RELATED WORD EXTRACTION FROM WIKIPEDIA FOR WEB RETRIEVAL ASSISTANCE

Kentaro Hori
Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka City, Fukuoka, Japan

Tetsuya Oishi, Tsunenori Mine, Ryuzo Hasegawa, Hiroshi Fujita, and Miyuki Koshimura
Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan

Keywords: Wikipedia, Web retrieval.

Abstract: This paper proposes a web retrieval system with extended queries generated from the contents of Wikipedia.By using the extended queries, we aim to assist user in retrieving Web pages and acquiring knowledge. To extract extended query items, we make much of hyperlinks in Wikipedia in addition to the related word extraction algorithm. We evaluated the system through experimental use of it by several examinees and the questionnaires to them. Experimental results show that our system works well for user's retrieval and knowledge acquisition.

1 INTRODUCTION

In recent years, many people can access Web very easily thanks to the vast spread of internet as well as the availability of convenient search engines.

For instance, Google[1], Yahoo[2], and Goo[3] are commonly used. Giving a few keywords, these systems retrieve such Web pages that users want to see from among the huge databases residing on the internet. Google, in particular, successfully presents us the most suitable pages on the first page of the retrieval results by applying the PageRank algorithm(Page and Lopes, 1998) which evaluates relevance of pages based on page links.

Even though, since the Web sources are so enormous and constantly increasing, it is often the case that we are not satisfied with the results given by them.

As for a method to improve the retrieval results, conjunctive query can be used. If you give multiple keywords all of which are relevant to your query, you will be able to obtain better retrieval results compared to those obtained for a single keyword. However, it may not always be the case that every keyword you give is appropriate for your intended query. Even if the keywords are all relevant to the query, you may not be always satisfied with the retrieved results. This is because search engines will look over such Web pages that are relevant to the query, but do not contain any keyword in the query. Also, if the area within which you want to look for some information is not very familiar to you, you will be unable to give appropriate keywords for your query. Thus, conjunctive query is not enough.

To solve the problem, there have been many works on such systems that can offer keywords relevant to one given by the user, such as (Masada et al., 2005)(Mano et al., 2003). In order to offer related keywords, these systems require some additional information other than the keyword given by the user. One method uses, as the additional information, the retrieved results themselves that are obtained for the initial query(Murata et al., 2008).

For instance, pseudofeedback methods take top ten Web pages in the ranking of the results as the relevant documents, and other Web pages as irrelevant documents. Then, some related words are extracted from the above classified documents. The advantage of the method is that it does not add users burden, and that it always returns some relevant results. However, these days, advanced services on the internet such as blogs, bulletin boards, and online shopping are becoming sources of tremendous information that

[1]http://www.google.co.jp/
[2]http://www.yahoo.co.jp/
[3]http://www.goo.ne.jp/

would appear in the retrieval results as irrelevant ones. If these are ranked higher in the result list, and used as the seemingly relevant documents for the pseudofeedback method, it would surely be the case that significant amount of irrelevant information is included in the final results.

Therefore, we present a new method and a system for related word extraction that uses Wikipedia[4] as the information source. Wikipedia is a well known online encyclopedia that is well organized with rich contents, words, and internal links. Moreover, since it can easily be updated by anyone, many researchers have been giving attention to it. Using Wikipedia, systems will be able to remove irrelevant information from their retrieval results, and improve the accuracy of the results especially when the area of interest is unfamiliar to the user.

The problem with Wikipedia is as follows: it may occur with high probability that the keyword given by the user is absent in it, since the total amount of information contained in Wikipedia is quite small compared to those contained in whole Web pages on the internet, and the users query may be exotic to Wikipedia. So, we categorize queries in advance, and build a most useful system for a user who wants to use the Web as if it were a large virtual dictionary. The usefulness is evaluated on the basis of accuracy improvement and the quantity of information that is new and interesting to the user.

