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.

Our monolingual experiments are three-fold: we investigated decompounding for Korean, in particular partial credit of compound parts; we integrated query expansion in our Japanese runs; and we explored various sources for building stopword lists.

Our bilingual approach was an experiment to construct a system within a short timeframe using publicly available resources. The low quality of retrieval suggests that such an approach is not viable in a production 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 campaign, 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 explored three directions: pseudo-relevance feedback for Japanese retrieval, the handling of compound terms for Korean retrieval, and the creation of stopword lists for all three languages.

Our approach to pivot bilingual retrieval is a crude attempt to provide bilingual retrieval in a short timeframe by using publicly available translation resources and tools. Our goal was to assess whether such an approach could provide quality suitable for a production system.

In Section 2, we briefly present our base retrieval system. Section 3 summarizes our approach to handle compounds in Korean searches, Section 4 discusses our results with pseudo-relevance feedback, and Section 5 describes our monolingual experiments with building stopword list. 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 addition, the stemmer identifies compound terms and their components. We use this feature in our Korean experiments.

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:

$$bel_{term}(Q) = 0.4 + 0.6 * tf_{norm} * idf_{norm}$$

where

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

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

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 a weak 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. However phrases, compounds or misspellings 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 Korean compounds

In this section, we investigate how indexing and search can leverage compound terms and their components effectively. To that end, we assume that compound terms and components can be identified prior to indexing and query formulation, for example using the LinguistX toolkit.

3.1 Prior research

We consider that Korean is 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].

Alternatively, some research has focused on using character n-grams as indexing units for both Asian and European languages (cf. McNamee and Mayfield at CLEF[13] and previous research at NTCIR-2 and NTCIR-3 [17, 18]). Such approaches alleviate the issue of identifying compound terms and their components. Indeed character n-grams may capture compound components without explicitly identifying components.

3.2 Experiments

At CLEF-2000, we investigated query formulation to capture German compounds and their components [16]. In this section, we build upon our past experience with German and evaluate a combination of indexing and query formulation approaches for Korean search. Compound terms are identified using the LinguistX morphological analyzer in both documents and queries.

Indexing We index both simple terms, compound terms, and components of compound terms. For example, when the term 홈런경쟁에서 is normalized to the compound term 홈런#경쟁, we index 홈런#경쟁 as 홈런#경쟁 as well as 홈런 and 경쟁. 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. The *Loose Phrases without Partial Credit* configuration corresponds to the INQUERY unordered distance operator of 3 (`#uw3`). The *Loose Phrases with Partial Credit* configuration is related to the INQUERY `#phrase` operator.

Partial credit introduces compound parts as search concepts and allows part 경쟁 from compound 홈런#경쟁 in the query to match on term 경쟁, compound 홈런#경쟁, or compound 과당#경쟁 in documents.

Table 1 summarizes the differences between the query structures in the four approaches.

3.3 Results and discussion

Table 2 summarizes our experimental results on leveraging compound information. In the reported experiments with *Partial Credit*, we gave more importance to parts than to the compound itself by setting $w = w_1/2$. Further investigation is required to examine how different weighting schemes may influence retrieval effectiveness.

Experimental results suggest that partial credit is helpful. The difference is statistically significant in all *Loose Phrases* configuration, but only in the TDNC configuration of *Strict Phrases*. Our results on the use of compound parts is consistent with the findings of Yun, et al. [24] on partial matching.

The larger effect observed on *Loose Phrases* may be the result of our choice of weights, where the whole compound construct is not allowed to contribute as much as the compound parts. Consequently, the detrimental effect of *Loose Phrases* is attenuated.

The lesser effect observed on *Strict Phrases* may be explained by a bias introduced by our indexing scheme when we indexed both compounds and their parts. In *No Decompounding* runs, compounds in queries will only match the identical compounds in documents; simple terms, on the other hand, may match the identical simple terms in documents as well as compound parts. Before we can conclude our effort on handling compounds, we intend to evaluate *No Decompounding* under a non-biased indexing approach, where simple terms in queries can only match simple terms in documents.

Experimental results show that *Strict with Partial Credit* and *No Decompounding* (resp. `StrictPC` and `ND`) runs outperform the corresponding *Loose Phrases* (resp. `LoosePC` and `Loose`) runs in all configurations, albeit with no statistically significant difference. This result was not anticipated. We expected *Loose Phrases* to perform at least as well as *Strict Phrases* since a compound found in a document would also satisfy the *Loose Phrase* operator. Upon further examination of individual results, we noticed that *Loose Phrases* captured terms that were unrelated to the original compound query term. As a result, relevant documents were pushed further down the retrieved list.

