BUPT_PRIS at TREC 2012 Crowdsourcing Track1:

Chuang Zhang, Minjie Zeng, Xiaokang Sang, Kailai Zhang, Houfu Kang

Pattern Recognition and Intelligent System Lab,

Beijing University of Posts and Telecommunications, P.R.China

Abstract

In this paper, the strategies and methods used by the team BUPT-WILDCAT in the TREC 2012 Crowdsourcing Track1 will be mainly introduced. The Crowdsourcing solution is designed and carried out on the CrowdFlower Platform. Corwdsourcing tasks are released on the AMT. The relevance labels are gathered from workers of AMT and optimized by the inner algorithms of Crowdflower Platform.

0. Introduction

The task of TREC 2012 Crowdsourcing Track1 is Text Relevance Assessing Task and it's goal is to evaluate approaches to text relevance assessing. There are 18260 topic-docno pairs to be judged. These topic-docno pairs represent the "test set" for the Text Relevance Assessing Task (TRAT). For each of the 10 topics, participants will need to provide both a binary relevance decision and a probability of relevance. If a probability of relevance is not possible for a run, that run will only be evaluated based on its binary judgments. And our team only obtained and submitted binary judgments. Moreover, participants of TREC2012 Crowdsourcing Track1 are allowed to do anything to produce the judgments except use the existing qrels.

Considering the issues mentioned above, my team took two steps to complete the TRAT task. In the first part of the paper, we will introduce the approaches our team adopted to process the data set. Because the test documents are sent by two disks being mixed with a considerable number of other unnecessary documents and some test documents are too lengthy, a proper way to ameliorate these documents is very significant.

In the second part, we will elaborate the Crowdsourcing solution we designed to TRAT task. Such as the principles we followed to design jobs on the CrowdFlower platform, the approach we took to design the interface of our tasks, and the methods we took to ensure the good quality of our workers. At last we will briefly introduce the results we gained of the Crowdsourcing Text Relevance Assessing Task.

1. Part 1: Data Processing

In this part, we will introduce our methods of processing the data set of TREC2012 Crowdsourving TRAT tasks. With respect to the data set, there are 10 topics and a certain number of documents for use in the TRAT task. The format of 10 topics is the same, which consists of topic's title,

description, and narrative. However, there are four different format of documents. Totally, 18260 topic-docno pairs composed by topics and documents need to be judged in the TRAT task. These topic-docno pairs represent the "test set" for the TRAT.

First of all, owing to the test data being mixed with lots of irrelevant data, we ought to sort out the exact test data we need of TRAT tasks. The documents we received are stored and shipped in two disks, which contain 4 different formats of data. Therefore, we use the corresponding program to process the data set in order to select the test documents from the two disks we received. The principle of the corresponding program is to sort out the test document in accordance with the document number. For example, in the graph below the FBIS-8665 is the document number, therefore, we can select the document FBIS3-8665 from the FBIS data set according to the DOCNO number.

```
<DOC>
<DOCNO> FBIS3-8665 </DOCNO>
<HT>    "dreeu049_y_94002" </HT>
```

Secondly, on the basis of TREC 2012 Crowdsourcing: Text Relevance Assessing Task Guidelines, all of the participants are allowed to do anything to produce the judgments except use the exiting qrels. That means, we can preprocess the test data before upload it to the Crowdsourcing platform. Therefore, in order to produce better results, we decide to remove some test data that is obviously irrelevant. By reading these ten topics, we find that those relevant documents should contain some key words of the corresponding topic. So we use several key words to screen the test documents. For example, the contents of topic411 is shown in the graph below.