2 RELATED WORK

Bedsides pseudofeedback, there are two more feedback methods called explicit feedback and implict feedback. The former depends on the users evaluation of documents, whereas the latter automatically collects documents by analyzing users operations such as scroll, click and zoom.

Another system uses personal information that reflects a user profile as an auxiliary information(Sieg et al., 2007)(Yoshinori, 2004)(Qiu and Cho, 2006)D Concretely a user profile will be derived from schedules or some database containing his/her interests or favorites. These help the system to offer related words that meet his/her intention. For instance, when a user wants to know about the weather forecast, the system would examine just the regions which are near the place where he/she lives. As far as the user wants such information that is very specific to his/her interest, it should be more appropriate to derive related words from user profiles than to find out them from among the Web that includes words from so many fields of general interest. Yet these system would have some difficulty to show relevant words when the user really wants to obtain information quite new to him. This is because there should be few information in the user files which are supposed to suggest words about alien culture, unseen incidents, and unfamiliar history, etc.

As for work concerning Wikipedia, Nakayama(Ito et al., 2008) et.al. have succeeded in constructing the association thesaurus dictionary for extracting related words from a given query. They calculate co-occurrence of words that have links to other pages in terms of the relatedness between those linked pages. However, as Wikipedia's internal links are provided only arbitrarily by the author of that specific page, some pages contain many links, others very few, or even no internal link at all. In such a case, their system will not work. On the other hand, our system uses the related word extraction algorithm of our own where all words, with or without linked words, are taken into account, and free from the above problem. The detail of the algorithm is described in section 5.

[4]http://ja.wikipedia.org/

3 SYSTEM OVERVIEW

Figure 1 shows an overview of our system.

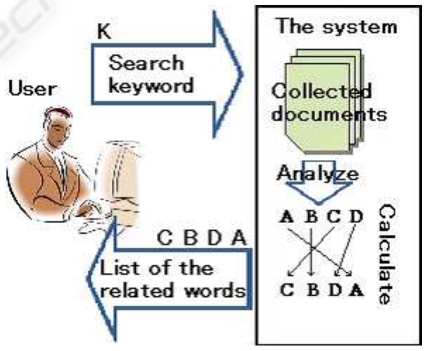


Figure 1: Image of system.

First, the user inputs a query, refereed as the *initial query*, to the system. The goal of our system is to find out related words for the query.

Second, the system collects documents called *relevant documents* that are related to the initial query and are used as sources for extracting related words. Our system takes Wikipedia as a source of relevant documents and extract some paragraphs from Wikipedia pages that are related to the initial query. The details will be described in section 4.

Third, the system performs morphological analysis on the above collected relevant documents by means of MeCab[5]. With MeCab it squeezes only nouns from the documents, then discards words like "something" or "anything" that are registered in the system as unnecessary in advance. Furthermore, a special rule is installed to it so that a connected word like "graduation thesis" will not be broken apart into two nouns "graduation" and "thesis."

Finally, the system calculates for each of the above obtained words an estimate value as the related word. There have been many works on the way to do this. For instance, RSV (Robertson Selection Value)(Robertson, 1990) algorithm considers such words as more important that appear more often (or less often) in relevant documents than in irrelevant documents. RSV is not very suitable for our system, since there is basically no way of providing irrelevant documents. Therefore, we use a related word extraction algorithm(Kuramoto et al., 2007) of our own that calculates the value of a word based on the notion of distance between words in a document. We also make some correction of the value of words on the basis of internal links within Wikipedia. The details are described in section 5.

4 OUR NEW METHOD

We use Wikipedia as the relevant document for our system. The reason why we chose Wikipedia is as follows.

- Wikipedia is one of the largest encyclopedias on the Web having more than 590,000 Japanese words (as of June, 2009).
- Its content is very rich[6](Thomas, 2006).
- A number of people can easily access, even modify its content and make it always up-to-date.

4.1 Extraction Method

The way of collecting relevant sentences slightly differs according to the number of keywords in the initial query.