4 Experiments with pseudo-relevance feedback

Our next 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 been successful 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].

4.1 Experimental settings

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

$$sw = \frac{\beta}{|R|}\sum_{d\in R}(tf_{norm} * idf_{norm}) - \frac{\gamma}{|\overline{R}|}\sum_{d\in \overline{R}}(tf_{norm} * idf_{norm}) \quad (3)$$

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. tf_{norm} and idf_{norm} 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 tf_{norm} part of the selection weight. Weights of original query terms remain unchanged.

Parameter settings We used NTCIR-3 as a training corpus to select the following parameters: n, the number of relevant documents, m the number of non-relevant documents, and N the number of terms used in query expansion.

4.2 Results and discussion

During our training phase, we observed that our approach was very sensitive to the chosen parameters.

	No Partial Credit	Partial Credit
Strict (ND)	홈런#경쟁	홈런#경쟁$\langle w \rangle$ 홈런$\langle w_1 \rangle$ 경쟁$\langle w_1 \rangle$
Loose	NPHR(홈런 경쟁)	NPHR(홈런 경쟁)$\langle w \rangle$ 홈런$\langle w_1 \rangle$ 경쟁$\langle w_1 \rangle$

Table 1. Query formulation for compound term 홈런#경쟁. The weights w and w_1 control how much the compound and its parts, respectively, contribute to the score.

	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 2. 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.

This sensitivity is found in our official runs as well with the examples of tlrrd-t-02 and tlrrd-t-03. Overall, the set of parameters selected during our training phase seems to carry over to the NTCIR-4 runs. Table 3 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 4). Indeed, we observe that nearly half of the longer queries (runs DN and TDNC) have a relative difference between the initial run and the relevance-feedback run 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. 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 now discuss the behavior of search for 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 tf_{norm} 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.

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].

5 Experiments with stopword lists

Our last monolingual experiments focus on how to construct stopword lists with little or no language knowledge. In particular, we contrast leveraging collection information and query log information.

5.1 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 pro-

		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

Table 3. 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 4. Number of queries affected positively (+) or negatively (-) by relevance feedback processing. Δ refers to the relative difference in average precision for each query.

posed 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.

5.2 Experiments

We used two types of information to construct stopword lists: collection and query log statistics. When possible, we also relied on manual review of the lists created using collection statistics.

Using collection statistics For each language, we extracted the n most frequent terms in the collection. For the reported experiments, we use $n = 200$ for all languages[1]. The selected terms were further stemmed.

Manually-edited lists For Japanese and Chinese, the statistically-generated lists were also manually edited. Editors were given the choice to limit or add terms to lists.

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 varied between 20% and 40%. Lists were also normalized using stemming.

5.3 Results and discussion

Table 5 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 6 reports average precision (MAP) when no stopword list is used (`none`), when the stopword list is built using collection statistics (`collection`) and the list is further edited (`manual`), and when stopwords are extracted from query logs (`query log`).

The first conclusion we can draw is that stopword lists, however created, are a useful tool to render search more effective. We observe statistically significances differences for most of the runs.

The next observation we make is an exception to

[1]This value was set through limited experiments using values used in prior research.

Language	Manual	Collection	Query log	Overlap
Japanese	289	200	45	28[a]
Chinese	117	200	38	22[a]
Korean	–	128	41	22[b]

Table 5. 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. [a] Terms in common between Manual and Query log. [b] Terms in common between Collection and Query log.

that first conclusion: stopword processing has little influence on short queries (using Title field only) on average. This can be explain by the fact that most short queries contain very few stopwords if at all. However, certain individual queries may be influenced. For example, Japanese query 012 performs worse after stopword removal using the collection-based lists (collection and manual), 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. However, NTCIR queries repeat terms (concepts) in the various fields. When the retrieval system uses query term occurrences as weights, as is the case in WIN, the effect of stopwords is typically diminished by these stronger query concepts. 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.

We find no statistical differences between runs using stopword lists, independant of the method of list generation. One reason is that the stopwords common to all lists are the most frequently used in queries. However, because there is little overlap between the stop lists, this suggests that we may further refine our collection-based stopword lists.

Most queries achieve the same performance under both conditions. However we observe that certain individual queries are affected. For example, in the Japanese D run, query 058 performs better using the stoplist based on 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 as the queries used to build the list. Collection-based stopword lists that are human edited are expected to be more generally applicable.

