Similarity Computation in Novelty Detection

Ming-Feng Tsai, Ming-Hung Hsu, and Hsin-Hsi Chen

Department of Computer Science and Information Engineering
National Taiwan University
1, Section 4, Roosevelt Road,
Taipei, Taiwan, 106

E-mail: { mftsai, mhhsu }@nlg.csie.ntu.edu.tw;
hh_chen@csie.ntu.edu.tw

1 Introduction

The novelty track was first introduced in TREC 2002. Given a TREC topic, the goal of this task in 2004 is to locate relevant and new information from a set of documents. From the results in TREC 2002 and 2003, we realized the major challenging issue of recognizing relevant sentences is the lack of information used in similarity computation among sentences. In this year, we utilized the method based on variants of employing an information retrieval (IR) system to find relevant and novel sentences. This methodology is called IR with reference corpus, which can also be considered as an information expansion of sentences. A sentence is considered as a query of a reference corpus, and similarity between sentences is measured in terms of the weighting vectors of document lists ranked by IR systems. Basically, relevant sentences are extracted by comparing their results on a certain information retrieval system. Two sentences are regarded as similar if their corresponding returned document lists by the IR system are similar. In novelty parts, we used similar approach to extract novel sentences from the sentences of the relevant part. An effectively dynamic threshold setting approach that is based on what percentage of relevant sentences is within a relevant document is presented. In this paper, we paid attention to three points: first, how to utilize the results of an IR system to compare the similarity between sentences; second, how to filter out the redundant sentences; third, how to determine appropriate relevance and novelty threshold.

2 Procedure

The flow of our proposed method, called IR with reference corpus, is illustrated in Figure 1, which contains an IR system and a reference corpus inside. To begin with, each sentence from the given documents is treated as a query to a certain IR system that retrieves documents from the reference corpus. Then, a sentence can be translated into a vector that uses each document of whole reference corpus as one dimension and sets the relevant weight assigned by the IR system as the weight of each dimension, where the weight of a dimension is zero if the IR system does not retrieve that document. For example, a reference corpus may contain *m* documents, then each sentence of given documents can be regarded as a vector with *m* dimensions those weight is the relevant score returned by the IR system. Finally, a similarity metric, such as cosine similarity, is utilized to measure the similarity between vectors. The threshold technique is applied to the following operation, retrieval or filter. In the following, we will discuss this approach in detail.

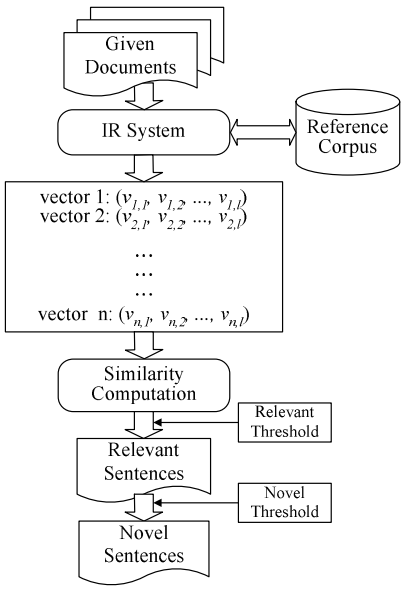

Figure 1: Flow of IR with reference corpus approach.

2.1 IR System and Reference Corpus

In the experiments, the document sets used in TREC-6 text collection (Voorhees and Harman, 1997) were considered as a reference corpus. It consists of 556,077 documents. Okapi IR system (Robertson, Walker and Beaulieu, 1998) was adopted in out experiment. In the initial experiments, Okapi was in the option of *bm25*, and it got average precision of 0.2181 on TREC-6 text collection.

In order to investigate the influence of the reference corpus, we utilized the document segmentation tool to separate each document of TREC-6 text collection into several coherent paragraphs. It totally contained 3,614,500 paragraphs after segmentation. This year, we used this segmented corpus as the reference corpus to experiment further.

2.2 Similarity Computation

The cosine similarity computation is considered appropriate for our task. The metric is shown as follows.

$$\cos(s_i, s_j) = \frac{\sum_{k=1}^{l} v_{i,k} \times v_{j,k}}{\| s_i \| \cdot \| s_j \|} \quad (1)$$

where s_i is represented as a sentence-vector ($v_{i,1}$, $v_{i,2}$, ..., $v_{i,l}$), l denotes the number of documents retrieved from the reference corpus by the IR system; s_j is another sentence-vector.

2.3 Threshold Setting

In this phase, we considered that what percentage of sentences was relevant within a document. In TREC 2002, Larkey *et al*. showed that about 5% of the sentences contained relevant materials for average topic. In TREC 2003, however, the percentage was soared by 40% of total sentences.

