RMIT and Gunma University at NTCIR-9 Intent Task

Michiko Yasukawa* J. Shane Culpepper† Falk Scholer† Matthias Petri†

*michi@cs.gunma-u.ac.jp †{shane.culpepper, falk.scholer, matthias.petri}@rmit.edu.au

Introduction

Self-indexing algorithms have interesting theoretical and practical performance on basic pattern matching operations, but ranked search capabilities on large datasets is still open. We investigate the problem of using self-indexing algorithms to solve the **ranked document search** problem.

Ranked Self-Indexing

For the NTCIR-9 INTENT task, we used our experimental search engine, NEWT. NEWT is an enhanced version of the *greedy* top-k approach described in [1]. All prior published work on ranked self-indexes use a trivial TF×IDF ranking metric, and have generally focused on phrase queries instead of bag-of-words queries. For the INTENT task, two bag-of-words ranking functions were implemented. The first metric is referred to as *raw term frequency* ranking. For this metric, we simply compute the aggregate of raw frequency counts per document, $f_{t,d}$, for each term or substring, t.

$$\text{RAW} = \sum_{t \in q} f_{t,d}$$

We also implemented a simple BM25 variant as follows:

$$\text{BM25} = \sum_{t \in q} \log\left(\frac{N - f_t + 0.5}{f_t + 0.5}\right) \cdot \text{TF}_{\text{BM25}}$$

$$\text{TF}_{\text{BM25}} = \frac{f_{t,d} \cdot (k_1 + 1)}{f_{t,d} + k_1 \cdot ((1 - b) + (b \cdot \ell_d / \ell_{avg}))}$$

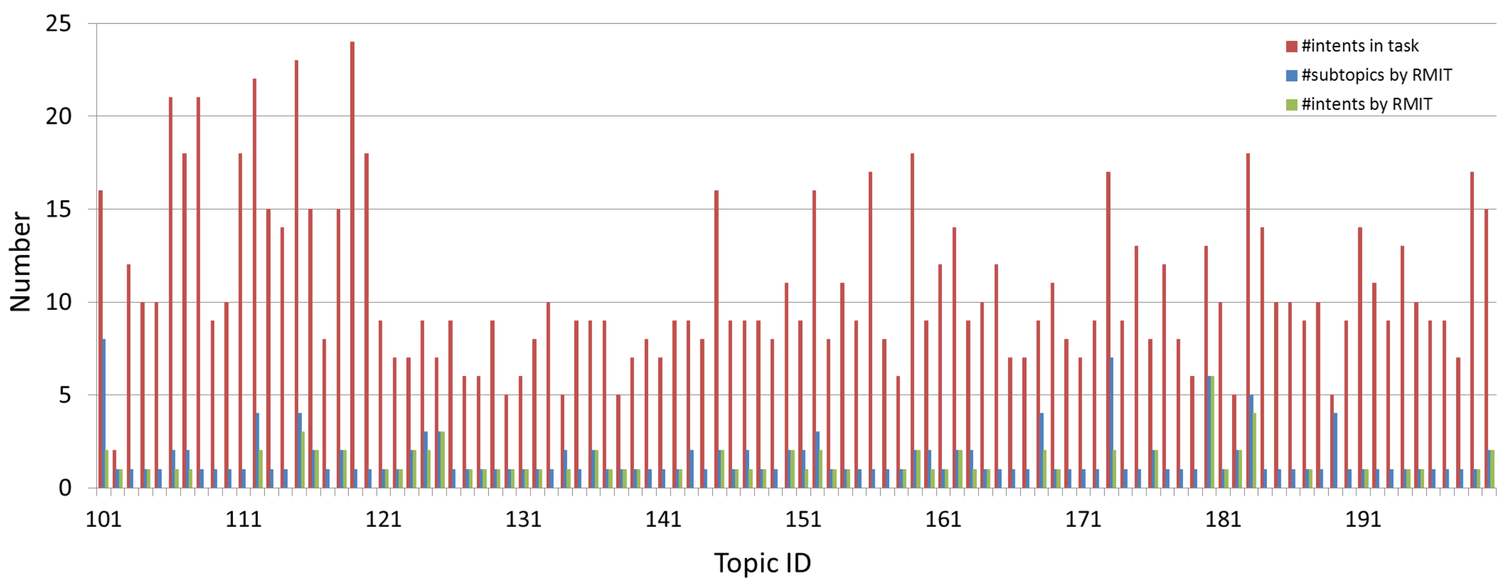
Here, N is the number of documents in the collection, f_t is the number distinct documents appearances of t, $k_1 = 1.2$, $b = 0.75$, ℓ_d is the number of UTF8 symbols in the documents, and ℓ_{avg} is the average of ℓ_d over the whole collection. For self-indexes, there is an efficiency trade-off between locating the top-k $f_{t,d}$ values and accurately determining f_t. Finding the most efficient trade-off is a topic of future work.

