Simple Document-by-Document Search Tool "Fuwatto Search" using Web API

Masao Takaku[1] and Yuka Egusa[2]

[1] University of Tsukuba masao@slis.tsukuba.ac.jp
[2] National Institute for Educational Policy Research yuka@nier.go.jp

Abstract. In this paper, we propose a new search method *Fuwatto Search* that allows users to retrieve documents in a document-by-document manner via a Web API. We present an implementation of the proposed method (i.e., Fuwatto CiNii Search), which targets the CiNii Article database, one of the largest academic article databases in Japan. The experimental evaluation of Fuwatto CiNii Search with newspaper articles demonstrates the retrieval effectiveness of 0.25 for precision at 10 and 0.17 for mean average precision.

Keywords: document retrieval, Web API, effectiveness, CiNii Articles

1 Introduction

Digital libraries, domain-specific databases, and Web search engines are important for our daily lives. When we need certain information for health, fashion, travel, and many other areas, we typically use Web-based search services.

Most search services support keyword-based queries. In other words, we must specify keywords that are appropriate for our information needs. Keyword-based queries can cause a gap between users and search systems because users do not always have enough knowledge on how to express appropriate keywords that correspond to their information needs and database content.

One solution for this issue is to use another querying method, i.e., the document-as-a-query method. A search system that supports document-by-document queries accepts an existing document as a query and returns a list of similar documents. Currently, some news websites, electronic commerce websites, and digital library services support this function. On such websites, a user can obtain a list of other items that are similar to the current item on that page. Such content-based similarity search functions can be beneficial in certain situations. However, these functions are not always available for all databases.

In this paper, we propose a new content-based similarity search methodology, "Fuwatto Search." Fuwatto Search allows users to search any arbitrary search service with a document-by-document query via a simple keyword-based search Web API.

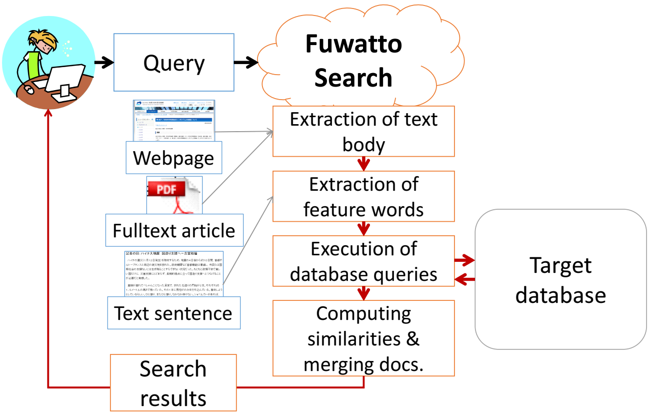


Fig. 1. Overview of Fuwatto Search

2 Proposed Methodology

An overview of the proposed *Fuwatto Search* methodology is shown in Figure 1.

Fuwatto Search is a document-by-document search method for remote databases. Fuwatto Search can search target databases that do not support document-by-document search functions. It iteratively accesses the keyword search functionality of a target database, and then analyzes and re-ranks the retrieved results. This process is described as follows.

1. Extraction of full text
   Fuwatto Search accepts PDF, HTML, and plain text data as a query. When a PDF or HTML page is given to Fuwatto Search as a query, the system first extracts the text content. For web pages written in HTML, the page header, footer and navigational elements are excluded. The system extracts the main body elements (i.e., the text) of an HTML page. We use the `extractcontent.rb` text extraction tool to extract main body elements from an HTML page [1].
2. Extraction of feature words with weights
   Extracted words are segmented using a Japanese morphological analyzer (i.e., MeCab). We simultaneously count the term frequency (TF) in the query document. TF is multiplied by the term occurrence cost, which is pre-defined in the MeCab dictionary.
3. Search execution
   We select the top n words from a vector of the feature words (computed by the previous process). Each of the top n words is sent as a query to the target database to determine if there is a match. If no hits are returned for a given word, we exclude that word from the set of feature words. We then tests the next feature word using the same process.

4. Merging search results
   The system merges all documents returned as search results into a single ranked list of retrieved documents in terms of specificity and similarity, and then presents the ranked results to the user.

Details of the process are described in Section 3.3.

This procedure was designed to avoid the zero hit problem, which returns no search results. This allows users to obtain as many relevant results as possible. This methodology can search any database in a document-by-document manner, even if the target database does not support any capability to accept documents as queries.