Therefore, we applied the collection of statistics in 2003 to relevant and novel threshold setting. We also discovered the percentage of relevant sentences became less when total number of given sentences was more. Therefore, we used logarithmic regression as follows to simulate the relationship between total number of the given sentences and percentage of the relevant sentences.

A dynamic threshold-setting model is proposed as follows. Assume normal distribution with mean μ and standard deviation σ is adopted to specify the similarity distribution of the given sentences with a topic. We compute the cosine of a topic vector T and a given sentence vector S_i ($1 \le i \le m$), where m denotes total number of the given sentences. The percentage n denotes that top n percentage of the given sentences will be reported. Similarity thresholds ($TH_{relevance}$) are determined by the percentage.

$$\mu = \frac{\sum_{i=1}^{m} \cos(T, S_i)}{m} \tag{2}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m} (\cos(T, S_i) - \mu)^2}{m}} \tag{3}$$

$$TH_{relevance} = \mu + z\sigma \tag{4}$$

$$\phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-y^2/2} dy = 1 - n \tag{5}$$

We first compute the percentage n, and then derive z by Formula (5). Finally, $TH_{relevance}$ is computed by Formula (4). Therefore, the relevance threshold is determined by the total amounts of given sentences.

In the novelty part, a threshold of novelty decision, $TH_{novelty}$, determines the degree of redundancy. If the similarity score of two sentences is larger than $TH_{novelty}$, then one of them has to be filtered out depending on their appearing order in the context. For example, if the fifth sentence is similar to the second sentence, then the fifth one will be filtered out. In this way, the redundant sentences are filtered out and only the novel sentences are kept. The remaining sentences are the result of the novelty detector. Two algorithms are proposed as follows. Assume there are r relevant sentences, $s_1, s_2, \ldots, s_r$, for topic t.

(1) Static threshold approach

Let T be a set containing novel sentences found up to know. Initially, we take $T = \{s_1\}$. For each relevant sentence s_i ($2 \le i \le r$), if there exists a sentence in T whose similarity with s_i is larger than a predefined threshold, then s_i is not a novel sentence and is removed; otherwise, s_i is kept in T.

(2) Dynamic threshold approach

Assume s_1 is a novel sentence. First compute the similarities between s_1 and s_i ($2 \le i \le r$). Then, determine the novelty threshold, $TH_{novelty}$, in the same way as $TH_{relevance}$. Filter out the top n percentage of sentences with the higher similarities with s_1. Let R be the remaining sentences. If the number of sentences in R is less than 30[1], then regard these sentences as novel sentences and stop. Otherwise, select the first sentence in R, regard it as a novel sentence and repeat the same filtering task.

[1] A sample size of at least 30 has been found to be adequate for normal distribution

3 Experiments

3.1 Finding Relevant Sentences

Given the set of documents for each topic, this part is to identify all relevant sentences from them. We first treated each given sentence as a query to the IR system, and then got a vector of document weight assigned by the IR system. Next, we applied the cosine function to measure similarity between sentences. In the part of threshold setting, we used the statistics in TREC 2002 novelty track to simulate the relation of total number of given sentences and percentage of relevant sentences. Formula 6 and Figure 2 showed the trend. Because some topics might get less percentage, we applied a parameter to multiply the percentage calculated by Formula (6) to retrieve more sentences. Take Ln-4 as an example. That means that it multiplies 4 to the calculated percentage.

$$n = -24.052Ln(x) + 196.25 \tag{6}$$

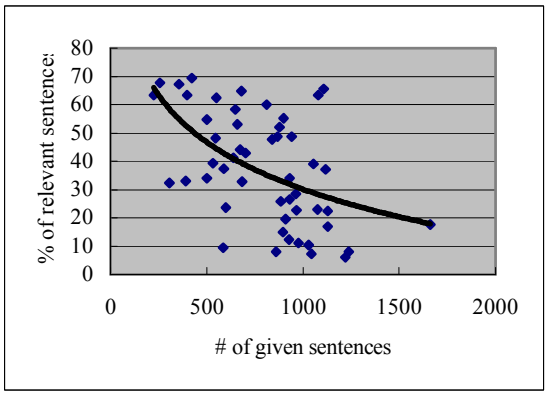

Figure 2: An illustration of logarithmic trend.

Figure 3 shows the experimental results of relevance detection in TREC 2003. These results are totally different from those in TREC 2002, because the number of qrels of relevance information in 2003 is dramatically more than that in 2002. In TREC 2002, the percentage of relevant sentences within the whole given sentences was about 5%, but in TREC 2003 some topics even had about 50 percent of relevant sentences. Therefore, our average recall gets lower since our relevance threshold is too high. That demonstrates the issue of identifying an appropriate threshold in the novelty detection is very important.

