BUPTTeam Participation in NTCIR-12 Short Text Conversation Task

Yongmei Tan
Beijing University of Posts and Telecommunications, China
ymtan@bupt.edu.cn

Minda Wang
Beijing University of Posts and Telecommunications, China
wangminda@bupt.edu.cn

Songbo Han
Beijing University of Posts and Telecommunications, China
hansongbo@bupt.edu.cn

Abstract

This paper provides an overview of BUPTTeam's system participated in the Short Text Conversation (STC) task of Chinese at NTCIR-12. STC is a new NTCIR challenging task which is defined as an IR problem, i.e., retrieval based a repository of post-comment pairs from Sina Weibo. In this paper, we propose a novel method to retrieve post result from the repository based on the following four steps: 1) preprocessing, 2) building search index, 3) comment candidates generation, 4) comment candidates ranking. The evaluation results show that our method significantly outperforms state-of-the-art STC Chinese task.

Team name

BUPTTeam

Subtasks

Short Text Conversation (Chinese)

Keywords

Retrieval, Elasticsearch, Random Walk

1. Introduction

Short Text Conversation (STC) is a new NTCIR-12 task which tackles the following research goal: a STC system which reuses an old comment from the repository to satisfy the author of the new post (Shang et al., 2016).

Compared to QA task like NTCIR-8 CQA task (Ishikawa, 2010), the query type of CQA is only questions, but the query type of STC contains any type of sentences including questions. That means the task of STC has more widely scope of information retrieve. STC task get the top 10 best comment candidates instead of the best one. STC is more difficult than QA task, because it involves varied circumstances of different context. Besides, it is more similar to human conversation, helping researchers find approaches of building simulator of human-computer conversation.

STC task could be approached as an IR problem. The methods to solve IR problem are in different ways, such as taking features of sentence length, different degree of politeness, url resource (Ishikawa et al., 2010), the similarity of questions and answers, the position of answer (Song et al., 2010), the readability of answer (Kuriyama, 2010).

Given the new post, we assume the effectiveness of comments depends on the similarity between the new post and the old comment, or the similarity between the new post and the old post. If the new post is not relevant to the old post or the old comment, the old comment shouldn't be appropriate for the new post.

In this paper, we propose a novel method to retrieve the new post result from the repository based on the following specific steps: preprocessing, building search index, comment candidates generation and comment candidates ranking. We then show the effectiveness of the method of measuring similarity between short texts.

Our contribution is twofold: 1) we apply Elasticsearch[1] to index the repository. In this way, it is very easy and fast to find the related information from repository; and 2) we put forward a graph-based approach for candidates ranking to find the most appropriate comments for a new post.

2. System Architecture

The architecture of our STC system is described as Figure 1. It includes the following four components.

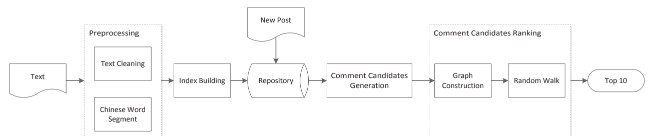

Figure 1: System Architecture

2.1 Preprocessing

There are some traditional Chinese, specific symbols, excess punctuations in raw text. We convert traditional Chinese to simplified Chinese and remove the specific symbols and excess punctuations in order to clean the text.

Word is the smallest meaningful linguistic element which is capable of independent activity. There is no clear distinction between the word marks (Zheng et al., 2013). Therefore, the segment for the Chinese words is the basis and the key to analyze Chinese text. We use Stanford Word Segment[1] to split Chinese text into a sequence of words (Chang et al., 2008).

2.2 Index Building

In the corpus of millions sentences, it's difficult to retrieve the most appropriate comment candidates for a new post. The first is computational efficiency. The repository is huge so that we could not generate comment candidates quickly. The second is short text similarity computing method. To address the above problems, we use Elasticsearch, which is a distributed scalable real-time search and analytics engine, to build index of posts and comments.

2.3 Comment Candidates Generation

Given a new post, there are three steps to generate comment candidates from the repository.

1) The top 10 posts are retrieved by Elasticsearch and then we get the corresponding comments from the repository.

2) The top 10 comments are retrieved by Elasticsearch.

3) From the above 2 steps, we obtain 20 comment candidates.

[1] http://nlp.stanford.edu/software/segmenter.html

In Elasticsearch, the relevance measure $score(p, c)$ between a new post p and a comment candidate c is given by[2]:

$$score(p, c) = pm(p) * cd(p, c) * tf(p) * idf(c) * nm(c) \quad (1)$$

$pm(p)$ is the normalization of a post p so that the results retrieved for one post can be compared with the results for another.

$$pm(p) = \frac{1}{\sqrt{\sum_{w_i \in set(p)} idf^2(w_i)}} \quad (2)$$

$set(p)$ is the set of words of a post p.