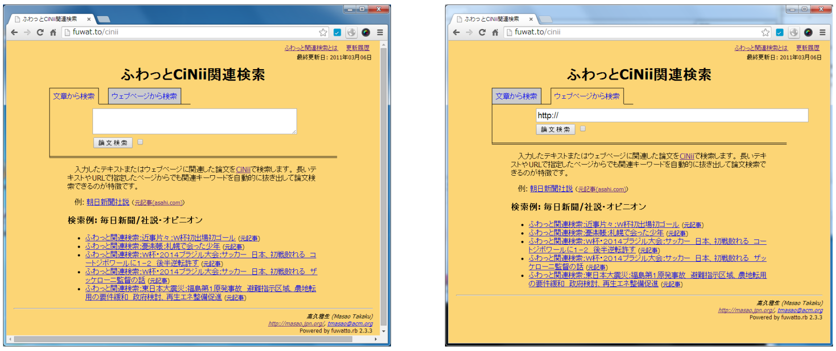
2.1 Mashup Using CiNii API

Here, we describe "Fuwatto CiNii Search," which uses CiNii Articles [2] as a target database for Fuwatto Search. Starting in January 2010, we developed Fuwatto CiNii Search as an implementation of the proposed methodology. Fuwatto CiNii Search is freely available at `http://fuwat.to/cinii`.

CiNii Articles is a Japanese academic article database service provided by the National Institute of Informatics. It contains bibliographic information for approximately 15 million articles. In addition, it supports a simple keyword-based search Web API using the OpenSearch protocol [3].

Fuwatto CiNii Search searches CiNii Articles database using the OpenSeach API. It uses a content extraction module for web pages, i.e., extractcontent.rb [1]. This content extraction module extracts body text from HTML web pages with block-based heuristics and simple rules for Japanese characters. The morphological analysis tool MeCab [4] is used to segment the query body text into the words, and to extract the feature words. To avoid the effects of noise, only Japanese nouns and adjectives are selected as feature words. For English, stop words [5] are also excluded from being used as feature words. The likeliness of feature words is computed from the MeCab (with the mecab-ipadic dictionary) morphological analysis results. The likeliness of a given word as a feature words is derived from the term occurrence cost score, which is pre-defined by the MeCab dictionary. The term occurrence cost is defined as a constant value that reflects the inverse term occurrence probabilities derived from the MeCab corpora.

Figure 2 shows screenshots of the Fuwatto CiNii Search top page. Fuwatto CiNii Search allows users to search a database using either a sentence query as input text or a webpage query to specify a query URL. It also provides links to national newspaper (i.e., Mainichi and Asahi) editorials as sample queries. Figure 3 shows the Fuwatto CiNii Search results page. The search results page lists the retrieved documents and shows links to corresponding CiNii Articles database items.



ふわっとCiNii関連検索
fuwat.to/cinii

ふわっと関連検索とは　更新履歴
最終更新日：2011年03月06日

ふわっとCiNii関連検索

文章から検索　ウェブページから検索

論文検索 □

入力したテキストまたはウェブページに関連した論文をCiNiiで検索します。長いテキストやURLで指定したページからでも関連キーワードを自動的に抜き出して論文検索できるのが特徴です。

例: 朝日新聞社説（元記事(asahi.com)）

検索例: 毎日新聞/社説・オピニオン

- ふわっと関連検索:近事片々:W杯初出場初ゴール (元記事)
- ふわっと関連検索:憂楽帳:札幌で会った少年 (元記事)
- ふわっと関連検索:W杯・2014ブラジル大会:サッカー　日本、初戦敗れる　コートジボワールに1－2　後半逆転許す (元記事)
- ふわっと関連検索:W杯・2014ブラジル大会:サッカー　日本、初戦敗れる　ザッケローニ監督の話 (元記事)
- ふわっと関連検索:東日本大震災:福島第1原発事故　避難指示区域、農地転用の要件緩和　政府検討、再生エネ整備促進 (元記事)

高久雅生 (Masao Takaku)
http://masao.jpn.org/, tmasao@acm.org
Powered by fuwatto.rb 2.3.3

(a) Text query

ふわっとCiNii関連検索
fuwat.to/cinii

ふわっと関連検索とは　更新履歴
最終更新日：2011年03月06日

ふわっとCiNii関連検索

文章から検索　ウェブページから検索

http://

論文検索 □

入力したテキストまたはウェブページに関連した論文をCiNiiで検索します。長いテキストやURLで指定したページからでも関連キーワードを自動的に抜き出して論文検索できるのが特徴です。