Initial Query with One Keyword

Let the initial query be "A". The system retrieves Wikipedia to find an article on "A". It extracts the highest ranked page of the retrieved result and searches for the paragraph where "A" appears in the page. The extracted paragraph is used as a relevant sentence.

[5]http://mecab.sourceforge.net/

[6]http://woy2007.sbcr.jp/

Initial Query with Two or More Keywords

Use the following elements.

- $Q_i (i = 1, \cdots, n)$:Given query
- $P_{Q_i} (i = 1, \cdots, n)$:Highest rank of result page of Wikipedia retrieved by Q_i
- $Extract(P, Q)$:Extract from "P" the paragraph where "Q" appears.
- $ExSentence$:Extraction result

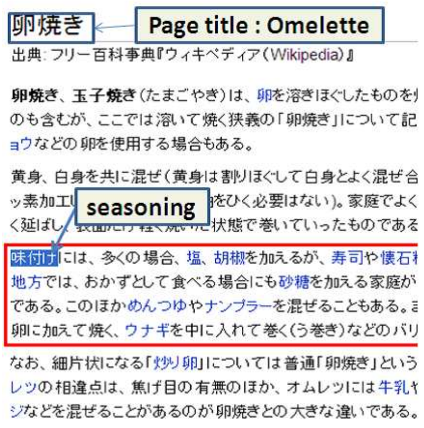
Extracted sentence are expressed by the following expressions.

$$ExSentence = \sum_{k=1}^{n} \sum_{j=1}^{n} (j \neq k) Extract(P_{Q_j}, Q_k)$$
$$+Extract(P_{Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n}, Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n)$$

Let the initial query be "Q_1 Q_2". The system retrieves Wikipedia with three queries "Q_1", "Q_2", and "Q_1 Q_2". It collects the highest ranked page for each query. Let us call them "P_{Q_1}", "P_{Q_2}", and "$P_{Q_1 Q_2}$", respectively. From "P_{Q_1}" the paragraph where "Q_2" appears is extracted. From "P_{Q_2}" the one where "Q_1" appears is extracted. From "$P_{Q_1\ Q_2}$" the one where both "Q_1" and "Q_2" appear is extracted.

For instance, when the system retrieves with query "Omelette seasoning", "Omelette" is first retrieved with Wikipedia. Since the page of "Omelette" becomes a hit, the system searches for a paragraph with the word "Seasoning" in "Omelette" page. When it is found, the paragraph with "Seasoning" is extracted as a relevant sentence. Next, when the system retrieves with "Seasoning", the page of "Seasoned laver" becomes a hit. However, nothing is extracted because there is no paragraph where word "Omelette" exists. Finally, it retrieves by "Omelette seasoning", and the page of "Omelet" becomes a hit. The paragraph where both "Omelette" and "Seasoning" words exist in this page is extracted. In this way, the extracted paragraphs are assumed to be relevant sentences, and the morphological analysis is performed. Figure 2 shows how the sentences containing the word "seasoning" are searched for, and the corresponding paragraph is extracted.

Although for initial queries with three or more keywords, a paragraph can be extracted as a relevant sentence as well, we did not experiment. This is because such a query is issued with a clear purpose so that we can get enough results, and the extraction time increases greatly by the combination.



卵焼き
出典: フリー百科事典『ウィキペディア（Wikipedia）』

卵焼き、玉子焼き（たまごやき）は、卵を溶きほぐしたものを
のも含むが、ここでは溶いて焼く狭義の「卵焼き」について記
ョウなどの卵を使用する場合もある。

黄身、白身を共に混ぜ（黄身は割りほぐして白身とよく混ぜ合
ッ素加工 をひく必要はない）。家庭でよく
く延ばし 状態で巻いていったものである

味付けには、多くの場合、塩、胡椒を加えるが、寿司や懐石
地方では、おかずとして食べる場合にも砂糖を加える家庭が
である。このほかめんつゆやナンプラーを混ぜることもある。
卵に加えて焼く、ウナギを中に入れて巻く（う巻き）などのバリ

なお、細片状になる「炒り卵」については普通「卵焼き」という
レッの相違点は、焦げ目の有無のほか、オムレツには牛乳
ジなどを混ぜることがあるのが卵焼きとの大きな違いである。

Figure 2: Actual extraction image.

