WIDM at the NTCIR-13 STC-2 Task

Yu-Han Chen
Department of Computer Science and Information Engineering
National Central University, Taiwan
102401053@cc.ncu.edu.tw

Sébastien Montella
Department of Computer Science and Information Engineering
National Central University, Taiwan
sebastien.montella@utbm.fr

Wei-Han Chen
Department of Computer Science and Information Engineering
National Central University, Taiwan
104522042@cc.ncu.edu.tw

Chia-Hui Chang
Department of Computer Science and Information Engineering
National Central University, Taiwan
chia@csie.ncu.edu.tw

ABSTRACT
In this paper, we describe our contribution for the NTCIR-13 Short Text Conversation (STC) Chinese task. Short text conversation remains an important part on social media gathering much attention recently. The task aims to retrieve or generate a relevant comment given a post. We consider both closed and open domain STC for retrieval–based and generation-based track. To be more specific, the former applies a retrieval-based approach from the given corpus, while the later utilizes the Web to fulfill the generation-based track. Evaluation results show that our retrieval–based approach performs better than the generation-based one.

Team Name
WIDM

Subtasks
Chinese subtasks

Keywords
Short Text Conversation, Chatbot, Retrieval System, Ranking SVM

1. INTRODUCTION
Natural Language Conversation is challenging due to the difficulty to understand an input sentence semantically and to give a corresponding reply. Short Text Conversation (STC), which refers to only one round conversation, plays a big part on social media. More and more users are just commenting posts of other users. As a consequence, the amount of conversations on social media is increased dramatically.

The NTCIR-13 STC Chinese task aims to retrieve or generate a relevant comment on a given post. The contest provides a dataset which consists of post and comment pairs from *Weibo*, a Chinese social media. Another smaller training dataset is provided with labels to indicate the appropriateness of selected post-comment pairs. Unlike previous NTCIR contest where only retrieval-based approach could be opted, generation-based is allowed.

Retrieval-based track [1] evaluates comments that are retrieved from the given *Weibo* corpus. Thus, our task first retrieves a set of candidate comments from our IR system then re-rank them and retrieve the top N comments retrieved. We used distributed word representation as [2] to represent posts and comments. We proposed two strategies for re-ranking. The first ranking method based on the cosine similarity between a post and a comment after getting the vector representation of theirs. The second ranking technique is based on a SVMRank [2] model with handcrafted features from post and comments.

Generation-based track enables much more creativity since the output is not restricted to the given repository. Therefore, we make use of Google search engine to select candidate sentences from the web. We use Google search engine to obtain search results by using the input post as a query and select candidate sentences from snippets for further ranking. We made use of eHownet [3] to generate features to sentence ranking.

This paper is organized as follow. Section 2 introduces related work to the techniques used in this work. Section 3 introduces our approaches, going through the retrieval-based and generation-based techniques step by step. Section 4 shows our experiment and parameters we've chosen. Finally, section 5 is our conclusion on the STC-2 Task.

2. RELATED WORK
Several methods have been proposed to deal with STC tasks. We can categorize these methods into two groups depending on the need of large data or not. Rule-based and reinforcement learning based models require few or no data at all to create a short conversation [1]. Indeed, only rules need to be defined to make the conversation going without correct input/output pairs. On the contrary, other methods such as retrieval-based or generation-based ones use large data from social media to generate response for the STC task. The data collected are used to build a question-answer pair repository for an information retrieval (IR) oriented solution. For example, Ji et al. (2014) in [4] proposed an IR system for STC via three stages. The first

[1] Retrieval-based track doesn't offer any chance of creativity since the selected comment would be retrieved from a repository.

[2] https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

[3] http://ehownet.iis.sinica.edu.tw/

stage is the retrieval of candidate post-comment pairs with the post as a query to get a reduced candidates set. The second stage computes features of the post-comment pairs that will be used for the last stage to train a linear *RankingSVM* model. The features introduced are called matching feature, which try to catch the link between the post and the comment. They use different way to compute features, from IR techniques such as *idf* metric to deep neural network to model the relation between post and comment.