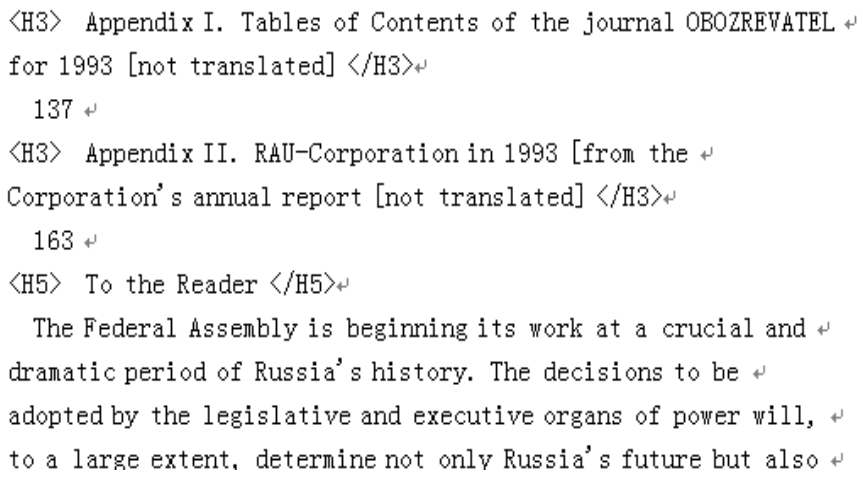
```
<num> Number: 411
<title> salvaging, shipwreck, treasure

<desc> Description:
Find information on shipwreck salvaging: the
recovery or attempted recovery of treasure from
sunken ships.

<narr> Narrative:
A relevant document will provide information on
the actual locating and recovery of treasure;
on the technology which makes possible the discovery,
location and investigation of wreckages which
contain or are suspected of containing treasure; or
on the disposition of the recovered treasure.
```

According to the title, description and narrative of topic411, the key words we considered of topic411 are salvaging, shipwreck（sunken ship）, treasure (precious deposits), and words in the brackets are the synonyms. The way to generate key words is that each member of our team will read the all ten topics and each of us will work out our own key words in the topic, then we will discuss and obtain the final key words of each topic.

Because we consider that all of the relevant documents should contain some of the key words, if a document contains no key words, we will mark the document with irrelevant tag. That means, the document that contains no key words is no need to be judged by the workers of the CrowdFlower

platform. Finally, thanks to this principle, we get around 4000 topic-docno pairs out of the total 18260 topic-docno pairs.

Thirdly, although we have obtained lots of judgments after completing the previous 2 procedures, the length of many remaining documents is too long. Therefore, we ought to abridge those articles that are too lengthy for workers to read. Furthermore, there are two situations that need to be considered when abridging those lengthy articles. On one hand, for the documents that have subtitles we will do the abridging in accordance with the minimum level subtitle. For example, we can get those first paragraphs next to the subtitle H5 out of the document shown below.



On the other hand, for those documents that have not subtitles, we will take the first three and the last there paragraphs of the document, and randomly take three paragraphs in the middle section of the document. So these nine paragraphs will make up a new article, which is sent to be judged by workers.

Lastly, in order to make documents more convenient for workers to read, we need to remove all the format tags and null strings of documents. Such as those tags like H5 in the picture above.

All in all, the data processing is a very significant part of TRAT task, for it can effectively save time and cost in the later sections.

2. Part 2: Crowdsourcing Solution

2.1 Creating Jobs on CrowdFlower

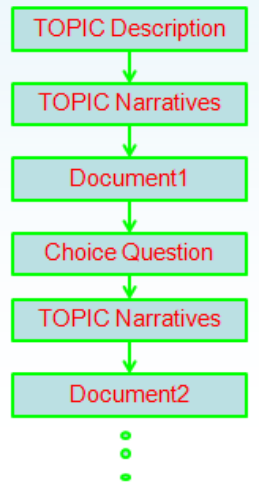
Considering its maneuverability and accessibility, we decide to carry out our Crowdsourcing solution based on the CrowdFlower Platform. And each worker is required to answer a multiple choice question after reading a topic-docno pair.