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.

6 Bilingual experiments using a pivot language

Our involvement with bilingual retrieval was limited. We investigated pivot-language bilingual retrieval using publicly available translation resources and tools. One of our objectives was to build a bilingual system in a short timeframe.

6.1 Building a bilingual driver

Our approach consisted of 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 built 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

		Relax				Rigid			
Lang.	Fields	none	manual	collection	query log	none	manual	collection	query log
JA	T	0.3685	0.3580	0.3657	0.3585	0.2684	0.2639	0.2680	0.2637
JA	D	0.2812	0.3335⋆	0.3505⋆⋆	0.3584⋆⋆	0.2098	0.2448⋆	0.2647⋆⋆	0.2580⋆⋆
JA	DN	0.2960	0.4148⋆⋆	0.4126⋆⋆	0.4036⋆⋆	0.2346	0.3203⋆⋆	0.3173⋆⋆	0.3088⋆⋆
JA	TDNC	0.3557	0.4365⋆⋆	0.4370⋆⋆	0.4264⋆⋆	0.2815	0.3387⋆⋆	0.3368⋆⋆	0.3275⋆⋆
CH	T	0.2092	0.2025	0.2080	0.2133	0.1783	0.1692	0.1771	0.1792
CH	D	0.1793	0.1874⋆	0.1972⋆⋆	0.2016⋆⋆	0.1378	0.1484⋆	0.1536⋆⋆	0.1563⋆⋆
CH	DN	0.2138	0.2574⋆⋆	0.2561⋆⋆	0.2590⋆⋆	0.1741	0.2058⋆⋆	0.2093⋆⋆	0.2092⋆⋆
CH	TDNC	0.2350	0.2648⋆⋆	0.2639⋆⋆	0.2680⋆⋆	0.1913	0.2146⋆	0.2143⋆	0.2177⋆
KR	T	0.3166	–	0.3136	0.3139	0.2849	–	0.2821	0.2820
KR	D	0.2601	–	0.2587	0.2748	0.2318	–	0.2297	0.2469
KR	DN	0.3188	–	0.3469⋆⋆	0.3450⋆⋆	0.2875	–	0.3130⋆⋆	0.3105⋆⋆
KR	TDNC	0.3499	–	0.3732⋆⋆	0.3694⋆⋆	0.3167	–	0.3382⋆⋆	0.3346⋆⋆

Table 6. 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 official runs correspond to the manual column, while our Korean official runs correspond to the collection column.

whole sentences during our Korean-English-Japanese runs.

Chinese-Japanese At the time of the experiments, Babelfish was not supporting Chinese to English translation. 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 from the original queries by removing stopwords. English stopwords were not translated to Japanese. When no Japanese translation was found for a given English term, we used the original Chinese term rather than the English term in the translated Japanese query.

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. The performance is rather poor. In particular, we suspect that our Chinese query processing, including stopword identification, fails to select good terms prior to translation. This observation is consistent with our monolingual runs.

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

We anticipate exploring a number of alternative approaches. First, using English as a pivot language may not be suited for Asian languages, as suggested by some participants at NTCIR. Another related Asian language may be a better pivot language candidate. Next, we question the translation resources used in our experiments. We have performed some research on building bilingual lexicons from corpora for European languages. We intend to investigate such approaches for Asian languages.

7 Conclusion

We explored three issues with our monolingual experiments and approached our bilingual participation as an engineering effort. While our performance may not have met our expectations, we consider our participation helpful to our research.

We investigated how identifying Korean compounds could render search more effective, in particular using partial credit. Future efforts may focus on alternative operators for capturing compounds, and the influence of weighting schemes on the contribution of partial credit.

We took a first step towards pseudo-relevance feedback. Our results were mixed, but our analysis empha-

	Relax				Rigid			
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

sized the volatility of query expansion. Future work may involve investigating when queries should be expanded.

We found that stopword removal was useful for longer queries, independent of how the stopword list was created. We also found that query logs could provide useful information to construct stopword lists in specific circumstances.

Our approach to bilingual search was an engineering experiment: we integrated Web resources into our retrieval system. The poor performance of the resulting bilingual system indicates that bilingual search still is a research issue that requires attention.

Acknowledgements

The authors would like to thank the NTCIR-4 organizers for their effort.

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*, 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*.

[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*.

[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*.

[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, 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, 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, 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*.

[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*.

[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.