A Product Retrieval System Robust to Subjective Queries

Kenji Sugiki
Graduate School of Information Science
Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601, Japan
Email: sugiki@el.itc.nagoya-u.ac.jp

Shigeki Matsubara
Information Technology Center
Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601, Japan
Email: matubara@nagoya-u.jp

***Abstract*—In recent years, electronic markets are increasing rapidly and attracting the attention of customers. In these sites, people search for products using retrieval systems. They, however, often cannot translate their subjective needs into keyword-based queries or adapt to the interfaces. In this paper, we describe a product retrieval system robust to subjective queries. Using a large amount of consumer reviews, the system allows users to input natural language queries and retrieves appropriate products even if the queries are highly subjective. To estimate the correspondence between a query and a review text, the system extracts 3-tuples consisting of a product name/category, its features, and the value from each text using rules based on syntactic patterns. It calculates each product scores based on correspondence rate of 3-tuples and presents ranked relevant products. In experimental results for a accommodation domain, it obtained higher average and total precision for 10 queries compared with a baseline that uses keyword based tf-idf method. Thus, we confirmed the effectiveness for subjective queries.**

I. Introduction

In recent years, such electronic markets as Amazon and eBay have grown rapidly due to increases of Internet users. These sites deal in vast amounts of products and need to provide useful search environments for their users. Most sites, however, just search for prefixed data items such as product names, categories, and features. Since user needs may be extremely variable and highly subjective, they often cannot translate their needs into objective queries. Therefore the system can't easily to accommodate their needs.

On the other hand, there has been a study on a natural language interface for a product retrieval system in which users describe their needs with natural language [1] in an accommodation domain. In addition, recommendation systems using a dialogical approach have been proposed [2], [3]. However, these methods have problems transforming natural language sentences into database query language expressions (e.g., *SQL*) due to the subjectivity and the variety of natural language expressions.

In this paper, we propose a product retrieval system robust to subjective queries written in natural language. We use product reviews to match the natural language queries and products. Recently, electronic market sites generically offer review services. Additionally, product reviews are often added by customers in blogs. When customers deciding to purchase a product, customers often use the reviews. Therefore, reviews are also useful for product retrieval tasks.

We developed a product retrieval system for Japanese accommodation domain. Using large amounts of reviews written by bloggers, the system can respond to subjective queries written in Japanese such as "*choshoku-ga baikingu-no yado* (*I am looking for a hotel with an all-you-can breakfast.*)". In retrieval experimental results using about 220,000 reviews written in Japanese, we confirmed the effectiveness of our system.

This paper is organized as follows: Section 2 briefly gives on overview of our system. Section 3 describes the extraction process, and Section 4 describes the retrieval process in detail. Section 5 reports our experiments and results.

II. Product Retrieval System Using Reviews

Our system allows users to input natural language queries that may be highly subjective. When inputting queries, the system retrieves products matching the query and presents them in order of relevance scores. To match queries, we use reviews about products or services. For example, as shown in Figure 1, if the query is about *plasma TV* and a review is written about *plasma TV* "*TH-42PX500*", "*TH-42PX500*" matches the query.

To determine the degree of correspondence between a query and reviews, the system transforms these contents into semantic representations, and then computes their *correspondence rate*. In this study, we put the information about the query or the review text into 3-tuples "(*object*, *item*, *value*)". When users search for the product, they imagine three components: a category, features (*items*), and values. For example, if there exists a review that states "*gashitsu-ga kire* (*picture quality is clear*)" about *plasma TV* "*TH-42PX500*", the information of the review is represented as a 3-tuple "(*TH-42PX500*, *gashitsu* (*picture quality*), *kire* (*clear*))". This 3-tuple representation is identical as that used in related works about opinion extraction tasks [4], [5], [6], [7], [8], [9]. Many of these studies used phrasal or syntactic patterns to extract each term or the relation between the *item* (or product name) and its *value*.