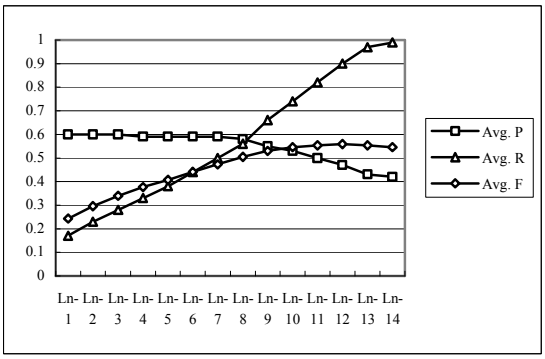

Figure 3: Experimental results of relevance detection.

3.2 Finding Novel Sentences

This part is to identify sentences that include new information among the relevant sentences. In other words, this part will filter out the redundant sentences. The key issue of finding novel sentences is how to differentiate the meaning of sentences accurately. We extended the idea, i.e., employing IR with reference corpus approach to expand a sentence, to find novel sentences. We made an experiment with two novelty threshold-setting algorithms: static and dynamic settings. In order to test this model, we used the perfect relevance results to experiment. The number of consulted paragraphs retrieved by IR system was set to 1000.

Figure 4 demonstrates the results of finding novelty with static threshold setting. When novelty threshold is 1, it does not filter out any sentences. The performance gets better as the novelty threshold is higher. Figure 5 shows the results of finding novelty with dynamic threshold setting. The result reveals that when the more percentage filtered, the worse the performance is. From these results, the performance will be better if we filter out fewer sentences. Therefore, we set the novelty threshold higher in the submitted runs to achieve better performance.

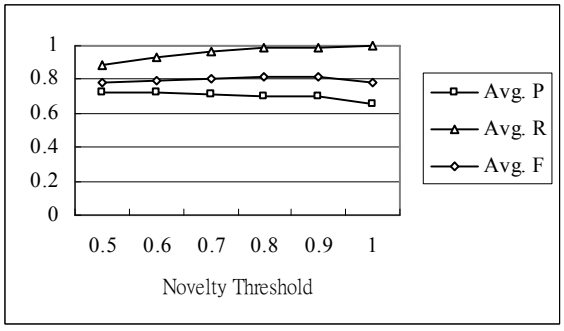

Figure 4: Experimental results of novelty detection with static threshold setting.

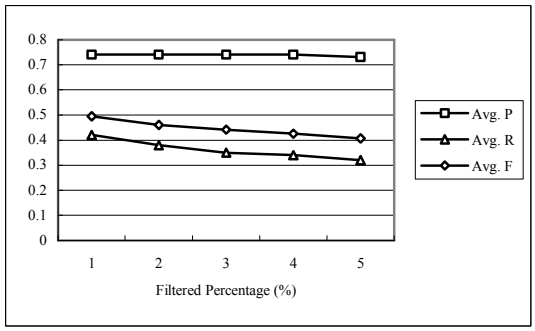

Figure 5: Experimental results of novelty detection with dynamic threshold setting.

4 Runs Submitted

4.1 Task1

Given the full set of documents for each topic, the task 1 is to identify all relevant and novel sentences. Table 1 shows the runs we submitted in task 1. In the runs, the number of consulted paragraphs was set to 1000, the dynamic relevance threshold used Ln-1 and Ln-2, and all runs only used the topic description. In the novelty part of task 1, all runs used the static threshold setting where NTU11, NTU13 and NTU15 were set to 0.8; NTU12 and NTU14 were set to 0.9.

Table 1: Submitted results in task 1.

	Relevant Detection			Novelty Detection		
	Avg P	Avg R	Avg F	Avg P	Avg R	Avg F
NTU11	0.25	0.77	0.357	0.11	0.74	0.186
NTU12	0.22	0.88	0.336	0.10	0.85	0.174
NTU13	0.26	0.69	0.344	0.12	0.65	0.184
NTU14	0.25	0.74	0.345	0.11	0.71	0.182
NTU15	0.25	0.78	0.352	0.11	0.74	0.183

4.2 Task2

Given all relevant sentences, the task 2 of Novelty Track is to identify all novel sentences. Table 2 shows the submitted results in task 2. We used two novelty algorithms to find novelty sentences. In task 2, NTU21, NTU22 and NTU23 used the static threshold setting; NTU24 and NTU25 used the globe threshold setting.

Table 2: Submitted results in task 2.

	Novelty Detection		
	Avg P	Avg R	Avg F
NTU21	0.45	0.99	0.6
NTU22	0.44	0.99	0.598
NTU23	0.44	0.99	0.598
NTU24	0.49	0.57	0.495
NTU25	0.50	0.52	0.477