OKSAT at NTCIR-12 Short Text Conversation Task

-Priority to Short Comments, Filtering by Characteristic Words and Post Classification-

Takashi SATO
Information Processing Center
Osaka Kyoiku University
Kashiwara Osaka Japan
+81-72-978-3823
sato@cc.osaka-kyoiku.ac.jp

Yuta MORISHITA
Graduate School of Education
Osaka Kyoiku University
Kashiwara Osaka Japan
+81-72-978-3823
morishita@ss.osaka-kyoiku.ac.jp

Shota SHIBUKAWA
Department of Arts and Sciences
Osaka Kyoiku University
Kashiwara Osaka Japan
+81-72-978-3823
shibu@ss.osaka-kyoiku.ac.jp

ABSTRACT

Our group OKSAT submitted five runs for Chinese and Japanese subtasks of the NTCIR-12 Short Text Conversation task (STC). We searched not only posts but also comments for terms of each query (post). We also gave more priority to short comments than longer ones. Then we filtered retrieved comments by characteristic words including proper nouns. We added attributes to the corpus and also to the queries. The retrieved comments, which had the same attributes as a query, got an extra score. We classified the queries into three classes for the Japanese subtask, and expanded and searched terms differently. Analyzing experimental results, we observed the effectiveness of our method.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *Information filtering, Query formulation, Retrieval models, Search process, Selection Process.*

General Terms

Experimentation, Performance, Measurement.

Team Name

OKSAT

Subtasks

Chinese subtask
Japanese subtask

Keywords

Information Retrieval, Short Text Conversation, Weibo, Twitter, Priority to Short Comments, Filtering by Characteristic Words, Post Classification, Gram Base Index.

1. INTRODUCTION

The automatic reply system of the micro-blog is an interesting theme in the AI field. Although many researchers have studied this area, fields where satisfactory replies were provided automatically have been very limited. NTCIR provided a task in this area. Our group OKSAT submitted five runs for Chinese and Japanese subtasks of the NTCIR-12 [1] Short Text Conversation task (STC) [2]. We searched not only posts but also comments for terms of each query (post). We also gave more priority to short comments than longer ones. Then we filtered retrieved comments by characteristic words including proper nouns. We added attributes to the corpus and also to the queries. The retrieved comments, which had the same attributes as a query, got an extra score. We classified the queries into three classes, namely 'simple follow', 'greeting' and 'other' for the Japanese subtask, and expanded and searched terms differently. Analyzing experimental results, we observed the effectiveness of our method.

2. OUTLINE OF OUR APPROACH

We searched a corpus by the following procedure for the Chinese subtask (C) and the Japanese subtask (J) of STC, and then we made runs.

(1) Make gram base indices for post and comment (**cmnt** for short) from the corpus.
(2) Prepare search terms from the queries (posts) to search the corpus, and search indices of (1), then get id pairs of post-cmnt.
(3) Score search results of (2) using a probabilistic model [3].
(4) Get cmnt texts from retrieved id pairs of (3).
(5) Give priority to short cmnts over longer ones.
(6) Filter cmnts by characteristic words (proper nouns) in the queries.
(7) Merge scores of (5) and (6). Then we get a run.

Figure 1 shows the above procedure.

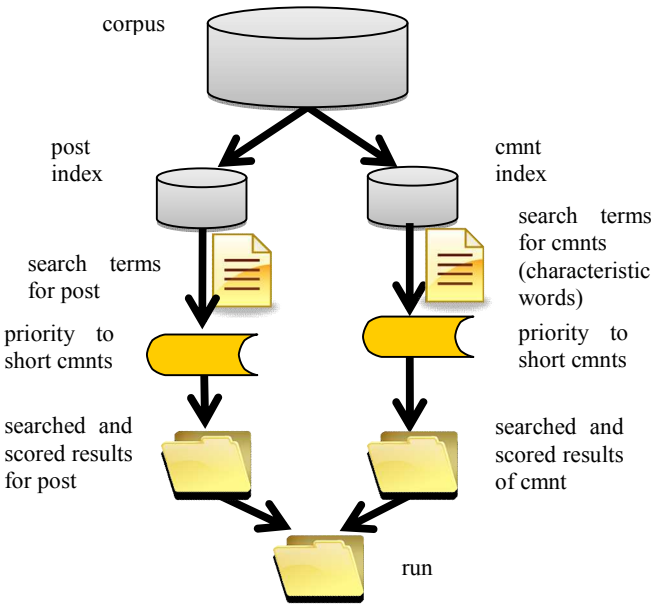

Figure 1. Procedure flow of our approach

3. CHINESE SUBTASK

3.1 Indexing

From the post and cmnt parts of an English translated version of the Chinese corpus, we made **post** and **cmnt** indices

correspondingly. These were gram based indices [4][5][6], so arbitrary string searches were possible using them.

