Unfortunately a lot of the query terms and translations were not found in the aligned corpus, because they were lemmatised whereas the corpus was not lemmatised. At least this mismatch did hurt the estimates. The procedure resulted in high translation probabilities for words that did not occur in the corpus and low probabilities for words that did occur.

run name	avp	above median	description
tnoutne1	0.3069	27/33	standard
tnoutne1l	0.3278	-	+ doclen norm
tnoutne1p	0.3442	-	+$\lambda = 0.7$
tnoutne2	0.2762	25/33	corpus frequencies
tnoutne3-stem	0.3366	-	Porter stemmer +doclen norm
tnoutne4	0.2946	20/33	pseudo relevance feedback (PRF)
tnoutne4-fix	0.3266	-	PRF bugfix +doclen norm, Porter
tnoutne4-retro	0.4695	-	retrospective relevance feedback

Table 3: Results of the bilingual runs

The pseudo relevance feedback runs were done with the experimental language models retrieval engine at the University of Twente, using an index based on the Porter stemming algorithm. The run tagged with *tnoutne3-stem* is the baseline run for this system. The official pseudo relevance feedback run used the top 10 documents retrieved to re-estimate relevance weights and translation probabilities, but turned out to contain a bug. The unofficial fixed run *tnoutne4-fix* performs a little bit worse than the baseline. The run *tnoutne4-retro* uses the relevant documents to re-estimate the probabilities retrospectively (see e.g. [9]). This run reaches an impressive performance of 0.4695 average precision, much higher even than the best monolingual English run. This indicates that the algorithm might be helpful in an interactive setting where the user's feedback is used to retrieve a new, improved, set of documents. Apparently, the top 10 retrieved contains too much noise to be useful for the re-estimation of the model's parameters.

5.3 Multilingual runs

Table 4 shows that our best multilingual run was a run with Dutch as a query language. This is on one hand surprising (because this run is composed of 4 bilingual runs instead of 3 for the EN$\rightarrow$X run. But the translation is based on the VLIS lexical database which is built on lexical relations with Dutch as a source language. Thus the translations in the NL$\rightarrow$X case are much cleaner than the EN$\rightarrow$X case. In the latter case, Dutch serves as a pivot language. On the other hand, the NL$\rightarrow$IT translation is quite cumbersome. We first used Xelda to translate the Dutch queries to English stopped and lemmatised files. These files were subsequently translated by Systran.

Another interesting point is that the intermediate bilingual run based on the parallel web corpus performed quite well, with an average precision of 0.2750 versus 0.3203 of Systran. The translation of this run is based on a translation model trained on the parallel web corpus. The English topics were simply stopped and translated by the translation model. We took the most probable translation and used that as Italian query. We plan to experiment with a more refined approach where we import the translation probabilities into structured queries.

run name	avp	above median	description
tnoutex1	0.2214	25/40	baseline run
tnoutex2	0.2165	26/40	merged
tnoutex2f	0.2219	-	fixed
tnoutex3	0.1960	25/40	Web based EN-IT lexicon
tnoutnx1	0.2256	23/40	query language is Dutch

Table 4: Results of the $X \rightarrow EN, FR, DE, IT$ runs

5.4 The CLEF collection

This section reports on some of the statistics of the CLEF collection and compares it to the TREC cross-language collection. Table 5 lists the size, number of judged documents, number of relevant documents and the judged fraction, which is the part of the collection that is judged per topic.

collection	total docs.	judged docs.	relevant docs.	no hits in topic	judged fraction
english	110,250	14,737	579	2, 6, 8, 23, 25, 27, 35	0.0033
french	44,013	8,434	528	2, 4, 14, 27, 28, 36	0.0048
german	153,694	12,283	821	2, 28, 36	0.0020
italian	58,051	8,112	338	3, 6, 14, 27, 28, 40	0.0035
total	366,008	43,566	2,266		0.0022

Table 5: CLEF collection statistics, 40 topics (1-40)

collection	total docs.	judged docs.	relevant docs.	no hits in topic	judged fraction
english	242,866	18,783	2,645	26, 46, 59, 63, 66, 75	0.0014
french	141,637	11,881	1,569	76	0.0015
german	185,099	8,656	1,634	26, 60 ,75, 76	0.0008
italian	62,359	7,396	671	26, 44, 51, 60, 63, 75, 80	0.0021
total	631,961	46,716	6,519		0.0013

Table 6: TREC collection statistics, 56 topics (26-81)

Table 6 lists the same information for the TREC collection. The collections are actually quite different. First of all, the CLEF collection is almost half the size of the TREC collection and heavily biased towards German and English documents. Although the CLEF organisation decided to judge only the top 50 of documents retrieved and not the top 100 documents retrieved as in TREC, the number of documents judged per topic is only a little lower for the CLEF collection: about 814 documents per topic vs. 834 for TREC. Given the fact that the 56 TREC topics were developed over a period of two years and the CLEF collection has 40 topics already, the organisation actually did more work this year compared to pervious years. Another striking difference is the number of relevant documents per topic, only 57 for CLEF and 116 for TREC. This might actually make the decision to only judge the top 50 of runs not that harmful for the usefulness of the CLEF evaluation results.

6 Conclusions

This year's evaluation has confirmed that cross-language retrieval based on structured queries, no matter what the translation resources are, is a powerful technique. Re-estimating model parameters based on pseudo relevant documents does not result in improvement of retrieval performance. However, the relevance weighting algorithm shows an impressive performance gain if the relevant documents are used retrospectively. This indicates that the algorithm might in fact be a valuable tool for processing user feedback in an inter-active setting. Finally, merging based on the collection size re-estimation technique proved not successful. Further analysis is needed why the technique did not work on this collection, as it was quite successful on the TREC-8 collection.

Acknowledgements

We would like to thank the DRUID project for sponsoring the translation of the topic set into Dutch. We thank Xerox XRCE for making the Xelda morphological toolkit available to us. Furthermore we would like to thank Jiang Chen (RALI, Université de Montréal), Jian-Yun Nie (RALI) for help with the PTMiner web mining tools, and Michel Simard (RALI) for helping with the construction of aligned corpora and building translation models.

References

[1] 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.

[2] 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*, NIST Special Publication 500-242, pages 227–238, 1999.

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

[4] W. Kraaij, R. Pohlmann, and D. Hiemstra. Twenty-one at TREC-8: using language technology for information retrieval. In *Proceedings of the eighth Text Retrieval Conference TREC-8*, NIST Special Publications, 2000.

[5] W. Kraaij and R. Pohlmann. Viewing stemming as recall enhancement. In H.P. Frei, D. Harman, P. Schäuble, and R. Wilkinson, editors, *Proceedings of the 19th ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR96)*, pages 40–48, 1996.

[6] J.Y. Nie. Parallel web corpora for CLIR? In *Proceedings of CLEF 2000*, 2000.

[7] J.Y. Nie, M. Simard, P. Isabelle, and R. Durand. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts in the web. In *ACM-SIGIR'99*, pages 74–81, 1999.

[8] R. Pohlmann and W. 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.

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