These works, however, extracted (subjective) opinion information by only extracting adjective, adverb, and verb as opinion representations (in our study, *value*) or constructing

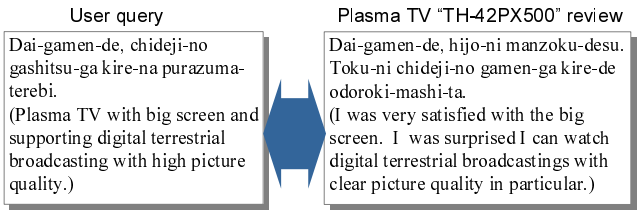

Fig. 1. Example of review matching query

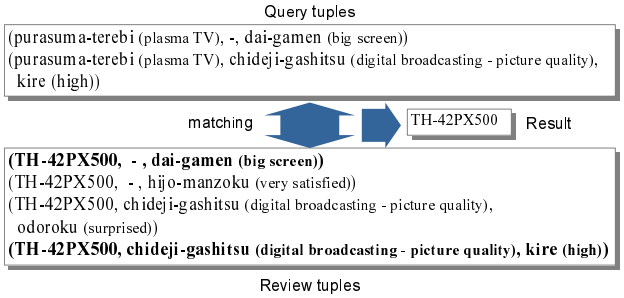

Fig. 2. Example of matching between query and review tuples

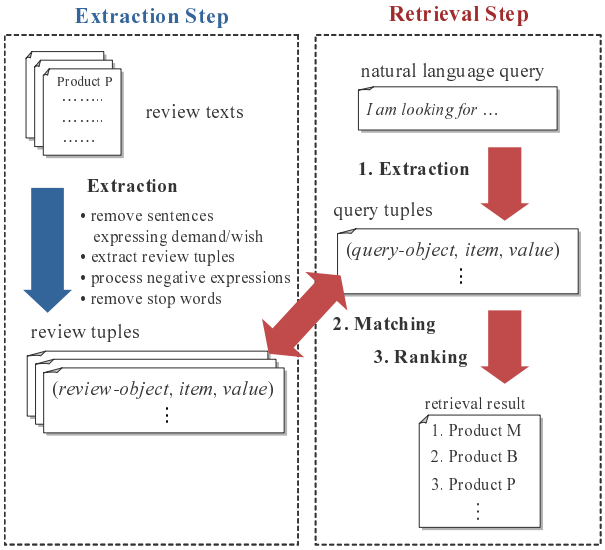

Fig. 3. Overview of our product retrieval system

opinion representation dictionaries. On the contrary, our system extracts objective information including facts as well as opinions. Therefore, since our study does not specialize in opinion representation, it extracts adjectives, verbs, and nouns as well as adjectives. Here, we determine the 3-tuple of the review as "(*review-object*, *item*, *value*)" and call it a *review tuple*. For example, if a review is writes that "*gashitsu-ga kire* (*image quality is high*)" about *plasma TV* "*TH-42PX500*", the system extracts review tuple "(*TH-42PX500*, *gashitsu* (*image quality*), *kire* (*high*))".

Additionally, the system extracts the 3-tuples from the query as well as the review. Here, we define the tuple of the query as "(*query-object*, *item*, *value*)" and call it a *query tuple*. For example, if the user inputs "*gashitsu-ga kire-na purasuma-terebi* (*Plasma TV with high picture quality*)", the system extracts query tuple "(*purazuma-terebi* (*plasma TV*), *gashitsu* (*picture quality*), *kire* (*high*))". We can judge that "*TH-42PX500*", which is included in the product category of *plasma TV*, matches the query. Figure 2 shows matching between the tuples extracted from the query and the review shown in Figure 1.

A system overview is given in Figure 3. The system consists of the following two components: (1) extracting review tuples from the review text; and (2) retrieving products that match the query. In (1) extraction part, the system extracts review tuples by applying transformation rules based on the syntactic pattern of modification relations between *bunsetsus* [1]. In (2) retrieval part, the system transforms a query into query tuples, calculates the *correspondence rate* between the query and review tuples, and then presents products matching the query.

[1] A *bunsetsu* is a Japanese phrasal unit that consists of one or more adjacent content words followed by any number of function words.

