Thomson Legal and Regulatory at NTCIR-4: Monolingual and pivot-language retrieval experiments

Isabelle Moulinier
Thomson Legal and Regulatory
Research and Development Group
610 Opperman Drive, Eagan, MN 55123, USA
Isabelle.Moulinier@thomson.com

Abstract

Thomson Legal and Regulatory participated in the CLIR task of the NTCIR-4 workshop. We submitted formal runs for monolingual retrieval in Japanese, Chinese and Korean. Our bilingual runs from Chinese and Korean to Japanese rely on English as a pivot language.

During our monolingual experiments, we compared building stopword lists using query logs to building stopword lists from collection statistics with further manual editing. We investigated decompounding for Korean, more precisely partial credit of compound parts. Finally we incorporated pseudo-relevance feedback in our Japanese runs.

Our bilingual approach was an experiment to construct a system within a short timeframe using publically available resources. The low quality of retrieval suggests that such an approach is not viable in a real environment.

Keywords: *stopword lists, Korean compounds, pseudo-relevance feedback, online resources.*

1 Introduction

Thomson Legal and Regulatory participated in the Cross-Lingual Information Retrieval task of the NTCIR-4 workshop. For this year's participation, we participated in four subtasks: monolingual Japanese retrieval, monolingual Chinese retrieval, monolingual Korean retrieval, and pivot bilingual retrieval using English as the pivot language. Characteristics of the tasks and collections are described in [10].

At NTCIR-3, we participated in the Japanese, Chinese and bilingual subtasks and investigated word versus character n-grams indexing and associated query syntax. NTCIR-4 is our first attempt at Korean and pivot bilingual retrieval.

With our monolingual experiments, we explore three directions: stopword lists, pseudo-relevance feedback for Japanese retrieval, and the handling of compound terms for Korean retrieval.

Our approach to pivot bilingual retrieval is a crude attempt to provide bilingual retrieval in a short timeframe by using publically available translation resources and tools. Our goal was to assess the quality of such systems built in a couple of days on top of a monolingual retrieval system.

In Section 2, we briefly present our base retrieval system. Section 3 describes our monolingual experiments with building stopword list. Section 4 summarizes our approach to handle compounds in Korean searches and Section 5 discusses our results with pseudo-relevance feedback. Section 6 reports on our pivot bilingual search effort. Conclusions are drawn in Section 7.

2 The WIN system

The WIN system is a full-text natural language search engine, and corresponds to TLR/West Group's implementation of the inference network retrieval model. While based on the same retrieval model as the INQUERY system [4], WIN has evolved separately and focused on the retrieval of legal material in large collections in a commercial environment that supports both Boolean and natural language searches [23].

2.1 Indexing

During indexing, we used words as indexing units. Words are identified using a third party tokenizer. For NTCIR-4, we used the tokenizer included in the LinguistX toolkit commercialized by Inxight [9]. Where appropriate, words are also stemmed using the same toolkit. In particular, stemming Korean terms using the toolkit helps us identify compound terms.

WIN does not apply a stopword list during indexing, but it does when searches are performed. As a result, all terms are indexed, although it is possible to omit some terms in document length statistics.

2.1.1 Document Scoring

WIN supports various strategies for computing term beliefs and scoring documents. We used a standard tf-idf for computing term beliefs in all our runs. The belief of a single concept is given by:

$$bel_{term}(Q) = 0.4 + 0.6 * ntf * nidf$$

where

$$ntf = \frac{\log(tf + 0.5)}{\log(tf_{max} + 1.0)} \quad (1)$$

$$nidf = \frac{log(C + 0.5) - log(df)}{log(C + 1.0)} \quad (2)$$

and tf is the number of occurrences of the term within the document, tf_{max} is the maximum number of occurrences of any term within the document, df is the number of documents containing the term and C the total number of documents in the collection. tf_{max} is an approximation for document length.

The document is scored by combining term beliefs using a different rule for each query operator [4]. The final document score is an average of the document score as a whole and the score of the best portion. The best portion is dynamically computed based on query term occurrences.