References

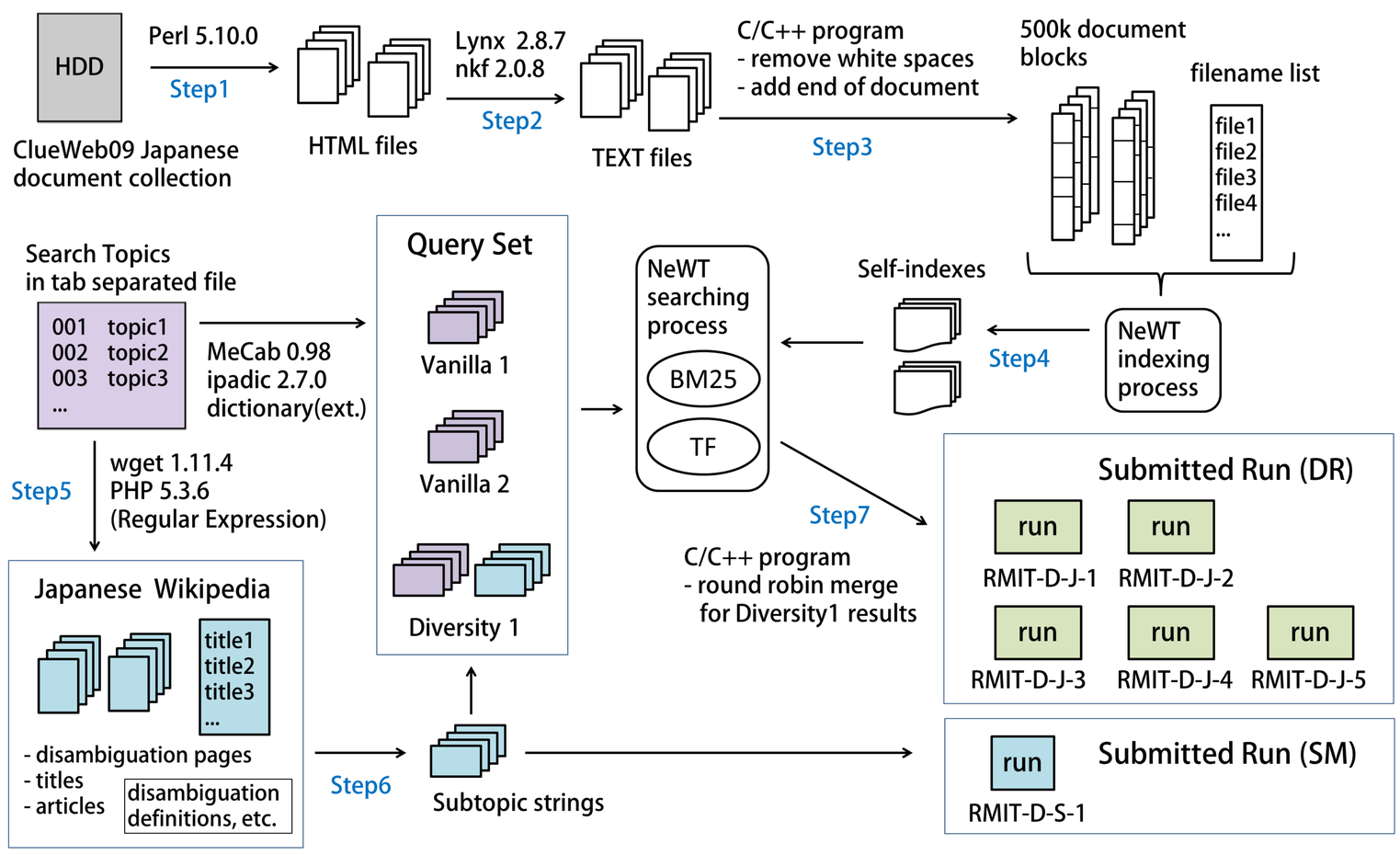
[1] J. S. Culpepper, G. Navarro, S. J. Puglisi, and A. Turpin. Top-k ranked document search in general text databases. In M. de Berg and U. Meyer, editors, *Proceedings of the 18th Annual European Symposium on Algorithms (ESA 2010), Part II*, volume 6347 of *LNCS*, pages 194–205. Springer, 2010.