5 RELATED LEVEL CALCULATION OF WORD

5.1 Outline of the Algorithm

The algorithm extracts, from words appearing in the text T, a group of words that are related to a given set of keywords K, and thus are expected to be important for K. Based on the distance from a keyword, we evaluate each word in K or T, and form a set of words to be output as a result of related word extraction.

The idea behind this algorithm is a distance between words. Concretely, it pays attention to the order of words that appear in sentences, and it considers the words that appears near to A to be relevant to A.

5.1.1 Notation

We define some symbols as follows:

- K: a set of keywords that give a basis for extracting related words.
- T: a text to be given degrees of relation with K.
- $k_i(i = 1,\cdots,m)$: m keywords appearing in K, where the keywords $k_1, k_2, \cdots, k_m$ appear in this order.
- $s_h(h = 1,\cdots,n)$: n sentences appearing in T, where the sentences $s_1, s_2, \cdots, s_n$ appear in this order.
- $F_{k_i}(s_g)(g = 1,\cdots,n)$: $F_{k_i}(s_g) = 1$when $s_g$contains key word k_i, otherwise $F_{k_i}(s_g) = 0$.
- $t_j(j = 1,\cdots,o)$: o words appearing in T, where the words $t_1, t_2, \cdots, t_n$ appear in this order.

t_j represents a noun word extracted by performing morphological analysis of words in the text T. The extracted words are arranged in appearance order. In case a word appears more than twice, these occurrences are considered to be different.

5.2 Evaluation of Word in Text

Evaluation of words in the text T is performed as follows:

1. Calculate basic value $BV(s_h)$ of $s_h(j = 1,\cdots,n)$ with $k_i(i = 1,\cdots,m)$ as a criterion.
2. Smooth $BV(s_h)$.
3. Calculate a final value $V(t_j)$ using word frequencies.

5.2.1 Calculating $BV(s_h)$

This algorithm evaluates the word based on key word group K focusing on the distance between words that appear in K and text T, and first calculates the evaluation value that gives the basis.

We define $BV_{k_i}(s_h)$ as the score of s_h with respect to k_i.

$$BV_{k_i}(s_h) = \sum_{g=1}^{n} (n - |g - h|) F_{k_i}(s_g) \qquad (1)$$

Here, $|g - h|$ is a distance of s_h and a sentence in which k_i appears. When k_p appears in s_q, $|g - h| = |q - q|$ and $F_{k_p}(s_q) = 1$, so $BV_{k_p}(s_q) = n$. For the next sentences s_{q+1} and s_{q-1}, the distances are $|q - (q+1)| = 1, |q - (q-1)| = 1$, respectively, and thus $BV_{k_p}(s_{q+1}) = BV_{k_p}(s_{q-1}) = n - 1$. Note that $BV_{k_i}(s_h)$ takes a larger value, as s_h gets closer to the sentence in which k_i appears.

After having calculated every $BV_{k_i}(s_h)$ $(i = 1,\cdots,m)$, the algorithm calculates $BV(s_h)$ with the following formula.

$$BV(s_h) = \sum_{i=1}^{m} BV_{k_i}(s_h) \qquad (2)$$

An example that calculates $BV(s_h)$ is shown in Table 1, where keyword A appears twice in text T.