例: 朝日新聞社説（元記事(asahi.com)）

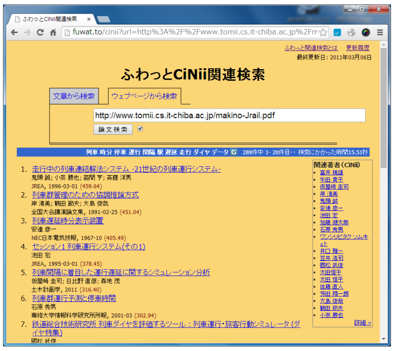
検索例: 毎日新聞/社説・オピニオン

- ふわっと関連検索:近事片々:W杯初出場初ゴール (元記事)
- ふわっと関連検索:憂楽帳:札幌で会った少年 (元記事)
- ふわっと関連検索:W杯・2014ブラジル大会:サッカー　日本、初戦敗れる　コートジボワールに1－2　後半逆転許す (元記事)
- ふわっと関連検索:W杯・2014ブラジル大会:サッカー　日本、初戦敗れる　ザッケローニ監督の話 (元記事)
- ふわっと関連検索:東日本大震災:福島第1原発事故　避難指示区域、農地転用の要件緩和　政府検討、再生エネ整備促進 (元記事)

高久雅生 (Masao Takaku)
http://masao.jpn.org/, tmasao@acm.org
Powered by fuwatto.rb 2.3.3

(b) URL query

Fig. 2. Screenshots of Fuwatto CiNii Search



ふわっとCiNii関連検索
fuwat.to/cinii?url=http%3A%2F%2Fwww.tomii.cs.it-chiba.ac.jp%2Fm

ふわっと関連検索とは　更新履歴
最終更新日：2011年03月06日

ふわっとCiNii関連検索

文章から検索　ウェブページから検索

http://www.tomii.cs.it-chiba.ac.jp/makino-Jrail.pdf

論文検索 ☑

列車 時分 停車 運行 間隔 駅 遅延 走行 ダイヤ データ Ci　289件中 1- 20件目・・ 検索にかかった時間15.51秒

1. 走行中の列車連結解法システム -21世紀の列車運行システム-
   鬼頭 誠; 小奈 勝也; 斎間 亨; 斉藤 洋男
   JREA, 1996-03-01 (459.64)
2. 列車群管理のための協調推論方式
   岸 清美; 鶴田 節夫; 大島 俊哉
   全国大会講演論文集, 1991-02-25 (451.04)
3. 列車遅延時分表示装置
   安達 彦一
   NEC日本電気技報, 1967-10 (405.49)
4. セッション1 列車運行システム(その1)
   池田 宏
   JREA, 1995-03-01 (378.45)
5. 列車間隔に着目した運行遅延に関するシミュレーション分析
   仮屋崎 圭司; 日比野 直彦; 森地 茂
   土木計画学, 2011 (316.40)
6. 列車群運行予測と停車時間
   石原 秀男
   専修大学情報科学研究所所報, 2001-03 (302.94)
7. 鉄道総合技術研究所 列車ダイヤを評価するツール：列車運行・旅客行動シミュレータ (ダイヤ特集)
   國松 武俊

関連著者(CiNii)
- 富井 規雄
- 牛田 貴平
- 仮屋崎 圭司
- 岸 清美
- 鬼頭 誠
- 安達 彦一
- 池田 宏
- 加藤 健太郎
- 石原 秀男
- ワンシリピタク ソムキェト
- 井口 雅一
- 笠井 浩司
- 國松 武俊
- 太田恒平
- 太田 恒平
- 佐藤 直人
- 芳田 翔一朗
- 大島 俊哉
- 鶴田 節夫
- 小奈 勝也

詳細 »

Fig. 3. Example search results

3 Experimental Evaluation

Here, we describe an experimental evaluation of our proposed methods. We evaluated retrieved results on the basis of effectiveness using content from newspaper articles on the web.

3.1 Topics

We used 34 newspaper articles for the evaluation. These articles were chosen from articles published by two major national newspapers, Sankei and Asahi newspapers). The newspaper articles were selected by random sampling from a population of articles published from March 19 to May 9, 2010.

For the evaluation, we used newspaper articles as the search topics. The articles were used as Fuwatto CiNii Search query documents, and the retrieved results were evaluated on the basis of their relevance to the original document. We also attempted several similarity scoring and search methods, and compared the obtained results according to topical relevance by manual assessment.