III. Extraction of Opinion Tuples

In this section, we describe a method to extract the review tuple "(*review-object*, *item*, *value*)" from the review. We assume that each review is written about a certain product. Hence, we decide that the system extracts *item-value pairs* in review tuples from the review and determines the products described by consumers as *review objects*. To extract *item-value pairs*, we focused on modification relation between *bunsetsus*.

We consider the following types of *item-value pairs* that appear in review texts:

1) subject-predicate relation
   "*Heya-wa kire-deshi-ta.* (*The room was clean.*)"
   ⇒ (*Hotel A*, *heya* (*room*), *kire* (*clean*))

   "*Choshoku-ga insho-teki-deshi-ta.* (*The breakfast was impressive.*)"
   ⇒ (*Hotel B*, *choshoku* (*breakfast*), *insho-teki* (*impressive*))
2) modifier-head relation
   "*Totemo shinsetsu-na hoteru-jugyoin-deshi-ta.* (*She was very friendly hotel staff.*)"
   ⇒ (*Hotel C*, *hoteru-jugyoin* (*hotel staff*), *totemo shinsetsu* (*very friendly*))

Hence, we used the patterns based on modification relations between *bunsetsus*.

To create rules to transform the review into review tuples, we investigated the relation between the appearances of *items* and *values* in review texts and the appearance patterns of modification relations between *bunsetsus*.

We confirmed the following frequent patterns: [2]:

(1) $heya\text{-}ga_X$ (*The room*) → $kire_Y$ (*was clean.*)

[2] X,Y are *bunsetsus* that include *item* or *value*, and "→" shows the modification relation between *bunsetsus*.

(2) *heya-ga*$_X$ (*The room*) → *kire-de*$_{Y_1}$ (*was clean*) → *kaiteki-deshi-ta*$_{Y_2}$ (*, and very comfortable*)

(3) *kire-na*$_Y$ (*The nice-looking*) → *heya-desu*$_X$ (*room*)

The following review tuples were extracted from the above three patterns, respectively:

(1) (*Hotel A*, *heya*$_X$ (*room*), *kire*$_Y$ (*clean*))

(2) (*Hotel A*, *heya*$_X$ (*room*), *kire*$_{Y_1}$ (*clean*)), (*Hotel A*, *heya*$_X$ (*room*), *kaiteki*$_{Y_2}$ (*comfortable*))

(3) (*Hotel A*, *heya*$_X$ (*room*), *kire*$_Y$ (*nice-looking*))

Therefore, we created the following three rules. Here, O is the review-object.

(1) $X \rightarrow Y \quad \Rightarrow (O,X,Y)$

(2) $X \rightarrow Y_1 \rightarrow Y_2 \quad \Rightarrow (O,X,Y_1), (O,X,Y_2)$

(3) $Y \rightarrow X \quad \Rightarrow (O,X,Y)$

In addition, we created the following rule that includes no *item*:

(4) $Y \quad \Rightarrow (O, -, Y)$

Example: "*shinsetsu-de yokatta-desu.* (*kindly and good.*)"

⇒ (*Hotel A*, - , *shinsetsu* (*kindly*)), (*Hotel A*, - , *yoi* (*good*))

Here "-" shows no element (*item* or *value*).

This is because often the *item* corresponding to the *value* is omitted and only the *value* appears in the review text.

To extract additional information, we determined that *item* adds *bunsetsus* including the pattern "noun + *no* (noun modifier particle)" that modifies *item*, and also *value* adds *bunsetsus* including adverbs or the pattern "noun + case particle".

A. Extraction of review tuples

1) Preprocessing: The system divides the Japanese review text into sentences and gives each sentence a dependency tree using Japanese dependency parser KNP [10]. Then the system removes the sentence including such phrasal expressions for desire or demand as "~ *shi-te-hoshi* (*I want someone to* ~)", "~ *nozomashi* (*I wish* ~)", and "~ *ba ureshi* (*It would be nice if* ~)" since users almost never input such queries.