Table 1 shows the specifications of the computer we used. And Table 2 shows the statistics of our indices and their creation time. **C(E)** stands for the English translated version of the Chinese corpus.

Table 1. Specifications of computer

CPU	Intel Core i5-4430@3.0GHz 4C/4T
MEM	8GB, DDR3-1600
O S	FreeBSD 10.1, 64bit
HDD	1TB, SATA 6GB/s, 64MB Cache

Table 2. Statistics of C(E) indices

	post	**cmnt**
data size (MB)	629	202
index size (MB)	1,559	546
time (sec.)	414	140

3.2 Search Terms

We made search terms from queries with the following procedures.

(1) Extract words from a query using TreeTagger [7].
(2) Filter words from (1) using stop words list.
(3) Add phrases.
  (3-1) 'not' + verb such as 'not manage' in Post ID test-post-10160.
  (3-2) Greeting phrase such as 'Happy New Year' in Post ID test-post-10530.
  (3-3) Proper noun such as 'Du Pu' in Post ID test-post-10550.
  (3-4) Whole post text also.

We used (2) and (3) as search terms for the post index, and (3-2) and (3-3) as search terms for the cmnt index.

3.3 Searching and Scoring

We searched the post and cmnt indices of **3.1** with the search terms of **3.2** and scored and ranked retrieved post-cmnt id pairs (the row numbers of the corpus) by a probabilistic model using tf-idf.

Table 3 shows the number of search terms of 100 queries, time to search indices and time to score and rank the retrieved tweet id pairs for the posts and cmnts respectively.

Table 3. Search terms, searching and scoring time C(E)

	post	**cmnt**
search terms	1,048	45
searching (sec.)	74.4	0.31
scoring (sec.)	571*	6.39

In this table, * shows that the computation is executed by 4 processes (the number of CPU cores) in parallel.

3.4 Priority to Short Comments

In the Chinese corpus, the same ID is assigned to the same text. Using this property, we counted the number of identical cmnts in the corpus. Table 4 shows the top ten identical cmnts.

Table 4. Top 10 of the identical cmnts

order	**count**	**cmnt id**	**cmnt text**
1	14830	repos-cmnt-10000003350	Ha
2	8096	repos-cmnt-10000012720	Ha
3	8075	repos-cmnt-10000002260	Ha-ha
4	4443	repos-cmnt-10000004920	ng
5	4297	repos-cmnt-10000007650	Ha
6	4222	repos-cmnt-10000025830	Good
7	3676	repos-cmnt-10000002620	Good
8	3468	repos-cmnt-10000006340	Yes
9	3216	repos-cmnt-10000019930	Like
10	3209	repos-cmnt-10000023490	Great

The reason why the identical cmnt texts appear in Table 4 is that they are different in Chinese even if they are the same in English. The cmnts frequently used are short and correspond to one word of English text. Furthermore, they can be used as highly general purpose cmnts. So, we gave more priority to short cmnts than longer ones. We thought that conversations might be established although shorter texts had less content.

However the fewer the number of words in a cmnt, the more its information decreases. Then we determined that the base number of words is 3. The score multiplied by the number of words Wn is equation (1), where n is the number of words in a cmnt.

$$Wn = \begin{cases} \sqrt{3/n} & (n \geqq 3) \\ 1 & (n = 1, 2) \end{cases} \qquad (1)$$

3.5 Scoring by Proper Noun in Queries

A proper noun often becomes the important keyword in a conversation. We performed a search specifically for proper nouns in order to guarantee association with the query. We included not only words judged to be proper nouns but also to be unknown by TreeTagger as proper nouns. When one thing is expressed by two or more adjacent words, we made them into one search term.

With the above procedure, we were able to extract place names, person names and event names mainly. We also included greetings as an exception. 38 queries (45 terms) out of 100 queries included proper nouns.

We used proper noun terms extracted in order to search the cmnt index. We did this because we thought that the cmnts related to a query could be found by searching cmnts directly with a proper noun of the query. The score of the cmnts which have a proper noun in the query increased. Then we expected that the cmnts with less relation were filtered.

3.6 Scoring by Attribute Information

We added attributes to the corpus and queries. Considering that the same content often appears multiple times in the corpus, we added attributes to posts and cmnts which appear very often (and ones that resemble other cmnts). One cmnt may be given multiple attributes.