3.2 Relevance Assessments

We recruited one assessor for relevance assessment of the retrieved documents.

Relevance assessments were graded on three levels: A) relevant, B) partially relevant, and C) non relevant.

The documents pool for the assessment consisted of the top 50 search results retrieved from 34 newspaper article queries for several search methodologies described below. The assessor made relevance judgments on 22,005 documents, which on average, consisted of 647 documents per topic.

3.3 Experimental Runs

In the experimental evaluation, we implemented several versions of Fuwatto Search with modified feature words extraction and query execution methods, and then compared the results from these runs in terms of effectiveness. The following describe the methods used for feature word extraction and document ranking.

1. Feature word extraction and weighting
   - **TF**: Feature words are chosen on the basis of the term frequency.

     $$weight(t) = TF(t)$$

     Here, t is a given word, $TF(t)$ is the term frequency for word t in the original document, and $weight(t)$ is the likeliness score for a feature word.
   - **LogCost**: Feature words are chosen on the basis of the total score value of logarithm of occurrence cost value computed by the MeCab morphological analyzer.

     $$weight(t) = \sum_i log_2(Cost(t_i))$$

     Here, $Cost(t_i)$ is each term occurrence cost value.
   - **IDF**: The document frequency for each feature word is obtained by sending a given feature word as a keyword query to the target database. Feature words are chosen by multiplying the weighting score by the inverse document frequency in the target database.

     $$weight'(t) = \frac{weight(t)}{log_2(DF(t))}$$

     Here, $DF(t)$ is the document frequency of term t in the target database.
2. Keyword querying
   - **AND**: The top n feature words are selected in order of score, and these feature words are joined (by the AND operator) to construct a new query. The joined query is sent to the target database. If less than m documents are retrieved from the target database, the query is modified to use fewer feature words (e.g., $n-1$, $n-2$, $\ldots$,) until the number of retrieved documents is greater than m.

  - **Comb**: All combinations of three words from the top n feature words are extracted. These three words are then joined (AND operator) to generate query keywords. Then, all query keywords are sent to the target database.

3. Re-ranking
  - **Rerank**: The cosine similarity between a feature words vector and each retrieved document vector is computed, and all retrieved documents are re-ranked in the order of highest similarity.
  - **PRF(α)**: The top k ranked documents are retrieved and used to create a new words vector. This new vector is merged into the original feature words vector with a normalizing factor. All retrieved documents are re-ranked in the order of highest similarity with the modified words vector.

$$weight'(t) = (1 - \alpha)weight(t) + \alpha \cdot weight_d(t)$$

    Here, $weight_d(t)$ is the new vector by pseudo relevance feedback (PRF), and α is a constant merging factor.

We used the number of feature words $n = 10$ and the number of search results $m = 100$ as evaluation parameters. In addition, we used the number of feedback documents $k = 20$ as the PRF weight parameter.

3.4 Evaluation Metrics

We used precision at 10 (Prec@10) and mean average precision (MAP) as evaluation metrics for the retrieved documents. In the evaluation, we considered relevance levels A and B as relevant.

4 Results and Discussions

4.1 Term extraction, weighting and rankings

Table 1 shows the evaluation results for each method. The CiNii Articles is an academic database; therefore, we show evaluation results for newspaper articles on both the science section and other sections.

The querying and ranking methods based on Comb+Rerank demonstrated major improvements (0.11 over AND-based methods in terms of Prec@10). On an average, the Comb+Rerank method could retrieve at least one or extra relevant documents than the AND method. This seems to be due to the fact that the Comb method could obtain a greater number of relevant documents from the database, and Rerank could place relevant documents at higher rank using vector similarity. For the case of AND method, retrieval performance tends to rely heavily on the primary feature word vector. The Comb method seems to improve this drawback.

Among the methods used for weighting feature words, the IDF method demonstrated a slight performance improvement than over LogCost and TF; however, the differences among these methods were negligible.

On the other hand, the results for the PRF methods demonstrated weaker performance than the original Comb+Rerank methods.