Recently, new techniques such as recurrent neural network are also applied to sequence-to-sequence model construction. For example, [1] proposes a neural network-based response generator by implementing an encoder-decoder model using recurrent neural network. Most generated responses are relevant to the post with a good grammar.

3. OUR APPROACHES

While the name of retrieval-based and generation–based methods suggest two kinds of approaches for STC, it also suggests a closed vs. open domain contest tracks. Technically speaking, retrieval-based approaches seem easier since they are free from grammatical and spelling checking during response generation, which are the major challenges for generation-based approaches. Thus, even if generation-based approaches are preferred for creativity and marginal output, we consider the generation-based track as an open domain STC and make use of the Web for sentence sources. We introduce hereinafter both approaches and detail the concepts and tools applied.

3.1 Retrieval-Based Track

Retrieval-based techniques for short conversation requires large dataset to include all kinds of topics that a user may come up with. There are more than 4 million of Post-Comment pairs from *Weibo*, and a second dataset which is made of 15 ranked comments for a post. The huge amount of post-comment pairs guarantees a small lexical gap. A short conversation system should give a response for any topics. However, this can be challenging when topics are rarely pointed out or related to closed-domain.

Our retrieval-based system follows Ji et al. (2014)'s approach. The first module selects candidate comments via an IR system. Then, the second module re-ranks the candidates just retrieved to provide a ranked list of the comments for each query.

3.1.1 Retrieval System

We use *Solr*[4] to store the Post-Comment pairs. *Solr* offers integrated IR techniques to store and query among the documents stored. In order to index documents with Chinese words but not Chinese characters, we changed the default tokenizer to *HMMChineseTokenizerFactory*. For query processing, we use the *Solr Dismax* parser to match multiple fields with different relevance weights on post (0.8) and comment (0.2) and return the max score across fields. In other words, we will stress more on post than on comment during searching process. However, if a comment also matches some of the query terms, it could be also retrieved.

[4] http://lucene.apache.org/solr/

For a query *q* (*i.e.* the post), *Solr* tokenizes *q* and checks the (inverted) index documents containing words from q. Each document is given a score based on the *tf-idf* metric. The *tf-idf* metric is recalled in (1) with fi,k as the frequency of the word *k* in the document *i* . *nk* is the number of documents in which the word *k* appears. *Lucene* uses this metric because of the high quality of the search results measuring the importance of terms within a document.

$$(1) \qquad tf - idf = \frac{f_{i,k}}{\sum_{j=1}^{t} f_{i,j}} \times \log(\frac{N}{n_k})$$

We decided to retrieve the top *N* documents from *Solr* and to re-rank the documents with different techniques described in the following.

3.1.2 Re-ranking

A re-ranking step here can be useful to reveal relevant documents that have been ranked low by *Solr*. We want to give a chance for these documents to be selected for its comment field to be potentially the future answer of our system. Since users can use of different words to express the same idea, it is challenging to identify them. To cope with this issue, we want to use words representation inspired by [2] to get a customized vector representation of sentence.

A skip-gram model is first applied to learn vector representation of words of length 300 as described by Mikolov et al. in [2]. For a sentence vector representation, we concatenated the minimum value of each dimension among all of the word's representation, and the maximum value of each dimension, which results in a 600 dimension vector.

We expect our sentence representation to catch not only the semantic of the query but also the *post* field from the documents retrieved in order to compare them to re-rank the documents well. We adopted two different techniques for re-ranking.

- *Cosine Similarity*

After retrieving the top *N* documents from *Solr*, we used our vector representation of sentence to transform our query and the *comment* field into vectors of 600 dimensions. Our goal is to measure how close the query given and the comments retrieved semantically. We calculated cosine similarity between the query's and the comment's vector representation. The cosine similarity gives us a metric to judge whether the comment retrieved is responding to the same topic of the query or not. Thus, the higher the cosine similarity, the more alike query and comment. We re-ranked the N documents based on the cosine similarity. The top documents will then be given as replies to the query.