Setting keywords and excluded words from original posts/cmnts, we retrieved posts/cmnts which resembled one another in the corpus. Table 5 shows the attributes added.

Table 5. Attribute list

attribute	cmnts added	example of cmnt
positive	150,420	Attractive
agree	141,373	Right
laugh	45,478	Ha
surprise	14,959	Oh
beautiful	14,468	Really beautiful
praise	14,326	Has the talent
lovable	6,726	Lovable
cheer	6,100	Come on!
greeting	4,923	Good night

We added attributes to 398,775 cmnts, which is 7% of all cmnts. We added attributes to 100 queries in the same manner. The retrieved cmnts, which had the same attributes as a query, got an extra score.

3.7 Submitted Runs

We made the following four runs by combinations of the search term sets and scoring techniques.

OKSAT-C-R4: search terms from query only
OKSAT-C-R3: OKSAT-C-R4 + priority to short cmnts of **3.4**
OKSAT-C-R2: OKSAT-C-R3 + scoring by proper noun of **3.5**
OKSAT-C-R1: OKSAT-C-R2 + scoring by attribute of **3.6**

For a comparison, we added a run (OKSAT-C-R5) where we only line up the top ten of Table 10, i.e. no search version. Table 6 shows the official STC Chinese results of our runs.

Table 6. Official Chinese results of OKSAT runs

	Mean nDCG@1	Mean P+	Mean nERR@10
OKSAT-C-R1	0.3267	0.4691	0.3858
OKSAT-C-R2	0.2567	0.3976	0.3743
OKSAT-C-R3	0.2567	0.3965	0.3745
OKSAT-C-R4	0.1433	0.2705	0.2488
OKSAT C-R5	0.2733	0.3796	0.3672

Table 6 summarizes the results of our runs as follows.

(1) OKSAT-C-R1 is the best performance in our runs.
(2) OKSAT-C-R4 is lower than OKSAT-C-R5 (no search version).
(3) Performance of OKSAT-C-R2 and OKSAT-C-R3 are not so different from OKSAT-C-R5.

3.8 Query by Query Analysis

We have some comments about a few queries.

(1) Post ID 'test-post-10010' and 'test-post-10860' have the same post in the corpus. Post ID 'repos-post-1000163280' is identical to 'test-post-10010' and 'repos-post-1001179210' is identical to 'test-post-10860'. Post ID 'repos-post-1000163280' is 19 posts in the corpus and 'repos-post-1001179210' is 19 posts in the corpus, too. OKSAT-C-R4 found these posts and listed counterpart cmnts because our search terms included the query text itself as (3-4) of **3.2**. However these cmnts are judged as not relevant.

(2) Post ID 'test-post-10280' has very similar post in the corpus, 'repos-post-1000651920', 'repos-post-1000914680' and 'repos-post-1001789390'. We found 52, 41 and 2 posts respectively. OKSAT-C-R4 found relevant cmnts in these cases.

3.9 Comparing with Runs of Other Teams

Our best run (OKSAT-C-R1) is ranked 7th out of 44 runs when the measure is Mean nDCG@1. If the run of the best ranked run of each term is compared, our team is third out of 16 teams. If the measure is Mean P+, it becomes 8th of the runs and 4th of the teams. Both the Mean nDCGG@1 and Mean P+ are within 0.05 of the top 8 runs.

4. JAPANESE SUBTASK

4.1 Indexing

We deleted the part following '@', indicating the quotation, from tweet strings of posts and cmnts of the corpus. Then we made gram based post and cmnt indices correspondingly. We used the same computer as for the Chinese subtask. Table 7 shows the statistics of our indices and their creation time. J stands for the corpus for the Japanese subtask.

Table 7. Statistics of J indices

	post	cmnt
data size (MB)	36.6	21.0
index size (MB)	106	61
time (sec.)	17	7.6

The data size of cmnts is smaller than for posts because cmnts have more quotation portions deleted than posts, on average.

4.2 Search Terms

We used the following procedures to make search terms from a query.