2) Extraction process: The system applies the above 4 rules to dependency trees by giving the following POS pattern restrictions to the rules:

rule 1 X: "noun + *ha/ga/mo* (case/dep. particle)"
Y: verb, adjective, "verbal noun + *suru* (aux. verb)"

rule 2 X: "noun + *ha/ga/mo* (case/dep. particle)"
Y: verb, adjective, "verbal noun + *suru* (aux. verb)"

rule 3 X: "noun + *ha/ga/mo/wo/ni/da/desu* (case/dep. particle, aux. verb)"
Y: adjective

rule 4 Y: verb, adjective, "verbal noun + *suru* (aux. verb)"

The rules are applied to each sentence as follows: First, the system applies *rule 2,1* if the sentence has a modification relation between the *item candidate* and the *value candidate*. Next, *rule 3* is applied to these candidates. Finally, if only a *value candidate* exists, then *rule 4* is applied to it.

An example of the extraction process is shown in Figure 4. First the system parses the sentence and generates the dependency tree. Then the system identifies *item* and *value candidates* and attempts to apply the rules in the order of 2, 1, 3, 4 to these candidates. Finally the system applies *rule 2* twice and then applies *rule 4*. Eventually, the system extracts five review tuples.

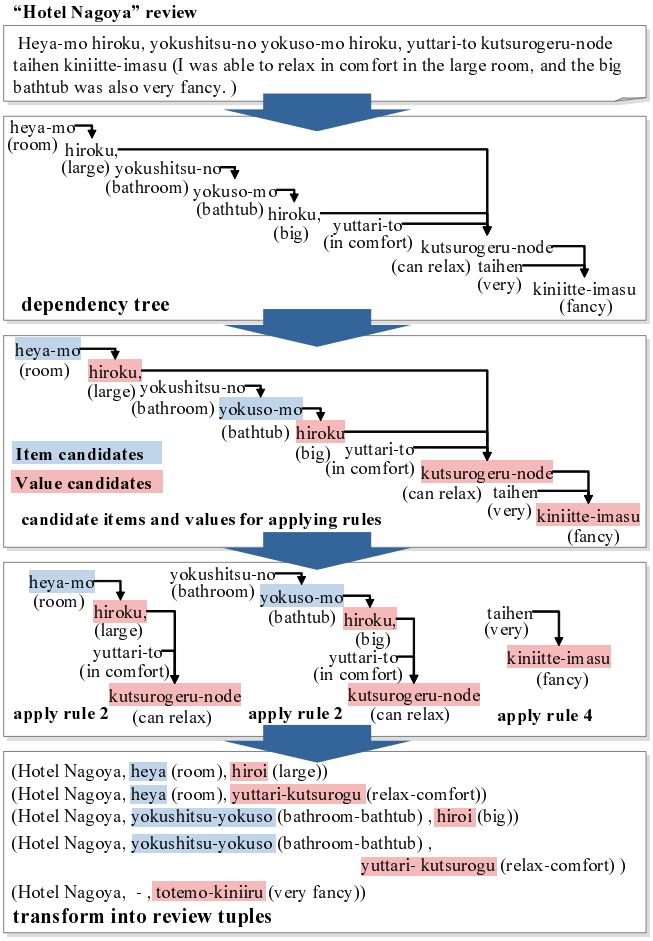

Fig. 4. Example of review tuple extraction

3) Post-processing: The system processes negative facial expressions and then removes function or stop words. In the first step, the system unifies negative outward expression such as "*not*" and "*un-*" and adds negative flags to the review tuple because we want to collectively retrieve such expressions as "*jubun-de-nai* (*not satisfactory*)" , "*fu-jubun-da* (*unsatisfactory*)" without retrieving only one expression and to distinguish negative *values* from positive *values* (as outward expressions).

In the second step, the system removes function or stop words. We determine that morphemes included in *item* are nouns and remove other morphemes (prefix, particles etc.). We also determine that *value* only includes nouns, verbs, adjectives, adverbs, prefixes, and suffixes and other morphemes are also removed.

The system then removes words that have almost no meaning, such as "*koto, mono* (*thing*)", "*omou* (*imagine*)",

and "*kangaeru* (*think*)" , so the system deals with some representations as identical.

