OKSAT at NTCIR-12 Short Text Conversation Task

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

Takashi SATO, Yuta MORISHITA, Shota SHIBUKAWA {sato, morishita, shibu}@ss.osaka-kyoiku.ac.jp (Osaka Kyoiku University)

[1] Introduction

- OKSAT submitted five runs for Chinese and Japanese subtask 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.
- 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.

[2] Our Approach

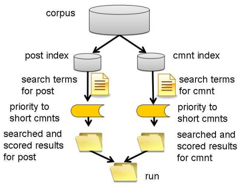

Figure 1. Procedure flow of our approach

[3.1] Chinese - 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, so arbitrary string searches were possible using them.

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] Chinese - 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] Chinese - 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

[3.4] C - 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.

[3.5] Chinese - Query by Query Analysis

- We have some comments about a few queries.
  - 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.
  - 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.

[4.1] Japanese - 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

[4.2] Japanese - Search Terms

- We used the following procedures to make search terms from a query.
  - (1) Extract words from a query using MeCab with an IPA dictionary (decab for short) and MeCab with a neologd dictionary (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] Japanese - 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.

[4.4] Japanese - 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 (1).

$$Pt_w = \begin{cases} \log_2(12800/t_w) & (100 \leqq t_w \leqq 12800) \\ 0 & (t_w > 12800) \\ 7 & (t_w < 100) \end{cases} \qquad (1)$$

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 $12800/t_w$.

[4.5] Japanese - 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.

[5.1] Priority to Short Comments

- We gave more priority to short cmnts than longer ones.
- We thought that conversations might be established although shorter texts had less content.
- In the Chinese subtask, the score multiplied by the number of words Wn is equation (2), where n is the number of words in a cmnt.
- 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 (3), where m is the number of characters in a cmnt.

$$Wn = \begin{cases} \sqrt{3/n} & (n \geqq 3) \\ 1 & (n = 1, 2) \end{cases} \qquad (2) \qquad\qquad Cm = \begin{cases} \sqrt{20/m} & (m \geqq 10) \\ \sqrt{2} & (m < 10) \end{cases} \qquad (3)$$

[5.2] Scoring by the length of text (J)

- We thought that the post which has long text expects long text for cmnt, so we try to add extra score from the length of text.
- The length of short text in corpus is between 1 and 140.
- We surveyed length of post text (1-140 chars) and length of its cmnt.
- For example, the post text which has 22 chars expects most the cmnt text which has 16-20 chars (20.6%).
- We re-calculated score for OKSAT-R3-J by equation (4).

$$S' = S + log(P \times 100) \times 1/100 \qquad (4)$$

- S' means original score made by OKSAT-J-R4 system.
- S means re-calculated score for OKSAT-J-R3.
- P means probability of every length of cmnt text.

[6.1] Scoring by Attribute Information(C)

- In the corpus, the some texts have attribute information
- So, we added attribute to some cmnt texts.
- Table 5 shows example attributes added to text.

Table 5. Example text added attribute

Corpus id	Text	Attribute
repos-cmnt-1000003490	Attractive	positive
repos-cmnt-1000037460	Agreement	Agree

•And, Table 6 shows 9 attributes we defined.

Table 6. 9 attributes

Attribute	Added texts	Attribute	Added texts
positive	150,420	praise	14,326
agree	141,373	lovable	6,726
laugh	45,478	cheer	6,100
surprise	14,959	greeting	4,923
beautiful	14,468		

[6.1] Scoring by Attribute Information(C)

- The score from attribute information is influenced by the number of added texts.
- The less the number of added text, the more their score is higher.
- The score is calculated by equation (5).

$$Attr_i = \sqrt{log\{(\sum a_k)/a_i\}} \times 0.1 \qquad (5)$$

- $\sum a_k$ means number of all texts added attribute.
- a_i means number of texts added i th attribute.

[6.1] Scoring by Attribute Information(C)

- We added attributes for word included in cmnt text.
- For example, when we added 'agree' attribute, we searched text including word 'agree' in cmnt corpus.
- And other example, we added 'happy' attribute, we searched text including word 'happy' but no including 'No'.

- We added attribute to about 7% of cmnt corpus.
- We also added attribute to query text.
- The attribute is expected for reply.

- If a query text's attribute matched cmnt text's attribute which is the result of search, the score re-calculated by equation (6).

$$Score = S + S_{attr} \qquad (6)$$

[6.2] Scoring by Attribute Information(J)

- We defined 5 kind of attributes for query texts when we generated search words.
- Every attribute had weight for scoring.
- Table 7 shows attribute and its weight.

Table 7. Attribute and weight

attribute	condition	weight
Special[s]	Special word	4
hope[h]	動詞＋たい	2
negative[n]	動詞＋ない	2
impression[i]	感動詞	3

- [h] and [n] are words effected by user's opinion.
- [i] is generated mainly from greeting words.
- For example, post is 'おはようございます' and cmnt is 'おはよう' .

[6.2] Scoring by Attribute Information(J)

- We defined Special word[s].
- We used MeCab system with 2 dictionaries (decab system and xecab system). Special word was consisted by difference between 2 systems.
- Table 8 shows a part of results using decab and xecab for two words following 'IPSJ' and '情報処理学会'.

Table 8. Sample result from two systems

System	IPSJ	情報処理学会
decab	No data in dictionary	情報処理＋学会
xecab	IPSJ	情報処理学会

- decab could not analyze 'IPSJ' but xecab could analyze.
- decab analyzed '情報処理学会' into '情報処理' and '学会' but xecab analyzd it into '情報処理学会'.
- So our system defined 'IPSJ' and '情報処理学会' as a [s].

[7.1] Chinese - 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 5.1
  - OKSAT-C-R2: OKSAT-C-R3 + scoring by proper noun of 3.5
  - OKSAT-C-R1: OKSAT-C-R2 + scoring by attribute of 6.1
- 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 9. 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

[7.2] Japanese - 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 **6.2**
  - OKSAT-J-R3: OKSAT-J-R4 + Scoring by the length of text of **5.2**
  - OKSAT-J-R2: search terms of **4.2**(2)-(6) + priority to short cmnts of **5.1**
  - OKSAT-J-R1: OKSAT-J-R2 + cmnt search using characteristic words of **4.4**
- 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 10. 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