2.1.2 Query formulation

Query formulation identifies concepts in natural language text, and imposes a structure on these queries. The structure corresponds to the shape of the belief network. In many cases, each term in the natural language text represents a concept, and a flat structure gives the same weight to all concepts. In other cases, misspellings, phrases or compounds can introduce more complex concepts, using operators such as “natural phrase”, “compound”, or “synonym”.

Identifying concepts is based on removing terms that do not convey meaning. Stopwords are a typical example of such terms. Patterns that occur frequently in queries are another example, for instance phrases like “Find cases about” or “Relevant documents may include”. WIN relies on manually defined lists to perform that processing. At this point, we only identify stopwords for Asian languages.

3 Experiments with stopword lists

With this set of experiments, we focus on how to construct stopword lists with little or no language knowledge. In particular, we constrast leveraging collection information and query log information.

3.1 Prior work

In recent years, several approaches have been put forward to create stopword lists in the context of non-English document retrieval.

Savoy [22] relies on collection statistics and additional manual filtering. Savoy follows the method proposed by Fox [6] and applies it to several European languages.

Another common approach is to translate an English stopword list into the target language. Chen and Gey [2] propose a variant, where stopwords in Arabic are identified as translating to only English stopwords.

Finally, stopwords and noise patterns can be manually identified from queries. This is the current approach in WIN, where the English stopwords and noise patterns were manually identified. McNamee [12] relies on a similar approach, where English patterns are extracted from TREC query logs and later automatically translated.

3.2 Experiments

We constructed stopword lists for each language following two different approaches: collection and query log statistics.

Using collection statistics For each language, we extracted the n most frequent terms in the collection. In the reported experiments, we arbitrarily set $n = 300$ for Japanese and $n = 200$ for Chinese and Korean. We used the NTCIR-3 collection for Japanese, while we used NTCIR-4 data for Chinese and Korean. The Japanese and Chinese lists were further edited by a native speaker. The Korean stopword list was normalized using stemming.

Query log statistics We experimented with automatically extracting stopwords from query logs. Using all fields in NTCIR-3 queries, stopwords were identified as terms that occurred in more than x% of the queries. In the experiments reported below, we set $x = 20\%$. We found no significant differences when x was set between 20 and 40%.

3.3 Results and discussion

Table 1 summarizes statistics about stopword lists per language. Interestingly, lists generated from query logs are not subsets of lists generated from collection statistics. Some Japanese examples of stopwords only identified by the query log approach are 関する (to be related) or 満たす (to fulfill, to satisfy).

Table 2 reports average precision (MAP) when no stopword list is used (none), when the stopword list is built using collection statistics (collection), and when stopwords are extracted from query logs (query log).

Language	Collection	Query log	Overlap	Collection only (query log only)
Japanese	289	45	28	261 (17)
Korean	128	41	22	106 (19)
Chinese	117	38	22	96 (16)

Table 1. Summary of stopword list statistics. Each entry corresponds to the number of stopwords in the list. Terms in common are reported in the overlap column while differences are reported in the last column.

We observe that, on average, short queries (using the Title field only) are not affected by stopword processing. However certain individual short queries may be affected. For instance, Japanese query 012 performs worse after stopword removal using the collection-based list, because the term 明 (light, bright) is identified as a stopword. Similarly, Korean query 009 performs worse after stopword removal using the collection-based list.

Longer queries tend to benefit more from stopword processing, although results vary per language. The differences between the D, DN and TDNC runs with no stopword removal are interesting. One might expect the lack of stopword processing to affect longer queries more drastically. Our interpretation of the results is that, as queries grow longer, the influence of stopwords on document scores is diluted if query concepts are strongly identified. This is indeed the case when the Title and Concept field (run TDNC) are added to the more discursive fields Description and Narrative (runs D and DN).

It is interesting to note that Korean runs with description fields do not benefit from stopword processing. We have not yet been able to explain this behavior. Our intent is to further examine those differences and investigate the interaction between stopwords and compounds.

There is no statistical difference between stopword lists based on collection statistics and stopword lists based on query logs. Because there is little overlap between the stop lists, this suggests that our collection-based stopword list requiress further examination.

Most queries achieve the same performance under both condition. However we observe that certain individual queries are affected. For example, in the Japanese D run, query 058 performs better using the stoplist based collection statistics, while query 041 performs better using the stoplist based on query logs. In the case of query 058, the term 検索 (to retrieve) is not identified as a stopword using query logs, while it was part of the human-edited collection stopword list. Reciprocally, どの was not identified as a stopword because it was not in the stoplist based on collection statistics.