- *Learning to Rank*

We adopted a second method to re-rank the documents. We designed 7 handcrafted features by the query and the comment retrieved by *Solr* along with the labeled training data. These features aim to give more information about the relationship between the query given and the retrieved comments from *Solr*. The 7 features are as follow:

1. The cosine similarity between the query's and the retrieved comment's sentence vector described earlier..

2. The cosine similarity between the nouns contained in the query and the comment. Nouns are good representatives of topics mentioned in a sentence.

3. The cosine similarity between the verbs in both query and the comment. Verbs stands for the action or state of entities within a sentence and therefore including them will help to rank higher comments with similar verbs.
4. The cosine similarity between the proper nouns that might be contained in the query and the comment.
5. The cosine similarity between the Points of Interest (POI) contained in the query and in the comment.
6. The number of common words between the query and the comment. More common words could mean that the comment retrieved should be more relevant.
7. The difference between the number of words in the query and the comment. The length of the query and the comment might help the machine learning algorithm.

We make use of the second dataset provided by the contest to train a Ranking SVM model to re-rank the comments using the 7 features. We used this dataset to train *RankingSVM* after computing the features for each query-comments pairs. The final model will re-rank the comments and the top one will be selected as an answer to the query.

We proposed for the retrieval-based part three different models for submission. The first model WIDM-C-R1 is taking the whole input sentence of the user as a query with the first technique introduced applied for re-ranking. The second model, coined WIDM-C-R2, is only querying *Solr* with the nouns, verbs and adjectives and the re-ranking is done with the first technique also. Finally, the last model WIDM-C-R3 is taking the complete user input, and we apply the second technique for re-ranking. All performance results are given in the section 4.2.

3.2 Generation-Based Method

As mentioned above, we consider the generation-based track as an open-domain sentence selection problem to avoid the grammar and spelling check to ensure the correctness of output response. We adopted an original strategy by making use of search engine to obtain several snippets for candidate sentence filtering. These will be ranked with different kind of features.

3.2.1 Candidates Generation

Our approach is to use Google search engine to get candidate sentences from the snippets obtained with the user input as a query. As shown in Fig. 1, Google search engine marks in red the words contained in the query. When the longest marked-in-red string is similar enough (>0.5) to the query, we consider the search result is relevant to the query. To ensure the correctness, if there are more than three snippets having the marked-in-red string longer than half the query, we extract the text after it.

If, however, the condition is not fulfilled, we will re-query Google with a new query, including the concatenation of strings marked in red provided that these are nouns or stand for a Wikipedia page title. As illustrated in Table 1, we use eHowNet to check if the word is a noun (*Na*, *Nc*, etc).

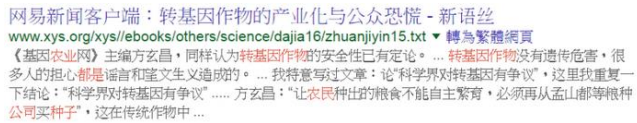
网易新闻客户端：转基因作物的产业化与公众恐慌 - 新语丝
www.xys.org/xys//ebooks/others/science/dajia16/zhuanjiyin15.txt ▾ 轉為繁體網頁
《基因农业网》主编方玄昌，同样认为转基因作物的安全性已有定论。 ... 转基因作物没有遗传危害，很多人的担心都是谣言和骇文生义造成的。 ... 我特意写过文章：论"科学界对转基因有争议"，这里我重复一下结论："科学界对转基因有争议" ..... 方玄昌："让农民种出的粮食不能自主繁育，必须再从孟山都等粮种公司买种子"，这在传统作物中 ...