Table 1: An example of scoring sentences using distance.

keyword K	A	B			
text T	AFB	ED	AFC	FE	DE
$BV_A(s_h)$	8	8	8	6	4
$BV_B(s_h)$	5	4	3	2	1
$BV(s_h)$	13	12	11	8	5

5.2.2 Smoothing of $BV(s_h)$

The above way of determining $BV(s_h)$ is unfair when considering the position of sentence s_h in T. For $s_h(h = 1,n)$ appearing at the edge of T, the value of $BV_{k_i}(s_h)$ ($j = 1$ or n) ranges from 1 to n ($1 \leq BV_{k_i}(s_h) \leq n$), while for the word appearing in the center of T, it ranges from $n/2$ to n ($n/2 \leq BV_{k_i}(s_h) \leq n$).

Thus, the expected values of $BV(s_h)$ differ depending on the position h at which s_h occurs. As the word appearing in the center of T will be given a larger value, a fair evaluation cannot be achieved.

To remedy this problem, we smooth the obtained $BV(s_h)$ using $EBV(h)$, the expected evaluation value of s_h at position h. $EBV(h)$ is calculated as follows:

$$EBV(h) = \frac{1}{2n} n(n+2h-1) - 2h(h-1) \qquad (3)$$

where n is the total number of sentence occurrences in T. Let $EBV(s_h)$ be the evaluation value after smoothing. It is calculated using the following formula.

$$EBV(s_h) = \frac{BV(s_h)}{EBV(h)} \qquad (4)$$

Table 2 shows a process of getting $EBV(s_h)$ from the given text T. Since $EBV(h)$ is the expected evaluation value at position h, the value for the position nearer the center (around $h = n/2$) becomes larger. Naturally, both left and right sides are symmetrical with respect to the center having a peak. As seen in Table 2, by smoothing of $BV(s_h)$ with $EBV(h)$, unfair evaluation in $BV(s_h)$ is well remedied.

Table 2: An example of calculating $EBV(s_h)$ using the expected value $EBV(h)$ at occurrence position h.

keyword K	A	B			
text T	AFB	ED	AFC	FE	DE
$BV(s_h)$	13	12	11	8	5
$EBV(h)$	3	3.6	3.8	3.6	3
$EBV(s_h)$	4.33	3.33	2.89	2.22	1.67

5.2.3 Calculating $V_T(t_j)$

In addition to the evaluation based on the distance between sentences, we take into consideration the concept of *Term Frequency* that is often used in the TF/IDF method. In other words, the words which appear larger number of times in a text T should be more important.

First, we compute the average $AveEBV(t_j)$ of the expected evaluation values $EBV(t_j)$, for word t_j that appears several times.

For instance, if sentence s_a and sentence $s_b(ab)$ are identical with t_c, $AveEBV(t_c) = (EBV(s_a) + EBV(s_b))/2$.

Of course, for the word t_j (t_j is a word in s_h) that appears only once in T, $AveEBV(t_j) = EBV(s_h)$.

Moreover, we compute the weight $W_T(t_j)$ using the tf value of word t_j as follows.

$$W_T(t_j) = 1 + \frac{tf(t_j)}{n} \log tf(t_j) \qquad (5)$$

Here, $tf(t_j)$ is the number of occurrences of word t_j in the text T and n is the total number of occurrences of words in T. Then by using $AveEBV(t_j)$ and $W_T(t_j)$, we calculate the evaluation value $V_T(t_j)$ of t_j in T as follows.

$$V_T(t_j) = AveEBV(t_j) * W_T(t_j) \qquad (6)$$

We assume that $V_T(t_j)$ is the evaluation value of word t_j in text T in this algorithm. Table 3 shows a process of calculating $V_T(t_j)$. Because word C appears twice in text T, $tf(C)$is 2. As for other words, the tf value is 1. Since $W_T(t_j)$ is calculated using tf values, the tf value is 1 for words which appear once, and takes a value greater than 1 for words which appear twice or more.

Then the final evaluation value $EBV(C)$ is calculated. We assume that the words in T are related to K in the order of F, A, B, E, D and C.