Similar examples can be found in Korean and Chinese. For instance, query 030 unexpectedly benefits from the query log based stoplist when 國家 is identified as a stopword.

Generating stopword lists from query logs is effective inasmuch as the anticipated queries follow the same patterns. A collection-based stopword list that is human edited is effective for more general queries.

In future work, we would like to revisit the arbitrary thresholds used in the experiments above. In particular, we aim to investigate whether the thresholds can be set automatically from collection and query log characteristics.

4 Experiments with Korean compounds

4.1 Prior research

We consider Korean as a compounding language, as it allows for the dynamic creation of terms by concatenating known words into sequences.

Yun et al [24] describe a retrieval model for Korean based on word formation. They identify the root of simple words, and the multiple roots in compound words. They propose a scoring algorithm that gives credit to terms that partially match compounds, and find the proposed method to perform on par with a n-gram approach.

During Cross-Lingual Evaluation Forum (CLEF) campaigns [3], researchers have found that, for German, Dutch or Finnish, breaking compounds into parts and searching on the parts was beneficial to both monolingual and crosslingual retrieval [8, 15].

Finally, character n-grams have been found effective for both European and Asian languages. The approach is to use character n-grams instead of or combined with words (cf. McNamee and Mayfield at CLEF[13] and previous research at NTCIR-2 and NTCIR-3 [17, 18]). By using n-grams, the problem of identifying compounds is alleviated.

4.2 Experiments

Our Korean experiments with decompounding build upon our experience with German compounds [16].

		Relax			Rigid		
Language	Fields	none	collection	query log	none	collection	query log
Japanese	T	0.3685	0.3657	0.3585	0.2684	0.2680	0.2637
Japanese	D	0.2812	0.3505⋆⋆	0.3584⋆⋆	0.2098	0.2647⋆⋆	0.2580⋆⋆
Japanese	DN	0.2960	0.4126⋆⋆	0.4036⋆⋆	0.2346	0.3173⋆⋆	0.3088⋆⋆
Japanese	TDNC	0.3557	0.4370⋆⋆	0.4264⋆⋆	0.2815	0.3368⋆⋆	0.3275⋆⋆
Chinese	T	0.2092	0.2080	0.2133	0.1783	0.1771	0.1792
Chinese	D	0.1793	0.1972⋆⋆	0.2016⋆⋆	0.1378	0.1536⋆⋆	0.1563⋆⋆
Chinese	DN	0.2138	0.2561⋆⋆	0.2590⋆⋆	0.1741	0.2093⋆⋆	0.2092⋆⋆
Chinese	TDNC	0.2350	0.2639⋆⋆	0.2680⋆⋆	0.1913	0.2143⋆	0.2177⋆
Korean	T	0.3166	0.3136	0.3139	0.2849	0.2821	0.2820
Korean	D	0.2601	0.2587	0.2748	0.2318	0.2297	0.2469
Korean	DN	0.3188	0.3469⋆⋆	0.3450⋆⋆	0.2875	0.3130⋆⋆	0.3105⋆⋆
Korean	TDNC	0.3499	0.3732⋆⋆	0.3694⋆⋆	0.3167	0.3382⋆⋆	0.3346⋆⋆

Table 2. Performance comparison between stopword processing. Performance is expressed at average precision. The ⋆⋆,⋆ sign indicates a statistical difference with the base run "none" with $\alpha = 0.01, 0.05$ using the sign test. Our Chinese and Korean official runs correspond to the collection column.

Indexing We index both simple terms, compound terms and parts of compound terms. This allows us to use a single index, but vary query formulation.

Query formulation We investigated different formulations: *No Decompounding* (ND), *Strict Phrases with Partial Credit* (StrictPC), *Loose Phrases without Partial Credit* (Loose), and *Loose Phrases with Partial Credit* (LoosePC). Due to our indexing scheme, *No decompounding* corresponds to strict phrases with no partial credit. Table 3 summarizes the differences between the query structures in the above approaches. Loose phrases correspond to a proximity of 3 between the terms in the phrase. Partial credit introduces compound parts as search concepts and allows part A from compound A#B in the query to match on term A, compound A#B, or compound A#C in documents.