$idf(w_i)$ is the inverse document frequency of the word w_i, which is the logarithm of the number of original pairs in the repository *numDocs*, divided by the number of comments containing the word w_i or the number of posts containing the word w_i in repository.

$$idf(w_i) = 1 + log_e^{\frac{numDocs}{docFreg+1}} \quad (3)$$

$cd(p, c)$ is the word overlap percent of the post p and the candidate c.

$$cd(p, c) = \frac{|set(p) \cap set(c)|}{|set(P)|} \quad (4)$$

$|set(p)|$ means the number of words in the $set(p)$.

$tf(p)$ is the square root of the frequency $f(w_i)$ of the word w_i appearing separately in posts or comments.

$$tf(p) = \sqrt{\sum_{w_i \in set(p)} f(w_i)} \quad (5)$$

$nm(c)$ means that a shorter comment candidate c is more important.

$$nm(c) = \frac{1}{\sqrt{|set(c)|}} \quad (6)$$

2.4 Comment Candidates Ranking

Referent graph is a strongly connected graph represented by G= (V, E), where V is the set of all comment candidates of the new post. E is the set of all edges in the referent graph (Han et al., 2011).

To find the most appropriate comments for a new post, we use a referent graph-based approach for candidates ranking instead of directly based on the relevance score of comment candidates for a new post.

• Referent Graph Construction

Given a new post, the number of comment candidates is 20. Each edge is between these comment candidates or between the new post and the comment candidate, so there are two types of edges in Referent Graph. The weight of the edge between one comment candidate c_i and another c_j is semantic similarity $SR(c_i, c_j)$ between comment candidates (c_i and c_j) defined as:

$$SR(c_i, c_j) = \frac{v(c_i) \cdot v(c_j)}{||v(c_i)||||v(c_j)||} \quad (7)$$

Where $v(c_i)$ is the vector of c_i . $SR(c_i, c_j)$ is the cosine similarity of c_i and c_j. $||v(c_i)||$ means the norm of vector $v(c_i)$.

The transition probability matrix T on the graph G can be calculated as:

$$P(p \to c_i) = \frac{score(p, c_i)}{\sum_{c_i \in N_p} score(p, c_i)} \quad (8)$$

$$P(c_i \to c_j) = \frac{SR(c_i, c_j)}{\sum_{c_j \in N_{c_i}} SR(c_i, c_j)} \quad (9)$$

Where N_p refers to a comment candidates set of a post p. N_{c_i} refers to the set of new post which are adjacent with the candidate c_i.

• Ranking

The random walk original vector α on G is the vector of $|\boldsymbol{V}| \times 1$. After completing of initialization of vector $\boldsymbol{\alpha}$, the sum of all the items of vector $\boldsymbol{\alpha}$ after standardization disposal is 1 thus to make sure that vector $\boldsymbol{\alpha}$ is a correct initialization vector.

Formula (10) and (11) illustrate the process of random walk with restart:

$$r^0 = \alpha \quad (10)$$

$$r^{t+1} = (1 - \lambda) \times T \times r^t + \lambda \times \alpha \quad (11)$$

Where r^t refers to the intermediate result of random walk with restart, t refers to times of iteration, and λ refers to a parameter. Making $r^{t+1} = r^t$, eventual stationary distribution can be calculated as shown in formula (12):

$$r = \lambda(I - mT)^{-1}\alpha, m = 1 - \lambda \quad (12)$$

For a post p, *E(c),* the effectiveness measure of a comment candidate c is defined as follows:

$$E(c) = score(p, c) \cdot r(c) \quad (13)$$

Finally comment candidates are ranked by *E(c)* and a ranking list of ten comments for a new post is acquired.

3. Experiments

3.1 Data Set

Table 1 shows the statistics of the retrieval repository, training data and test data. There are 196,495 Weibo posts and the corresponding 4,637,926 comments. There are 5,648,128 post-comment pairs. So each post has 28 different comments on average, and one comment can be used to respond to multiple different posts.

There are 225 query posts and each of them have about 30 comment candidates in training data. There are 6,017 post-comment pairs with "suitable", "neural", and "unsuitable" labels. "Suitable" means that the comment is clearly a suitable comment to the post, "neutral" means that the comment can be a comment to the post in a specific scenario, while "unsuitable" means it is not the two former cases.

100 posts are used for test. We are permitted to submit up to five runs to the task. In each run, a ranking list of ten comments for each test query is requested.

Table 1: The Dataset of Sina Weibo

Retrieval Repository	#posts	196,495
	#comments	4,637,926
	#original pairs	5,648,128
Training data	#posts	225
	#comments	6,017
	#labeled pairs	6,017

[2] https://www.elastic.co/guide/en/elasticsearch/guide/current/practical-scoring-function.html

Test Data	#query posts	100

3.2 Evaluation Metrics

The evaluation metrics are nG@1, nERR@10 and P+ (SaKai et al., 2015).

nG@1 shows the quantity of effective result (such as L1, L2 result) in the retrieved candidates. It will take three values: 0, 1/3 or 1 in this task.

nERR@10 shows the rank correctness of the candidates ranking, which means that the more effective result should be ranked as more front of the ranking list of retrieved candidates.