Table 3: $V_T(t_j)$ of each words.

keyword$K$	A	B			
text$T$	AFB	ED	AFC	FE	DE
$EBV(s_h)$	4.33	1.67	2.11	2.22	2.67

word	A	B	C	D	E	F
$AveEBV(t_j)$	3.61	4.33	2.11	1.67	2.50	2.56
$tf(t_j)$	2	1	1	2	3	3
$W_T(t_j)$	1.28	1	1	1.28	1.66	1.66
$V_T(t_j)$	4.62	4.33	2.11	3.20	4.00	5.23

5.3 Correction by Internal Links of Wikipedia

5.3.1 Outline

In addition, we pay attention to the hyperlink(internal link) to another article in the article on Wikipedia. This internal link can be arbitrarily set by the author of the article, and the word to which an internal link is set can be considered to be deeply related to the article, and thus an important word that you should pay attention to. However, since an internal link can be set arbitrarily it may not be related to the initial query. Then, we calculate a degree of relevancy to the initial query.

5.3.2 Computational Method

- LW_f $(f = 1cp)$: a word for which an internal link is specified
- $RWS(LW_f)$: the evaluation value of LW_f calculated by the related word extraction algorithm
- $LW_{f(k_i)}$: frequency that keyword k_i appears in the article to which the internal link of LW_f points.

A final evaluation value is calculated from the above-mentioned three items as follows.

- $WikiEX(LW_f) = \log(\Sigma_{i=1}^{m} LW_{f(k_i)}) + 1$
- $WordScore(LW_f) = RWS(LW_f) * WikiEX(LW_f)$

Since preliminary experiments show that $LW_f(k_i)$ ranges from 0 to 100 or more, we took the logarithm to reduce an extreme evaluation value difference. We take this $WordScore(LW_f)$ to be a final evaluation value. For a word without having an internal link, its evaluation value of the related word extraction algorithm is assumed to be a final evaluation value.

6 CLASSIFICATION OF QUERY

We classify queries into the following three classes on the basis of investigation by Andrei(Broder, 2002).

1. Navigational query
2. Transactional query
3. Informational query

A navigational query is the one for which just a single page is to be sought after. For instance, by giving “Google” as a query, most probably the user would like to see the Google’s top page. This type of queries need not to be augmented, as any existing search engine can easily present satisfactory results for them.

A transactional query is the one at which the user would start taking some action such as shopping, downloading, and looking for maps. For this type of queries, it should be effective to perform query augmentation based on user profiles as described in section 2.

An informational query is the one about which the user wants to obtain some knowledge. When the user has a word unfamiliar to him/her and wants to know about something concerning it, he/she would make a query about the word to obtain its related information.

Actually, informational queries are the principal subject of our system that are supposed to be treated most effectively. Therefore, we concentrate on this class of queries for our subject in the following experiments.

About query classification, we plan to use the automatic classification technology like (Fujii, 2007) in the future, but we manually classified the queries this time.

7 EXPERIMENTS

We evaluate our system by comparing its results to those obtained in [Goo] and [Web5]. [Goo] is a search engine operated by NTT Resonant. We compare the accuracy of results by our system to those obtained by [Goo]. [Web5] is a data obtained for the Web pages which are ranked in top five of the retrieval results for an initial query and taken as relevant documents(Oishi et al., 2008). We examine the accuracy of results in terms of the difference of relevant documents, between those of [Web5] and Wikipedia pages used in our system. The effectiveness of the related word extraction algorithm itself has been demonstrated by Oishi et.al.(Oishi et al., 2008) The accuracy of the results is measured in terms of MAP (Mean Average Precision). AP (Average Precision) is the average of ratios of the number of documents that user judges good to the number of whole documents provided as the result for the initial query. MAP is an average of AP over a set of queries.