ND	A#B
StrictPC	A#B$\langle w \rangle$ A$\langle w_1 \rangle$ B$\langle w_1 \rangle$
Loose	NPHR(A B)
LoosePC	NPHR(A B)$\langle w \rangle$ A$\langle w_1 \rangle$ B$\langle w_1 \rangle$

Table 3. Query formulation for compound term A#B. The weights w and w_1 control how much the compound and its parts respectively contribute to the score.

4.3 Results and discussion

Table 4 summarizes our experimental results with decompounding. In the reported experiments, we gave more importance to parts than to the compound itself by setting $w = w_1/2$. We are currently examining different weighting schemes on the compound parts to assess the impact of weights on results.

Some of the results surprised us. In particular, we expected *Loose phrases without Partial Credit* to perform at least as well as *No Decompounding* since loose phrases can capture compounds as well as phrases. Upon further examination of results, we noticed that loose phrases were actually too permissive and captured terms that were unrelated, instead of capturing compound terms. As a result, relevant documents were pushed down the result list.

Partial credit proved very helpful to counterbalance the influence of phrases, and somewhat helpful with strict phrases. This is consistent with the findings of Yun et al. [24] on partial matching.

However, we were disappointed by the lack of significant differences between *No Decompounding* and *Strict with Partial Credit* runs. This behavior can be explained by a bias introduced by our indexing scheme. While compound terms in queries can not match simple terms in documents, simple terms in queries can match both simple and compound terms in documents. We intend to evaluate the stricter approach where simple terms can only match simple terms to complete our effort on handling compounds.

5 Experiments with Pseudo-relevance feedback

Our last set of experiments focused on query expansion through pseudo-relevance feedback. We restricted these runs to the Japanese subtask.

There has been a lot of interesting research and results on the subject. For example, the relevance feedback incorporated in OKAPI BM-25 model has

	Relax				Rigid			
Fields	ND	StrictPC	Loose	LoosePC	ND	StrictPC	Loose	LoosePC
T	0.2904	0.3201	0.2756	0.3136⋆	0.2675	0.2899	0.2506	0.2821⋆
D	0.2300	0.2632	0.2052	0.2587⋆	0.2108	0.2365	0.1827	0.2297⋆
DN	0.3253	0.3495	0.3108	0.3469⋆	0.2959	0.3176	0.2821	0.3130⋆
TDNC	0.3471	0.3778⋆	0.3378	0.3732⋆	0.3183	0.3433	0.3072	0.3382⋆

Table 4. Average precision (MAP) of Korean runs. Our official runs correspond to LoosePC runs. The $\star$ sign indicates a statistical difference with the base run (No partial Credit) with $\alpha = 0.05$ using the sign test.

been successfull at CLEF (cf. [22]) and at NTCIR (e.g. [20]). Sakai and Sparck-Jones [21] and Lam-Adesina and Jones [11] investigated using document summaries to support pseudo-relevance feedback.

By contrast with recent developments, our approach is simpler and follows the work outlined by Haines and Croft [7].

5.1 Experimental settings

Term selection We use a Rocchio-like formula to select terms for expansion:

$$sw = \frac{\beta}{|R|}\sum_{d \in R}(ntf * nidf) - \frac{\gamma}{|\overline{R}|}\sum_{d \in \overline{R}}(ntf * nidf)$$

where R is the set of documents considered relevant, $\overline{R}$ the set of documents considered not relevant, and $|X|$ corresponds to the size of set X. ntf and $nidf$ are defined in Section 2.

Note that we select terms for expansion solely on the basis of documents. We do not favor terms that appear in the original query during term selection. The sets of documents R and $\overline{R}$ are extracted from the document list returned by the original search: R correspond to the top n documents, and $\overline{R}$ to the bottom m.

Reformulated query We append selected N terms to the original query, when the selected terms do not already appear in the query. In addition, each added term is weighted by the ntf part of the selection weight.

Parameter settings We used NTCIR-3 as a training corpus to select the number of relevant and non-relevant documents, as well as the number of terms to add to the query.