Subtopics and Intents

Our run for the subtopic mining task did not identify many subtopic strings. Consequently, our intent recall was low.



RMIT Runs



Collection Processing

We indexed the `ClueWeb09-JA` collection as follows. First, each document from the collection was extracted (Step1) and normalized using Lynx. The extracted documents were converted to UTF8 character code (Step2). Next, all whitespace was removed from each document to create contiguous UTF8 strings, followed by a distinct `end of document` identifier (Step3). The fully processed `ClueWeb09-JA` collection was partitioned into blocks of 500,000 documents and indexed with NEWT (Step4).

Topic Processing

We processed the search topics and generated two vanilla query sets (`Vanilla 1`, `Vanilla 2`) and a diversity query set (`Diversity 1`).
For `Vanilla 1` and `Vanilla 2`, word segmentation of search topics was performed using MeCab. All morphemes were used in `Vanilla 1`, but only nouns were used in `Vanilla 2`. For `Diversity`, search topics were retained without word segmentation, and subtopic strings were obtained from the Japanese Wikipedia used for query expansion.

Query Diversification

Topic Disambiguation: To identify "Disambiguation in Wikipedia" in titles, we used a set of round parentheses "(" and ")" as tokens for pattern matching. We also used Perl Compatible Regular Expressions(PCRE), incorporated into the scripting language PHP, to gather disambiguation definitions from articles.
Wikipedia summaries in topics: We performed word segmentation on the introduction by using MeCab, and chose the last noun in the first sentence. The introduction of a Wikipedia article is a summary of the most important aspects.
Topics as queries: Some search topics were not listed in the Japanese Wikipedia, and were excessively specific. In these instances, the query was represented as both the search topic and all of the nouns in the search topic.

Results

The best performance was achieved by run `RMIT-D-J-3` which used the `Diversity 1` queries with TF weighting, and a round-robin re-ranking approach for diversification of the results list.

Table1: Run and Query Set.

Run	Vanilla 1	Vanilla 2	Diversity 1
D-J-1			★BM25
D-J-2		★BM25	
D-J-3			★TF
D-J-4	★BM25		
D-J-5		★TF	

Table2: Effectiveness Results.

Run	I-rec@30	D-nDCG@30	D#-nDCG@30
D-J-4	0.8012	0.3617	0.5814
D-J-3	0.7836	0.3800	0.5818
D-J-2	0.7977	0.3575	0.5776
D-J-1	0.7752	0.3617	0.5684
D-J-5	0.6759‡	0.3118†	0.4938‡

† and ‡ indicate statistical significance relative to the baseline at the 0.05 and 0.001 levels, respectively, based on a paired t-test.