The AP is defined as follows:

$$\text{Average Precision} = \frac{\sum_{r=1}^{L} I(r)P(r)}{R}$$

Where R is the total number of relevant documents; L is the number of results retrieved by a system. In this experiment, we set L to 10 and R to 10, too; $I(r)$ is 1 if the r-th ranked document is relevant, and 0 otherwise; $P(r)$ is $\frac{count(r)}{r}$; $count(r)$ is the number of relevant documents among r documents returned by the system.

Figure 3 shows the result.

80 queries are used for this experiment. We concentrate on how much the accuracy of the retrieval results is improved compared to those obtained by an existing engine. First, we calculate AP for the results given by Goo. Next, each query is classified into 11 classes according to the value of AP, first 0, second 0.1 or less, and so on, and finally 1.0 or less. Then, we calculate MAP for each class of queries and for each method being compared. We can say that a MAP value given by any method for those queries

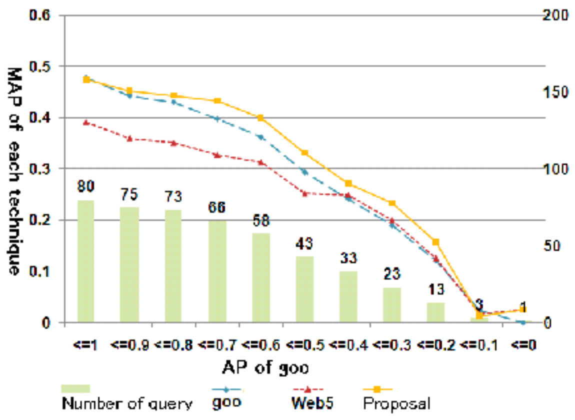


Figure 3: MAP for each AP of Goo.

having AP of less than 0.2 represents the accuracy of the method when Goo gives poor results, and that any MAP value for those queries having AP of less than 1.0 represents the overall accuracy of each method.

When Goo's AP is high, or the initial retrieval result is satisfactory for a user, the accuracy will hardly be improved. This is because adding a word to a very good query would not change the initial result, or even worse, degrade the result. As Goo's AP decreases, the accuracy of our method increases and becomes higher than that of Goo. This demonstrates that the query augmented by our method can reflect more clearly the users intention of the retrieval than the initial query does. The more accurate query is issued, the more unnecessary information is excluded from the result. Note also that many good related words can be extracted for informational queries, as a result of the fact that Wikipedia is a kind of encyclopedia. On the other hand, in another experiment where queries are not classified, our method shows poorer results. Although our method shows poor accuracy also for the data where Goo's AP is less than 0.1, we think it is just deviation due to too small number of the test set.

8 DISCUSSION

A function similar to our related word presentation is provided by Google. This is based on the retrieval records of all users in the world. We compare the related words given by Goole and those obtained by our method.

Table 4 shows the results for the query "iPS cell" for which a Japanese researcher won the Lasker prize and was much taken about. Google shows up words like Kyoto University and Yamanaka, who is the winner of the Novel prize, as the related words for the query word "iPS cell". These words are just temporary. On the other hand, our method give words that are more technical and relevant to the essential meaning of the query word. As for another example, table 5 shows the results for the query "NP complete problem", which is well known to people from the engineering field whereas unfamiliar to ordinal people. Google shows up just a single word as its related word. This is because few people give the query to Google, and the retrieval records are too small to extract related words. On the other hand, our method can provide many related words since Wikipedia has a good text concerning the NP complete problems.

Table 4: query:iPS cell.

Google	Proposal
Problem	Gene
Patent	Establishment
wiki	Group
Kyoto University	Embryonic stem cell
Regenerative medicine	Mouse
Nobel prize	Success
Yamanaka	Possible
Bayer	Human
Embryonic stem cell	introduce

Table 5: query:NP complete problem.