5.2 Results and discussion

During our training phase, we observed that our approach was very sensitive to the chosen parameters. The set of parameters selected during those runs seem to carry over to the NTCIR-4 runs. Table 5 summarizes the performance of these runs. Average precision and precision at 5 documents improved when pseudo-relevance feedback was added. However, differences are not statistically significant. Precision at 20 documents on the other hand tends to degrade with pseudo-relevance feedback.

A detailed analysis reveals that individual queries are greatly affected by pseudo-relevance feedback, either positively or negatively (cf. Table 6). Indeed, we observe that nearly half of the longer queries (runs DN and TDNC) have a variation greater than 10%, while 80% of the short queries (runs T) exhibit a similar variation.

We find the impact of relevance feedback with short queries less predictable. So far, we have identified the following factors to partially explain the variability of our results:

- the length of the original query,
- the number of relevant documents returned in the first n by the original search,
- the number of relevant documents returned in the bottom m documents by the original search, and
- the relative length of documents selected to extract terms for query expansion.

We finally discuss the behavior of search of a couple of queries, queries 026 and 012. Query 026 is negatively impacted by more than 40% in all relevance feedback runs, but not for the same reasons. The original T run returns no relevant document in the top 5, but returns one relevant document in the bottom 20. In the original DN and TDNC runs, one document in the top 5 is not relevant but its length is much greater than the length of other documents, and expansion terms are selected from that document. We need to study this phenomenom further to understand why ntf failed to prevent such selection.

On the other hand, query 012 is positively impacted by more than 20% when long queries (fields DN and TDNC) are used. In both cases, the original search returns 5 relevant documents in the top 5 and no relevant document at the bottom of the result list.

		Relax			Rigid		
Parameters	Fields	MAP	P5	P20	MAP	P5	P20
No PRF	T	0.3657	0.5782	0.5755	0.2680	0.4255	0.4064
$n = 5, m = 20, N = 20, \beta = \gamma = 1$[1]	T	0.3885	0.6145	0.5636	0.2965	0.4545	0.4127
$n = 20, m = 20, N = 5, \beta = \gamma = 1$[2]	T	0.3545	0.5964	0.5473	0.2719	0.4509	0.4091
No PRF	DN	0.4136	0.7055	0.6282	0.3178	0.5673	0.4709
$n = 5, m = 20, N = 20, \beta = \gamma = 1$[3]	DN	0.4337	0.7382	0.6436	0.3363	0.6109	0.4955
No PRF	TDNC	0.4372	0.7309	0.6536	0.3370	0.5782	0.4882
$n = 5, m = 20, N = 20, \beta = 1, \gamma = 4$[4]	TDNC	0.4466	0.7563	0.6673	0.3484	0.5927	0.5045
$n = 5, m = 20, N = 20, \beta = \gamma = 1$	TDNC	0.4466	0.7673	0.6545	0.3467	0.6255	0.4964

Table 5. Performance for pseudo-relevance feedback runs. [1] corresponds to official run tlrrd-t-02. [2] corresponds to run tlrrd-t-03. [3] corresponds to run tlrrd-dn-04. [4] corresponds to run tlrrd-tdnc-01

	Relax			Rigid		
	$\Delta > 10\%$ (+/-)	$\Delta > 20\%$ (+/-)	$\Delta > 40\%$ (+/-)	$\Delta > 10\%$ (+/-)	$\Delta > 20\%$ (+/-)	$\Delta > 40\%$ (+/-)
tlrrd-tdnc-01	24 (14/10)	12 (7/5)	2 (0/2)	28 (17/11)	17 (11/6)	3 (1/2)
tlrrd-t-02	45 (18/27)	35(13/22)	21 (6/15)	39 (16/23)	45 (21/24)	21 (7/14)
tlrrd-t-03	39 (19/20)	27 (14/13)	16 (7/9)	38 (21/17)	30 (17/13)	15 (10/5)
tlrrd-dn-04	27 (17/10)	18 (13/5)	4 (3/1)	30 (20/10)	18 (13/5)	5 (4/1)

Table 6. Number of queries affected positively (+) or negatively (-) by relevance feedback processing. Δ refers to the relative difference in average precision for each query.