(1) Extract words from a query using MeCab [8] with an IPA dictionary (**decab** for short) and MeCab with a neologd dictionary [9] (**xecab** for short).
(2) Filter words from (1) using stop words list.
(3) Classify queries into three classes, namely 'simple follow', 'greeting' and 'other', matching a classification database. For example the database includes 'フォローありがとう', 'RT ありがとう' and so on for the 'simple follow' and 'ただいま', 'おはよう', 'こんにちは' and so on for the 'greeting'.
(4) Expand search terms of the 'greeting' class of (3).
(5) Preliminary post search for the 'other' class of (3).
  (5-1) Characteristic words including proper nouns are extracted from (2) depending on the frequency of the word in the corpus.
  (5-2) Post index is searched for characteristic words by (5-1) and the top three cmnts are obtained.
  (5-3) Using three retrieved cmnts of (5-2), we get three sets of expanded search terms for cmnts.
(6) Get long phases, clauses and sentences from queries for post searches.
  (6-1) Whole query text.
  (6-2) Substring more than 14 characters or longer than half of the whole query text which is divided by punctuation marks, exclamation marks or question marks.

4.3 Searching and Scoring

We searched the post and cmnt indices of **4.1** for search terms of **4.2** and scored and ranked retrieved post-cmnt id pairs (the row numbers of the corpus) by a probabilistic model using tf-idf. We searched the corpus differently according to the class of **4.2**(3).

(1) We searched the post index by search terms of **4.2**(6-1). If more than ten cmnts were found for a query, the following searches were not executed for the query.
(2) We searched the post index by search terms of **4.2**(2) and (6-2) for 'simple follow' class.
(3) We searched the cmnt index by expanded search terms of **4.2**(4) for 'greeting' class.
(4) We searched the cmnt index by three sets of expanded search terms of **4.2**(5-3) for the 'other' class. Then we merged the three sets of results by rotation.

Table 8 shows the number of search terms for 204 queries, the time to search indices, and the time to score and rank the retrieved tweet id pairs for the posts and cmnts respectively.

Table 8. Search terms, searching and scoring time J

	post	cmnt
search terms	1,404	1,393
searching (sec.)	24.9	11.9
scoring (sec.)	147*	16.0

In this table, * shows that the computation is executed by 4 processes (the number of CPU cores) in parallel, as in Table 3.

4.4 Priority to Short Comments

In the Japanese subtask, we gave more priority to short cmnts with respect to the number of characters. We determined that the base number of characters was 20 (=40byte). The score multiplied by the number of characters *Cm* was equation (2), where *m* is the number of characters in a cmnt.

$$Cm = \begin{cases} \sqrt{20/m} & (m \geqq 10) \\ \sqrt{2} & (m < 10) \end{cases} \quad (2)$$

4.5 Scoring by Characteristic Word

In the Japanese subtask, we used not only proper noun words but also general noun words as filters when they were rare. Depending on the appearance of the number of times t_w in the corpus of a noun word w in the queries, we calculated the priority Pt_w by equation (3).

$$Pt_w = \begin{cases} \log_2(12800/t_w) & (100 \leqq t_w \leqq 12800) \\ 0 & (t_w > 12800) \\ 7 & (t_w < 100) \end{cases} \quad (3)$$

16,791 words are analyzed as nouns in the corpus by xecab, and they are used 4,694,031 times in total. There are nouns used more than 50,000 times. We regarded words used more than 12,800 times (28th from the top) as popular and less than 100 times as rare. We defined the priority for popular as 0 and rare as 7, and between them we used the logarithm of 12800/ t_w.

When a noun word w in a query (iquery) appeared in a cmnt retrieved (rcmnt), we added $Pt_w/10c$ to the score. When the word did not appear, we subtracted $Pt_w/10c$ from the score. c is the number of noun words in the retrieved cmnt and 3 if it is fewer than 3. The maximum of Pt_w is 7, 7/(3*10)=0.233 per noun word are added or subtracted at most. So, the score addition Q of this scoring becomes equation (4).

$$Q = \sum_{w \in iquery \wedge w \in rcmnt} Pt_w/10c \; - \sum_{w \notin iquery \wedge w \in rcmnt} Pt_w/10c \quad (4)$$

Although the score is adjusted by equation (4), we expected to filter cmnts depending on the degree of characteristics as in **3.5**. The score of each retrieved cmnt becomes (score * Cm + Q), if both scoring of **4.4** and **4.5** are applied.

4.6 Scoring by Attribute Information

We added an attribute for heaviness to a word in the queries. Then we used it when the post index of the corpus was searched. We show below types and their heaviness in terms of the attributes.

(1) Words which are analyzed by xecab but by decab : 4
(2) Exclamation (greeting) : 3
(3) Hope verb (verb +'たい') : 2
(4) Negation verb (verb +'ない') : 2

4.7 Submitted Runs

We made the following four runs by combinations of the search term sets and scoring technique.