Figure 1. A snippet example for the query "鼓励种转基因作物的种子公司都是玩弄各国政府各国农业部 还有农民的骗子".

Table 1. Checking for Na or Nc in EHowNet and Wikipedia page for marked-in-red word segments in search snippets

字串	In eHowNet	In Wiki
农业	Na	-
转基因作物	-	True
都是	-	-
农民	Na	-
公司	Nc	-
种子	Na	-

For example, the new query contains 农业+转 基因作物+农民+公司+种子. We then query Google search engine again with the new query. Within new search snippets, the sentences containing strings marked in red will be selected as candidates.

However, candidates selected from the Google search engine need, for both strategies, a first text-cleaning step. Indeed, noise can be included in snippets such as useless punctuation or incomplete sentences. We segment sentences based on period (.), question mark (?), exclamation mark (!), and ellipsis (…). Sentences containing ellipsis are removed from the candidates set. Then, we use CKIP parser to keep sentences which includes an S or VP tags, standing for "Sentence" level and "Verb Phrase" respectively. This aims to get reasonable candidates that will be ranked in the next part.

3.2.2 Candidates Ranking

Next, we define features that will be used to rank candidates with the labeled training dataset provided by the contest. Features are of two kinds: binary and numerical features. Binary features include the sentence type and some eHowNet categories as described in Table 2. We categorize sentences into 5 types including question, judgement, whether, narrative and expression based on simple keywords or POS Tags. Furthermore, eHowNet allows us to indicate whether a sentence contains mental state, mental act, modality value, and degree value or their hyponyms.

Table 2. Binary features classification

	Binary Feature	Classification
sentence type	question	"吗","哪 里","什 么","?","哪些"
	judgment	"是","不是"
	whether	string : "有","沒有"
	narrative	CKIP-pos tag:"VA", "VAC","VB","VC", "VCL","VD","VE","VF"
	expression	CKIP-pos tag: "VH", "VHC","VI","VJ","VK", "VL"
eHowNet	MentalState	MentalState and their Hyponym
	MentalAct	MentalAct and their Hyponym
	ModalityValue	ModalityValue and theirHyponym
	DegreeValue	DegreeValue and their Hyponym

The numerical features are inspired from [3]. We represent the candidate sentence with a vector by taking the average of all words embedding in the sentence [2]. We use metrics in the vector space as features. We use the minimum cumulative distance [3] which can be defined as the minimum distance that is needed to *travel* from one sentence to another one in the vector space (Fig. 2). We use also known metrics such as Euclidean distance and cosine similarity to catch relationship between post and comment.

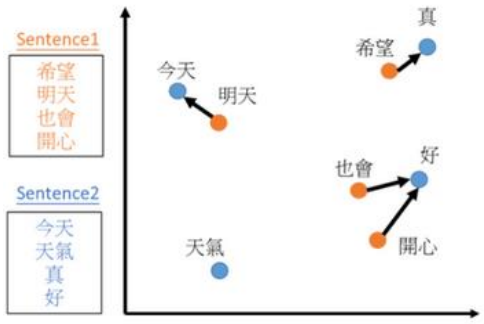

Figure 2. An Illustration of the minimum cumulative distance. The minimum cumulative distance is calculated by adding up the distance needed from all words in sentence 1 to match words in sentence 2.

The last numeric feature used is the bounding box overlap. We represent a sentence with a vector. By keeping the minimum and maximum values for a sentence representation, we can plot in the vector space the polygon corresponding to the sentence. Thus, by plotting two sentences in the vector space, we can get their intersection and therefore their similarity regarding the meaning the sentences convey. The bigger the intersection, the higher the relevancy.

The features are used to train a Ranking SVM model on the training dataset. We first compute the features for the whole post and comment pairs provided. Once trained, the model is used to rank candidates generated from the snippets. The top one will be given as the answer of the input query.