IV. Product retrieval

The system retrieves the product (*review-object*) that matches the *query-object* of the query tuple by comparing the *item* and the *value* of the query tuple with the review tuple. Here, *query-object* is a product class or category such as *notebook PC*, *car*, or *hotel*.

The system extracts query tuples from the query using the same method as for the extraction steps. Note that the *query-object* of the query tuple is selected by the user from the product category. In this paper, we experimented by limiting product categories to *hotel*. The head of the query is fixed as "*I am looking for a hotel ˜* ".

A. Matching review and query tuples

The system extracts query tuples from the query by the same method as the extraction process using dependency patterns. For example, it transforms query "*heya-ga kire-de, choshoku-ga tsui-te, nedan-ga yasui yado* (*a hotel with clean rooms, the breakfast is included, and inexpensive.*)" into three query tuples "(*hotel*, *heya* (*room*), *kire* (*clean*))", "(*hotel*, *chosyoku* (*breakfast*), *tsuku* (*included*))", "(*hotel*, *kakaku* (*price*), *yasui* (*inexpensive*))".

Then, the system calculates the *correspondence rate* between each query tuple and each review tuple to compare the *item-value* pair of the query with the review. The *correspondence rate* is calculated by :

$$c\_rate(qt, rt) = \begin{cases} \frac{EOP_{match}}{EOP_{all}} \times \frac{EV_{match}}{EV_{all}} & (has\ a\ item) \\ 0.1 \times \frac{EV_{match}}{EV_{all}} & (has\ no\ item) \\ 0 & (otherwise) \end{cases}$$

EOP_{match} : number of identical *bunsetsus* between items of a query and a review

EOP_{all} : number of all *bunsetsus* in an item of a query

EV_{match} : number of identical *bunsetsus* between values of a query and a review

EV_{all} : number of all *bunsetsus* in a value of a query

We defined a *bunsetsu* as the minimum alignment unit.

Based on the *correspondence rate*, the *review-object* score not only can reflect the frequencies of the review tuple that perfectly matches the query tuple but also reflects the frequencies of review tuples that only partly matches the query tuple. We expect that the more a review tuple correspond to the query tuple, the higher the score of its *review-object* will be.

For example, for query tuple qt_1 (*hotel*, *taiou* (*staff accommodation*), *kaiteki* (*comfortable*)) and review tuple rt_1 (*hotel A*, *taio* (*staff accommodation*), *itsumo kaiteki* (*always comfortable*)), the *correspondence rate* $c_rate(qt_1, rt_1) = 1 \times 1 = 1$. Meanwhile, for query tuple qt_2 (*hotel*, *ryori* (*meal*), *totemo manzoku* (*very satisfied*)) and review tuple rt_2 (*hotel B*, *ryori* (*meal*), *manzoku* (*satisfied*)), *correspondence rate* $c_rate(qt_2, rt_2) = 1 \times \frac{1}{2} = \frac{1}{2}$.

In addition, when the query tuple includes no *item*, the query tuple matches the review tuple including no *item*. That is, as shown in the above formula, the system calculates the *correspondence rate* by replacing *item* parts of the above formula by 0.1, so that the system retrieves products where even no *item* is included in the query tuple.

objective queries

num.	query
Q1	*chekku-auto-go-mo nimotsu-wo azuka-tte-moraeru yado* (*a hotel that keeps baggages after checkout*)
Q2	*shokuji-ga washoku-to yoshoku-de eraberu yado* (*a hotel that features both Japanese and western styles*)
Q3	*chekku-auto-jikan-ga osoi yado* (*a hotel with a late checkout time*)

subjective queries

num.	query
Q4	*furo-ga hiroku-te amenithi-ga jujitsu-shi-te-iru yado* (*a hotel with a large bathtub and many thoughtful amenities*)
Q5	*heya-no shomei-ga akarui yado* (*a hotel with bright room*)
Q6	*shuhen-ga kansei-na funniki-de, ochitsui-te-sugoseru yado* (*a hotel in a quiet area where I can have a very comfortable time*)
Q7	*konbini-ya resutoran-ga chikaku-te shokuji-ni komaranai yado* (*a hotel located near convenience stores or restaurants*)
Q8	*ochitsui-te choshoku-ga toreru yado* (*a hotel where I can eat breakfast leisurely*)
Q9	*heya-no interia-ga kakucho-takai yado* (*a hotel with a high quality interior*)
Q10	*beddo-ga hiroku-te, negokochi-ga yoi yado* (*a hotel with wide beds comfortable*)