To sum up, pseudo-relevance feedback is helpful for some queries, but not for others. We found the proportion to be very close to 50% and pseudo-relevance feedback, as we implemented it, not reliable enough. We believe that pseudo-relevance feedback could be rendered more effective if we could identify whether a query is likely to provide a good original first search. We plan to further investigate this issue, possibly building upon the approach proposed by Cronen-Townsend et al [5].

6 Bilingual experiments using a pivot language

Our involvement with bilingual retrieval was minimal. We attempted to provide bilingual retrieval in a short timeframe by using publically available translation resources and tools.

6.1 Building a bilingual driver

Our approach consisted in building a translation layer on top of our monolingual search engine, with no changes to the search engine.

To construct the translation layer, we were faced with direct translation and translation through a pivot language. An initial Web search failed to provide us with online tools that could translate Chinese and Korean into Japanese. Thus we decided upon the pivot language approach, having found resources for the English-Chinese, English-Korean and English-Japanese pairs of languages.

We build some tools to automatically query two online resources: Babelfish [1] and the Chinese-English online dictionary [14].

For our Chinese-English-Japanese experiments, we used word-by-word translation, while we translated whole sentences during our Korean-English-Japanese runs.

Chinese-Japanese At the time of the experiments, Babelfish was not supporting Chinese to English translation[1]. We relied on the MDBG Chinese-English dictionary to translate Chinese terms into possibly multiple English terms, and Babelfish to translate an English term into a single Japanese term. When multiple English translations were found, we grouped their corresponding Japanese versions under a SUM node, thus giving the same importance in the final structure query to terms with a single translation and terms with many translation.

Chinese concepts were identified by removing stopwords; English stopwords were not translated to Japanese. Finally, when an English term had no Japanese translation, we substituted the original Chinese term.

[1] We may have failed to automate client to query Babelfish for the Chinese-English language pair.

Korean-Japanese We translated the whole Korean query to English using Babelfish. The translated English sentence was in turn translated to Japanese, again using Babelfish. Japanese stopwords were removed as part of the regular Japanese query processing part of the search engine.

Overall, it took us a couple of days to build the translation tools and integrate them into a search driver.

6.2 Results

Table 7 summarizes the performance of our official runs. Without surprise, the performance is rather poor, in Chinese-Japanese runs in particular. We suspect that our Chinese query processing and stopword identification fail to identify good search concepts. This is consistent with our monolingual Chinese runs, where our performance is below average.

Following these results, we believe that bilingual retrieval and in particular pivot language bilingual retrieval can not be performed by adding a simple translation component to a monolingual search engine.

We may explore a number of different approaches. First, using English as a pivot language may not be suited for Asian languages. We may want to examine using an Asian language instead. Next, TLR has done some work with similarity thesauri and building bilingual lexicons from corpora in Western European languages. We may be able to extend our approach to non-European languages.

7 Conclusion

TLR submitted runs in monolingual and bilingual retrieval for this year's NTCIR workshop.

Our approach to bilingual did not focus on research issues, but rather resource availabilities and rapid implementation. We use English as a pivot language because free machine translation tools or bilingual dictionary are readily available for that language. Our poor performance during the workshop show that less crude approaches are required to address crosslingual search satisfactorily.

For our monolingual experiments, we explored three issues: creating stopword lists automatically, handling Korean compounds during search, and expanding queries using pseudo-relevance feedback. While our performance in the monolingual subtasks was below our expectation, we consider our participation helpful to our research.

We found that, when queries follow an anticipated format, query logs can be successfully leveraged to extract stopword lists. However, in the case of more general queries, we forecast that the more traditional approach using collection statistics and human editing is more robust. In future work, we plan to investigate how best to select terms using collection statistics to limit manual editing.

With the Korean retrieval subtask, we studied the impact of compounds on search. In particular, we showed that partial credit, i.e. using compound parts as well as the whole compound, was promising. Our future effort may focus on different weighting schemes for partial credit.

Our first study of pseudo-relevance feedback produced mixed results. This may be a consequences of our choices and implementation of the approach. However, our analysis has prompted future work and we may focus on predicting how well queries can perform for a given collection.

Acknowledgements

The authors would like to thank the NTCIR-4 organizers for their effort, and Elizabeth Lund for her help in reviewing the paper.