Table 1. Evaluation results

Methods	Prec@10			MAP		
	Total	Science	Others	Total	Science	Others
AND+TF	0.079	0.111	0.040	0.064	0.056	0.074
AND+TF+IDF	0.094	0.079	0.113	0.058	0.020	0.110
AND+LogCost	0.106	0.116	0.093	0.060	0.050	0.081
AND+LogCost+IDF	0.077	0.074	0.080	0.058	0.020	0.105
AND+Rerank+TF	0.121	0.147	0.087	0.046	0.067	0.020
AND+Rerank+TF+IDF	0.138	0.142	0.133	0.046	0.043	0.050
AND+Rerank+LogCost	0.100	0.095	0.107	0.038	0.043	0.031
AND+Rerank+LogCost+IDF	0.132	0.158	0.100	0.044	0.046	0.041
Comb+Rerank+TF	0.218	0.258	0.167	0.167	0.207	0.117
Comb+Rerank+TF+IDF	0.232	0.274	0.180	0.167	0.215	0.106
Comb+Rerank+LogCost	0.232	0.253	0.207	0.168	0.201	0.126
Comb+Rerank+LogCost+IDF	0.250	0.279	0.213	0.170	0.214	0.114
PRF(.25)+Comb+Rerank+LogCost+IDF	0.235	0.263	0.200	0.032	0.041	0.090
PRF(.50)+Comb+Rerank+LogCost+IDF	0.197	0.226	0.160	0.103	0.132	0.067
PRF(.75)+Comb+Rerank+LogCost+IDF	0.106	0.126	0.080	0.053	0.071	0.030
PRF(1.0)+Comb+Rerank+LogCost+IDF	0.062	0.079	0.053	0.017	0.037	0.020
PRF(2.0)+Comb+Rerank+LogCost+IDF	0.006	0.011	0.000	0.005	0.008	0.001

4.2 Document Categories

Of the total 34 topic documents, 19 articles were from the science section, and 15 articles were from other sections. These target article categories can also affect retrieval performance. For nearly all Comb-based methods, retrieval performance for science articles was better than other articles because term usage in the source documents from the science section is similar to the target academic database, which seems to benefit document retrieval by feature words.

The varying categories of source documents may affect retrieval performance. This implies that it is possible to obtain better performance with adaptive approaches utilizing the similarity search methods that can switch according to the categories that the original document belongs.

4.3 Response Time

The proposed methodology is a mash-up that employs Web API protocols. For the implementing the Fuwatto Search, once a query (text or URL) is received, querying is performed over the Internet in real time. In this searching model, real time responses are also a key point. Depending on the querying methods, there are differences in the number of search queries issued and the extracted feature words. Notably, these differences may affect response time.

We counted the response time as the average from five trials per query topic. On average, the “AND+LogCost+IDF” method took 2.4 seconds for all topics, whereas the “Comb+Rerank+LogCost+IDF” method took 15.1 seconds. This difference in response time can be explained by the number of HTTP requests to the target database. The AND+X method sent 20 HTTP requests on average, whereas the Comb+X method sent 131 requests.

There is a tradeoff between response time and retrieval effectiveness. The Comb+X method outperformed the AND+X method in retrieval effectiveness, as is described in Section 4.1. However, the Comb method requires more HTTP accesses than the AND method. The Comb+X method took at most 30 seconds; therefore, it is inadequate when real-time response performance is important. Thus, the AND method is used when real-time responses are important, while the Comb method is used for indirect uses, such as for searching through the embedded content (e.g., Google AdSense).

5 Conclusion

This paper has proposed the Fuwatto Search method. We have demonstrated the effectiveness of the proposed method through experimental evaluation and implementation using CiNii Articles and its API. The evaluation results for Fuwatto CiNii Search with newspaper articles demonstrate performance of 0.25 for Prec@10, and 0.17 for MAP.

In future, we plan to improve our weighting and querying methods. In addition, we intend to evaluate the proposed method using other text categories, such as more general web content or other target databases.

Acknowledgment

This work was partially supported grants from the Japan Society for the Promotion of Science KAKENHI (No. 25730193 and No. 26330362).

References

1. Nakatani, S.: Body text extraction of web pages (in Japanese) http://labs.cybozu.co.jp/blog/nakatani/2007/09/web_1.html (updated 2007-09-12, accessed 2014-06-15).
2. National Institute of Informatics: CiNii Articles http://ci.nii.ac.jp/en, (accessed 2014-06-15).
3. National Institute of Informatics: Metadata and API: CiNii Articles OpenSearch for Articles http://ci.nii.ac.jp/info/en/api/a_opensearch.html, (accessed 2014-06-15).
4. Kudo, T.: MeCab: Yet another part-of-speech and morphological analyzer https://code.google.com/p/mecab/ (accessed 2014-06-15).
5. Library of Congress: InQuery stopword list for THOMAS http://thomas.loc.gov/home/stopwords.html, (accessed 2010-02-10).