Fig. 5. 10 queries used in experiment

B. Calculating relevance score

The system calculates the *relevance score* of each product (*review-object*) and presents products in the order of these relevance scores. To calculate the score, we determine that more appropriate products have the following information:

(1) The more customers write identical information, the more authoritative the information is. (like *tf (term frequency)*)

(2) The fewer products customers write identical information about, the more valuable the information is. (like *idf (invert document frequency)*)

TABLE I
RESULTS OF RETRIEVING 10 QUERIES

	objective			subjective								
query	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	total	avg. prec.
proposal system	3/3	2/9	9/10	7/10	10/10	8/10	3/4	4/10	7/10	10/10	61/86	
	1.000	0.222	0.900	0.700	1.000	0.800	0.750	0.400	0.700	1.000	0.709	0.727
baseline system	7/10	7/10	8/10	6/10	4/10	5/10	7/10	2/10	4/10	8/10	58/100	
	0.700	0.700	0.800	0.600	0.400	0.800	0.700	0.200	0.400	0.800	0.580	0.580

Therefore, the system calculates the relevance score of each product (*review-object*) by the following formula:

$$
\begin{aligned}
Score(Q,O) &= \sum_{q_i \in Q} PF_i \cdot IOF_i \\
PF_i &= \sum_{r_j \in O} pf_j \times c\_rate(q_i, r_j) \\
IOF_i &= \log(\frac{OF}{of_i + 1} + 1)
\end{aligned}
$$

q_i: *review tuple* in query Q
o_j: *object tuple* in object O
pf_i: frequency that review tuple i_k appears in *review-object* i
of_j: number of *review-objects* whose *correspondence rate* to query tuple j are greater than 0
OF: number of all objects

In this formula, PF_{ij} is calculated by multiplying *correspondence rates* between review tuple i and query tuple j by the frequency of the review tuple i. The formula can reflect review tuples that partly matches the query tuple as well as review tuples that fully matches. PF_{ij} resembles the frequency of query tuple j in *review-object* i and IOF_j is logarithm of the frequency of objects appearing the query tuple j. PF_{ij} corresponds to the above (1) and IOF_j corresponds to the above (2).

V. RETRIEVAL EXPERIMENT

A. Outline of experiment

To evaluate the effectiveness of the retrieval method of our system, we experimented with an accommodation retrieval system. Reviews were downloaded from an accommodation reservation site called *Rakuten Travel*[1]. We used KNP [10] as a Japanese dependency parser and evaluated the system by precisions of experiment results of 10 queries, shown in Figure 5. For the top-10 accommodations that the system retrieved, the subject evaluates each accommodation based on checking all reviews of each accommodation for being relevant/non-relevant to each query (by considering whether he/her wants to stay the accommodation or not).

Note that the subject is not one of the authors of this paper.

As a baseline for our experiments, we developed a retrieval system using *tf-idf* term weighting. We determined that index terms are only included nouns, verbs, adjectives, adverbs, prefixes, and suffixes and considered all reviews of each accommodation as one document. It extracts keywords from natural language query and then retrieves documents (accommodations) using their keywords.

[1]Rakuten Travel "*The voices of customers*" http://travel.rakuten.co.jp/auto/tabimado_bbs_top.html

B. Experimental results and Discussion

The result is shown in Table I which represents the precisions of top-10 retrieved accommodations. Compared with the baseline system, our system obtained higher precisions except query Q2, and our system's total and average precision of each query are higher.

Our system presents relevant accommodations more properly and potentially could retrieve accommodations, getting semantic structures between queries and reviews. It, however, has the problem of low recall. It cannot retrieve relevant accommodations that the baseline system can retrieved as Q1, Q2 because it cannot extract the 3-tuples due to the miss of applying extraction rules or dependency parsing.