P+ depends most on the position of the best effective result in the ranking list of retrieved candidates. It gives the top ranked result the most ratio.

3.3 Experimental Results

The best five teams with their best run results are shown in Table 2. The runs have been sorted by Mean nG@1, P+ and nERR@10, respectively.

Table 2: Part of Official STC results

Run	nG@1	Run	P+	Run	nERR@10
BUPTTeam-C-R4	**0.3567**	BUPTTeam-C-R4	**0.5082**	BUPTTeam-C-R4	**0.4945**
MSRSC-C-R1	0.3367	MSRSC-C-R1	0.4854	MSRSC-C-R1	0.4592
OKSAT-C-R1	0.3267	splab-C-R1	0.4735	splab-C-R1	0.4449
ITNLP-C-R3	0.3067	OKSAT-C-R1	0.4691	Nders-C-R1	0.4196
splab-C-R1	0.2933	USTC-C-R5	0.4509	ITNLP-C-R3	0.4186

As can be seen, our approach on short text conversation task, reporting state-of-the-art performance on multiple evaluation metrics.

Table 3 shows the performance of our different runs in the task. The setting of each run will described as follows:

1) R1 ranks the comment candidates based on the relevance score.
2) R2 ranks the comment candidates by random walk with restart.
3) R3 ranks the comment candidates by random walk with restart except for the one with the highest relevance score.
4) R4 ranks the comment candidates by random walk with restart while date and time expressions are considered.
5) R5 ranks the comment candidates by random walk with restart except for the one with the highest relevance score and date and time expressions are considered. R5 removes punctuations and stop-words from text.

Table 3: Comparison of Performance on 5 Runs

Run name	nG@1	P+	nERR@10
BUPTTeam-C-R1	0.3400	0.4853	0.4770
BUPTTeam-C-R2	0.3533	0.4883	0.4805
BUPTTeam-C-R3	0.3533	0.4933	0.4830
BUPTTeam-C-R4	**0.3567**	**0.5082**	**0.4945**
BUPTTeam-C-R5	0.3467	0.4854	0.4800

As we can see from the table, R4, which consider date and time during ranking, can improve the evaluation measures significantly.

With the use of random walk, the result of R2 improves against R1 to a certain extent, which signifies the effectiveness of the random walk.

Random walk with restart can improve the performance in most setting for all five runs, which confirms the general effectiveness of this method. For our R4, we can see that ranking benefits from punctuations and stop words.

4. Conclusions

In this paper, we propose an approach for STC task of NTCIR-12. We use Elasticsearch to build search index and the random walk which is a graph-based method to rank comment candidates. The evaluation results show that our method significantly outperforms state-of-the-art STC task.

5. References

[1] Daisuke Ishikawa, ASURA: A Best-Answer Estimation System for NTCIR-8 CQA Pilot Task, 2010.

[2] Daisuke Ishikawa, Tetsuya Sakai, Noriko Kando. Overview of the NTCIR-8 Community QA Pilot Task: The Test Collection and the Task. NTCIR-8 Workshop Meeting, 2010.

[3] Kazuko Kuriyama. Best-Answer Selection Using a Machine Learning Tool at NTCIR-8 CQA Pilot Task, 2010.

[4] Karl Pearson. The Problem of the Random Walk. Nature, 1905, 268: 2113–2122.

[5] Leo Grady. Random walks for image segmentation. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2006, 28(11): 1768–1783.

[6] Lifeng Shang, Tetsuya Sakai, Zhengdong Lu, Hang Li, Ryuichiro Higashinaka, Yusuke Miyao. Overview of the NTCIR-12 Short Text Conversation Task, 2016.

[7] Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. Optimizing Chinese word segmentation for machine translation performance, ACL, 2008.

[8] Tetsuya Sakai, Lifeng Shang, Zhengdong Lu, and Hang Li. Topic Set Size Design with the Evaluation Measures for Short Text Conversation, 2015.

[9] Xianpei Han, Le Sun and Jun Zhao. Collective entity linking in Web Text: a graph-based method. Proceedings of International Conference on Research & Development in Information Retrieval. Beijing, 2011.

[10] Xiaoqing Zheng, Hanyang Chen and Tianyu Xu. Deep Learning for Chinese Word Segmentation and POS Tagging. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013.

[11] Young-In Song, Jing Liu, Tetsuya Sakai, Xin-Jing Wang, Guwen Feng, Yunbo Cao, Hisami Suzuki and Chin-Yew Lin.

Microsoft Research Asia with Redmond at the NTCIR-8 Community QA Pilot Task, 2010.

[12] Zhaochen Guo, Denilson Barbosa. Robust Entity Linking via Random Walks. Proc. CIKM, 2014.