References

[1] http://babelfish.altavista.com.

[2] A. Chen and F. Gey. Building an arabic stemmer for information retrieval. In *The Eleventh Text Retrieval Conference*, number SP 500-251. NIST Special Publication, 2002.

[3] http://www.clef-campaign.org.

[4] W. B. Croft, J. Callan, and J. Broglio. The inquery retrieval system. In *Proceedings of the 3rd International Conference on Database and Expert Systems Applications*, Spain, 1992.

[5] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of the Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299–306, 2002.

[6] C. Fox. A stop list for general text. *ACM SIGIR Forum*, 24(2), Fall 89/Winter 90 1990.

[7] D. Haines and W. Croft. Relevance feedback and inference networks. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2–11, 1993.

[8] T. Hedlund, H. Keskustalo, A. Pirkola, E. Airio, and K. Järvelin. Utaclir @ clef 2001: New features for handling compound words and untranslatable proper names. In Peters et al. [19].

[9] http://www.inxight.com/products/oem/linguistx.

[10] K. Kishida, K.-H. Chen, S. Lee, K. Kuriyama, N. Kando, H.-H. Chen, S. H. Myaeng, and K. Eguchi. Overview of clir task at the fourth ntcir workshop. In *Proceeding of the Fourth NTCIR Worshop*, 2004.

[11] A. M. Lam-Adesina and G. Jones. Applying summarization techniques for term selection in relevance feedback. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 1–9, 2001.

Run ID	MAP	Below Med/Avg	Equal Med/Avg	Above Med/Avg	MAP	Below Med/Avg	Equal Med/Avg	Above Med/Avg
tlrrd-C-J-T-01	0.1306	35/42	10/1	12/10	0.1065	38/44	9/1	8/10
tlrrd-C-J-D-02	0.0722	43/50	1/0	10/0	0.0544	44/51	1/1	9/2
tlrrd-K-J-T-01	0.1412	27/43	17/0	11/13	0.1116	30/43	14/0	11/12
tlrrd-K-J-D-02	0.1211	41/43	2/0	11/11	0.0964	39/42	2/1	13/11

Table 7. Summary of pivot language bilingual runs

[12] P. McNamee. Knowledge-light asian language text retrieval at the ntcir-3 workshop. In NTCIR3 [18].

[13] P. McNamee and J. Mayfield. A language-independent approach to european text retrieval. In C. Peters, editor, *Cross-Language Information Retrieval and Evaluation*, number 2069 in LNCS. Springer, September 2000.

[14] http://www.mdbg.net/chindict/chindict.php.

[15] C. Monz and M. de Rijke. The university of amsterdam at CLEF 2001. In Peters et al. [19].

[16] I. Moulinier, J. A. McCulloh, and E. Lund. West group at CLEF 2000: Non-english monolingual retrieval. In C. Peters, editor, *Cross-Language Information Retrieval and Evaluation*, number 2069 in LNCS. Springer, September 2000.

[17] *Proceedings of the Second NTCIR Workshop on Research in Chinese and Japanese Text Retrieval and Text Summarization*, Tokyo, Japan, 2001.

[18] *Proceedings of the Third NTCIR Workshop on Research in Chinese and Japanese Text Retrieval and Text Summarization*, Tokyo, Japan, 2002.

[19] C. Peters, M. Brashler, J. Gonzalo, and M. Kluck, editors. *Evaluation of Cross-Language Information Retrieval Systems*, number 2406 in LNCS. Springer, September 2001.

[20] T. Sakai, M. Koyama, M. Suzuki, and T. Manabe. Toshiba kids at ntcir-3: Japanese and english-japanese ir. In NTCIR3 [18].

[21] T. Sakai and K. Sparck-Jones. Generic summaries for indexing in information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 190–198, 2001.

[22] J. Savoy. Report on clef-2001 experiments: Effective combined query-translation approach. In Peters et al. [19].

[23] H. Turtle. Natural language vs. boolean query evaluation: a comparison of retrieval performance. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 212–220, Dublin, Ireland, 1994.

[24] B.-H. Yun, M.-J. Cho, and H.-C. Rim. Korean information retrieval model based on the principles of word formation. In *Second International Workshop on Information Retrieval with Asian Languages*, Japan, 1997.