Firstly, we should take measures to supervise the quality of workers so that we can quarantee the good quality of results. The method we take to control the workers quality is setting gold. And all of the gold is created by our own. The way to create the gold is that all four members of our team need to read the same document. If all of us get the same result in a document, it can be set as gold, otherwise, it cannot be. The ratio of the gold in each Crowdsourcing task is around 10 pencent. Moreover, In order to improve the effectiveness and reliability of gold, we should ensure that there is not a significant difference between the proportion of relevant documents and the proportion of irrelevant documents in the gold. Eventually, the percentage of relevant documents in the gold is about 35%, so the irrelevant

documents make up around 65%.

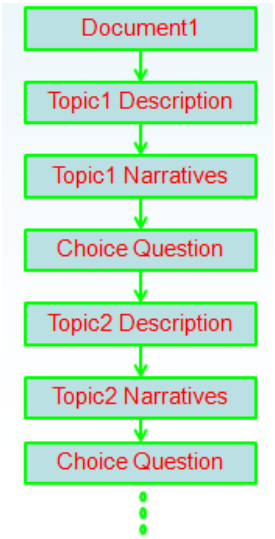
Secondly, we ought to design the pattern of our Crowdsourcing jobs. After scanning the topic-docno pairs carefully, we have found that there exist two different types of topic-docno pairs. One type is that one document only relates to one topic, and the other type is that one document relates to multiple topics. Therefore, we decide to make two kinds of Crowdsourcing jobs.

For the one-document-one-topic type, we issue the jobs separately according to the topics. That means there will be 10 jobs and each job only contains one topic but all of its corresponding documents. The pattern of this kind of Crowdsourcing job is shown as the graph below.



In this kind of job, the Topic Description of each topic will only be presented once, so that it can reduce the reading quantity of workers. But in order to guarantee the high accuracy, each document will follow one Topic Narratives. That means the Topic Narratives is presented repeatedly in each task. Certainly, every document is followed by a multiple choice question used to get the relevance judgment.

For the one-document-several topics type, we issue jobs in accordance with the number of topics that one document is corresponding to. In this kind of jobs, each job has different topics and different documents, yet, each document is corresponding to multiple topics and the amount of its corresponding topics is the same. The pattern of this kind of Crowdsourcing job is shown as the graph below.



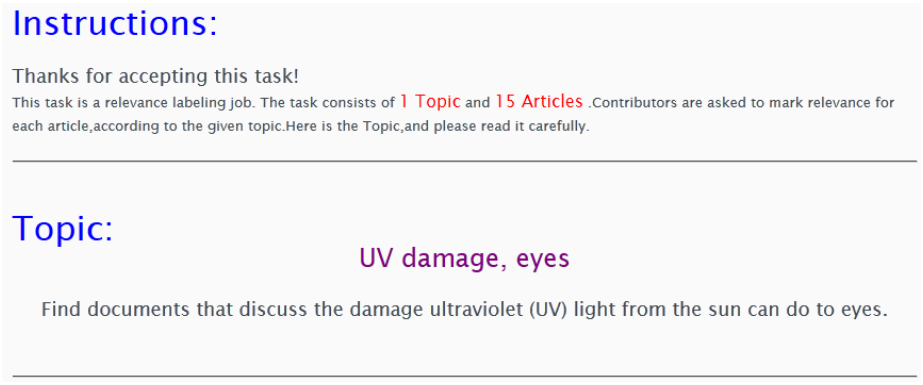
In this kind of job, although each document is corresponding to several topics, the document will only be presented once in each task. It is obvious that if we continue to use the previous model of job to this type of topic-docno pairs, workers will need to read the same document several times, which is a

really time-consuming and ineffective work. Because the documents usually are lengthier than topics and in this kind of job each worker is only required to read the same document once, which largely lessen the reading quantity of every worker.

Finally, there are twenty jobs created on the CrowdFlower platform, ten in each type of jobs.