Google	Proposal
Example	Class NP
	Polynomial
	NP difficult
	possible to return
	Satisfiability problem
	possible to reduce
	Discovery
	Theorem
	Cook

9 CONCLUSIONS

Using Wikipedia as a source of relevant documents, better words can be extracted compared to those obtained by using retrieval results of conventional search engines like Goo. The results of our system would be much better from the viewpoint of user in acquiring knowledge. Also, compared to the related words recommendation by Google, better words can be extracted especially when the initial keyword itself is seldom used for a query, or it is just temporarily used in a specific topic. This is because Google depends on the record obtained from a large number of retrieval results for anonymous people, whereas ours are based on information that is more specific to the user.

This time, the extraction of the related word was limited only to Japanese because we had experimented by using goo, Mecab, and Japanese version Wikipedia. In order to implement it with other languages, it is necessary to use Google as a search engine, and also a morphological analysis tool and Wikipedia for the other language being targeted. Moreover, the algorithm used is designed based on the feature of a Japanese syntax. So, when applying to other languages, we need the adjustment of the parameters or the other algorithms such as Strube(Strube.M and Ponzetto.S, 2006).

We are going to implement a user friendly interface such that the user can retrieve by choosing the related words offered by the system, and collect evaluation results.

In future research, we are going to give attention to another characteristic of Wikipedia that it tends to drive users to other area of interest as he/she goes through the text in it. This will open up a new application of the related word extraction method to some kind of knowledge discovery.

ACKNOWLEDGEMENTS

This work was supported by KAKENHI(21500102).

REFERENCES

Broder, A. (2002). A taxonomy of web search. In *SIGIR Forum Vol.36,No.2*. SIGIR.

Fujii, A. (2007). Modeling anchor text and classifying queries in web retrieval. In *internet conference2007*. in Japanese.

Ito, M., Nakayama, K., Hara, T., and Nishio, S. (2008). A consideration of association thesaurus construction based on link co-occurrence analysis considering sentences. In *DEWS2008*. IEICE. in Japanese.

Kuramoto, S., Hasegawa, R., Fujita, H., Koshimura, M., and Mine, T. (2007). A method for query expansion using the related word extraction algorithm. In *JAWS2007*. SIG-ICS. in Japanese.

Mano, H., Itoh, H., and Ogawa, Y. (2003). Ranking retrieval in document retrieval. In *Ricoh Technical Report No.29*. Ricoh. in Japanese.

Masada, T., Kanazawa, T., Takasu, A., and Adachi, J. (2005). Improving web search by query expansion with a small number of terms. In *NTCIR-5 Workshop Meeting*. NTCIR. in Japanese.

Murata, N., Toda, H., Matsuura, Y., and Kataoka, R. (2008). A query expansion method using access concentration sites in search resul. In *DBSJ Letters Vol.6, No.4, pp.45-48*. in Japanese.

Oishi, T., Kuramoto, S., Mine, T., Hasegawa, R., Fujita, H., and Koshimura, M. (2008). A method of query generation using the related word extraction algorithm. In *IPSJ SIG Notes 2008(56) pp.33-40*. IPSJ. in Japanese.

Page, L. and Lopes, J. (1998). The pagerank citation ranking: Bringing order to the web. In *Technical Report*. Stanford University.

Qiu, F. and Cho, J. (2006). Automatic identification of user interest for personalized search. In *International World Wide Web Conference*. ACM.

Robertson, S. (1990). On term selection for query expansion. In *Journal of Documentation, 46, 4, pp. 359-364*.

Sieg, A., Mobasher, B., and Burke., R. (2007). Web search personalization with onotological user profiles. In *In Proc. of CIKM '07*. CIKM.

Strube.M and Ponzetto.S (2006). Wikirelate! computing semantic relatedness using wikipedia. In *AAAI pp.1419-1424*.

Thomas, C. (2006). An empirical examination of wikipedia's credibility. In *First Monday*.

Yoshinori, H. (2004). User profiling technique for information recommendation and information filtering. In *Journal of Japanese Society for Artificial Intelligence 19(3) pp.365-372*. JSAI. in Japanese.