In contrast, the baseline system achieved higher precision than anticipated. Since it tend to retrieve accommodations that include a larger number of reviews and their accommodations included high frequency keywords, chances are high that their accommodations are relevant to queries.

The following are the causes for decreased the precision of various retrieval results:

Q8 The system could not reflect meanings of the query very well. That means that since it extracted review tuples "(*hotel*, - , *ochitsuku* (*leisurely*)), (*hotel*, *choshoku* (*breakfast*), *toreru* (*eat*))", each tuples matched independent reviews. For example, a certain accommodation included "*kire-na heya-de ochitsuite sugose-mashi-ta.* (*clean room and had a leisurely hours.*)" , "*goka-na choshoku deshi-ta.* (*I can eat lavish breakfast.*)" and had no relevant review to the query. The system should extracted as "(*hotel*, *choshoku* (*breakfast*), *ochituku*(*leisurely*) - *toreru* (*eat*))" *from the query and relevant reviews.*

Q2 *Since the system dropped out meaning "washoku-to yoshoku-de (both Japanese and western styles)", it can not retrieve relevant accommodations well. If it extracted as "(hotel, washoku (Japanese), eraberu (feature)), (hotel, yoshoku (Japanese), eraberu (feature))" from the query and relevant reviews, it could match them.*

VI. Conclusion

In this paper, we presented a product retrieval system robust to subjective queries written in natural language. Using vast amounts of product reviews written by consumers, the system responded subjectively a variety of representation of user queries.

Using with the results of an experiment in the hotel domain with about 220,000 hotel reviews, we confirmed the product retrieval effectiveness of the system. For higher recall, it need to be applied additional methods (e.g., developing item-value pair dictionary).

In the future, we must deal with a variety of lexicons with similar meanings such as "*clean*," "*clear*," "*nice-looking*," and "*good-looking*." Hence, we will examine the retrieval method using a thesaurus.

References

[1] M. Dittenbach, D. Merkl, and H. Berger, A Natural Language Query Interface for Tourism Information, *Proceedings of the 10th International Conference on Information Technologies in Tourism (ENTER 2003)*, pp. 152–162, 2003.

[2] J. Chai, V. Horvath, N. Nicolov, M. Stys, and N. Kambhatla, Natural Language Assistant: A Dialog System for Online Product Recommendation, AI Magazine. Vol. 23, no. 2, pp. 63-75. 2002.

[3] D. Mcsherry, Explanation in Recommender Systems, *Artificial Intelligence Review*, Vo. 24, Num. 2, pp. 179–197, 2005.

[4] M. Hu and B. Liu, Mining Opinion Features in Customer Reviews, *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI 2004)*, pp. 755–760, 2004.

[5] B. Liu, M. Hu, and J. Cheng, Opinion Observer: Analyzing and Comparing Opinions on the web, *Proceedings of the 14th International World Wide Web Conference (WWW 2005)*, pp. 342–351, 2005.

[6] J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack, Sentiment Analyzer: Extracting Sentiments about a Given Topic using Natural Language Processing Techniques, *Proceedings of the Third IEEE International Conference on Data Mining* (*ICDM 2003*), pp. 427–434, 2003.

[7] A.M. Popescu and O. Etzioni, Extracting product features and opinions from reviews, *Human Language Technology Conference* (*HLT*) / *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), pp. 339–346, 2005.

[8] N. Kobayashi, Opinion Mining from Web documents: Extraction and Structurization, Doctoral Dissertation, *Nara Institute of Science and Technology*, 2006.

[9] K. Tateishi, Y. Ishiguro, and T. Fukushima, A Reputation Search Engine That Collects People's Opinions Using Information Extraction Technology, *IPSJ Transaction on Databases*, Vol. 45 No. SIG 7 (TOD 22), pp. 115–123, 2004.

[10] KNP version 2.0, http://nlp.kuee.kyoto-u.ac.jp/nl-resource/knp.html