2.2 Interface Design of Jobs

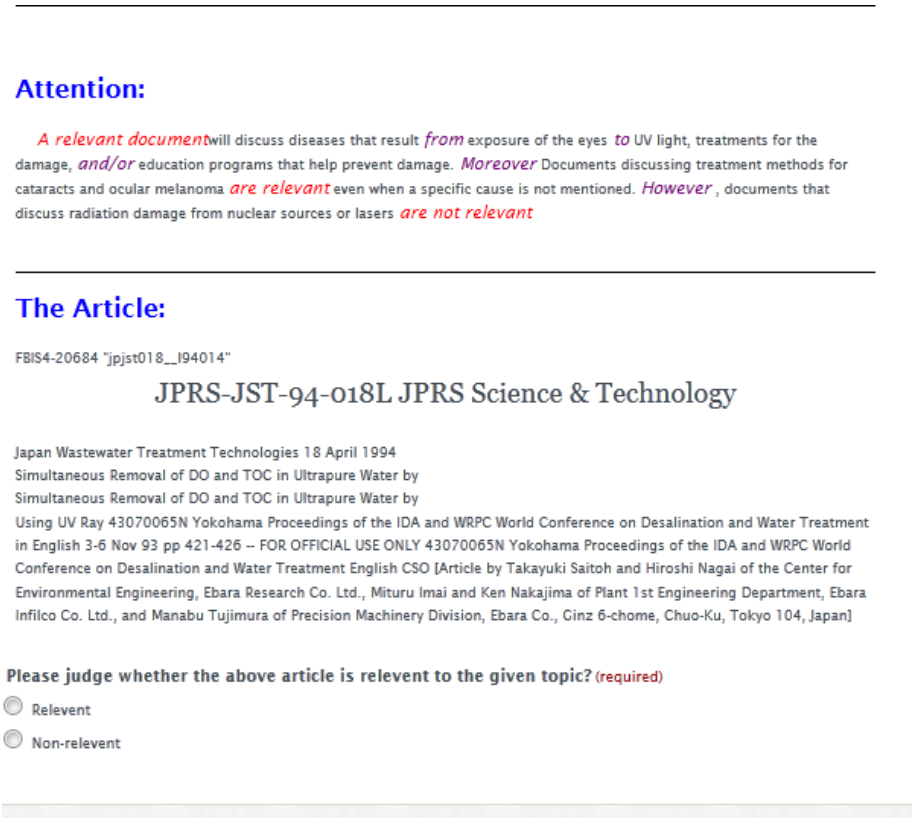
We use the CrowdFlower's own editor to edit jobs. The principle of our editing is to highlight the part that need to be paid special attentions to. Here is an example shown in the graph below.



In this picture, titles are enlarged and highlighted with different colors. Moreover, keywords and some significant conjunctions are also underlined with distinguishing colors, which makes jobs more convenient for workers to read and understand.

2.3 Jobs on CrowdFlower

Eventually, twenty jobs are created on the CrowdFlower platform. The preview of a job is shown in the picture below.



When workers log in different jobs, they are required to finish different amount of questions each time. In some jobs workers need to complete ten questions one time, while in other jobs workers need to complete fifteen questions each time. Owing to the divergent number of questions workers being required to complete in different jobs each time, we can compare the accuracy rate of workers in different jobs. Therefore, we can decide the optimum number of questions a job should present each time. However, because of the time limit, this experiment has not been carried out as we expected. However we get a preliminary result that reveals the accuracy rate of workers who are required to complete fewer questions each time is a little bit higher. However, the job in which a worker needs to complete a fewer number of questions each time is usually more time-consuming.

3. The Results

Ultimately, we successfully obtain the all 18260 judgments that have already been optimized by the CrowdFlower's own imbedded algorithm. Actually these 18260 judgments are generated from at least 54780 judgments. Because each topic-docno pair need to be judged at least three times by workers, the ultimate 18260 judgments are derived from at least 54780 judgments by the CrowdFlower's own algorithm. Among 18260 judgments, there are around 445 judgments are tagged with the relevance tag. That means, there the relevant rate of those topic-docno pairs is around 2.45 percent.