OKSAT-J-R4: search terms of **4.2**(2) + post search using attributes of **4.6**
OKSAT-J-R3: OKSAT-J-R4 + priority to short cmnts of **4.4**
OKSAT-J-R2: search terms of **4.2**(2)-(6) + priority to short cmnts of **4.4**
OKSAT-J-R1: OKSAT-J-R2 + cmnt search using characteristic words of **4.5**

For a comparison, we added a run (OKSAT-J-R5) where we only line up the top ten popular, short and approving cmnts, i.e. no search version. Table 9 shows the official STC Japanese subtask results of the accuracy of our runs.

Table 9. Official Japanese results of OKSAT runs

	2-1	2-5	12-1	12-5
OKSAT-J-R1	0.4574	0.3673	0.7817	0.7050
OKSAT-J-R2	0.4520	0.3583	0.7807	0.6865
OKSAT-J-R3	0.1460	0.1458	0.3876	0.3683
OKSAT-J-R4	0.1361	0.1366	0.3574	0.3543
OKSAT J-R5	0.1807	0.1282	0.5965	0.5196

The results from Table 9 are summarized as follows.

(1) OKSAT-J-R1 has the best performance of our runs.
(2) OKSAT-J-R5 is better than OKSAT-J-R3 and OKSAT-J-R4 except accuracy-2-5.
(3) Not only post search but also cmnt search is effective.

4.8 Query by Query Analysis

We have some comments about some queries.

(1) Nine queries have the same post in the corpus and more than ten posts were found. OKSAT-J-R1 and OKSAT-J-R2 find these posts and list counterpart cmnts because these run have search terms **4.2**(6-1). The accuracy of these queries is judged as high.
(2) The substrings **4.2**(6-2) of ten queries were found in more than ten posts in the corpus. They are effective for the 'simple follow' class.
(3) About queries classified as 'greetings', there were 14 queries which have more than ten cmnts after word expansion of **4.2**(4).
(4) Queries classified as 'others' were not easy. The preliminary cmnt search of **4.2**(5) worked well.

5. CHINESE vs. JAPANESE SUBTASK

We have some comments about differences between the subtasks.

(1) Cmnts of the Japanese subtasks are longer than that of the Chinese subtask. We think Japanese cmnts have more meaning, so we searched cmnts positively.
(2) The corpus of the Japanese subtask is about 10 times smaller than that of the Chinese subtask. We think that relevant cmnts are uncommon, so we expanded search terms positively.

6. CONCLUSIONS

Our group joined and submitted runs for the NTCIR-12 Short Text Conversation task. We searched not only posts but also comments for terms of each query. We also gave more priority to short comments than longer ones. We filtered retrieved comments by characteristic words. We added attributes to the corpus and also to the queries. We classified the queries into three classes for the Japanese subtask, and expanded and searched terms differently. Analyzing experimental results, we observed the effectiveness of our method.

7. ACKNOWLEDGMENTS

Our thanks to Noah's Ark Lab of Huawei Technologies for allowing us to use the Chinese corpus, and to the Japanese subtask organizer for preparing the corpus and queries.

8. REFERENCES

[1] M. Kato and K. Kishida, Overview of NTCIR-12 , in *Proceedings of the NTCIR-12 Conference*, Tokyo, Japan, 2016.

[2] L. Shang, T. Sakai, Z. Lu, H. Li, R. Higashinaka and Y. Miyao, Overview of NTCIR-12 Short Text Conversation Task, in *Proceedings of the NTCIR-12 Conference*, Tokyo, Japan, 2016.

[3] S.E. Robertson and S. Walker, Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval, in *Proceedings of the 17th International Conference Research and Development in Information Retrieval*, pp. 232-241, 1994.

[4] T. Sato, Fast full text retrieval using gram based tree structure, in *Proceedings of the ICCPOL '97*, Vol.2, pp.572-577, 1997.

[5] T. Sato and K. Han, NTCIR-3 CLIR Experiments at Osaka Kyoiku University - Compression of Gram-based Indices -, in *Proceedings of the NTCIR-3*, Tokyo, December 2002.

[6] T. Sato, T. Satomoto, and K. Han, NTCIR-3 PAT Experiments at Osaka Kyoiku University -Long Gram-based Index and Essential Words -, in *Proceedings of the NTCIR-3*, Tokyo, December 2002.

[7] TreeTagger - a language independent part-of-speech tagger, http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/ (2016/04/01).

[8] MeCab: Yet Another Part-of-Speech and Morphological Analyzer, http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html?sess=3f6a4f9896295ef2480fa2482de521f6 (2016/04/01).

[9] mecab-ipadic-NEologd : Neologism dictionary for MeCab, https://github.com/neologd/mecab-ipadic-neologd/blob/master/README.ja.md (2016/04/01).