4. EXPERIMENT

4.1 Experiment Details

We used *Solr* for our retrieval-based experiments and coded in Java for our system, and since we are dealing with simplified Chinese, we changed the tokenizer to "HMMChineseTokenizer Factory". For each document in *Solr*, we have *post, post_id, comment, comment_id*, respectively. We tuned the weight on the four attributes and put more weight on *post*, which means the documents retrieved will be more related to post.

For both retrieval-based and generation-based methods, we use the first dataset to train the word embeddings [2]. The length of the vectors for the retrieval-part is 300 and 250 for the generation part. We tried to retrieve 30, 50, 100 documents at a time to do the re-ranking. It shown that 50 is the best number of documents to retrieve since more potential documents are included. Thus, although some documents that were ranked low by *Solr* can be revealed by our re-rank approaches.

All the three models were tested with the same data, which was provided in this contest. For WIDM-C-R1, we used cosine similarity of our customized document vector to re-rank the documents, although simple, it turned out to be having the best performance both under our evaluation measure and the official's.

4.2 Results

We adopted NDCG as our evaluation metric, since it is a widely used and reliable method to assess the ranked documents. Based on our evaluation method, the first model WIDM-C-R1 which yields the first run had an 0.769; the second model WIDM-C-R2 got 0.705, and the last model WIDM-C-R3 resulted in 0.658. According to the result released by the contest, WIDM-C-R1, we got 0.3620, 0.4950, 0.5238 for nG@1 (normalized gain at cutoff 1), P+, nERR@10 (normalized expected reciprocal rank at cutoff 10) respectively.

We also adopted the NDCG metric for the generative model. Using 4-cross validation, we got 0.704 on training data. The results released by the contest shown performances of 0.1437, 0.2311, 0.2034 for nG@1, P+, nERR@10 respectively.

Therefore, we observed that the retrieval-based approach outperforms our generation-based method. On average, our top retrieval-based model is performing 141% better on the competition metrics than the generation based. Indeed, our retrieval-based uses only IR techniques combined with a huge post-comment pair corpus for word embedding.

The dataset coming from a social media, it is likely to cover most common topics. However, when using search engine as our generation approach, the results doesn't guarantee a good relevancy value because of noise that snippets might include. Despite a ranking step, if the candidates from the snippets are not relevant enough to the query of the user, the performance will be low. Such configuration may happen often since a snippet is not acting as a reply to a query but as a search result. Thus, our retrieval techniques perform better by making use of the huge post-comment pairs.

5. CONCLUSION & FUTURE WORK

In this paper, we propose two approaches for the STC tasks, retrieval-based and generation-based approaches. We use *Solr* to index our post-comment pairs and to make use of integrated IR techniques. Using the user input as a query, we retrieve several

candidates from the repository that will be re-ranked by different techniques applying distributed words representation. On the contrary, our generation-based approach is using Google search engine to get candidates from snippets after querying with the user input. A ranking of the candidate is also done with handcrafted features. Our experiment shows that our retrieval approaches perform much better than the generation-based one. Features engineering and features selections are ideas for future work to see whether, with a deeper features design, performances of the generation-based system can perform as well as the retrieval-based approaches.

6. REFERENCES

[1] Lifeng Shang and Zhengdong Lu and Hang Li, *Neural responding machine for short-text conversation*, arXiv preprint *arXiv:1503.02364*, 2015.

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean, *Distributed Representations of Words and Phrases and their Compositionality*, NIPS 2013

[3] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, Kilian Q. Weinberger, *From Word Embeddings To Document Distances*, Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 2015

[4] Zongcheng Ji, Zhengdong Lu, Hang Li, An Information Retrieval Approach to Short Text Conversation, August 2014.

[5] Lifeng Shang and Tetsuya Sakai and Hang Li and Ryuichiro Higashinaka and Yusuke Miyao and Yuki Arase and Masako Nomoto, *Overview of the {NTCIR}-13 Short Text Conversation Task*, Proceedings of NTCIR-13, 2017.