PRIS at 2012 Microblog Track

Jiayue Zhang, Sijia Chen, Yue Liu, Jie Yin, Qianqian Wang, Weiran Xu, Jun Guo
School of Information and Communication Engineering
Beijing University of Posts and Telecommunications

1 Preprocessing

Take account of that most tags are keyword rich and indicate the topic of tweets directly, but there was no space between two words. So Word Segmentation was used to separate the tags by space. This time we used the former max matching algorithm. The problem is that no dictionary is appropriate. Common words dictionary is partial and Oxford Dictionary doesn't distinguish plurality. Then we made a combination of Common words dictionary, Oxford Dictionary and D.A.B(Dictionary of American Biography).But due to abbreviation and unknown words, there is still some mistakes. To avoid undesirable influence from these mistakes, we remained both the original tags and separated tags.

Two pre-processing tasks were performed in our system: RT tweets removal and non-English tweets removal. We first extracted user remarks and remove the label "RT". The "new style" re-tweets to which the HTTP crawler returned 302 were all removed. When URLs and punctuation were removed, each tweet was judged to be English or non-English with the help of an English vocabulary word list, Alan Beale's Core Vocabulary. A tweet with more than a half English words would be left and used for retrieval task while tweets with any non-English content were rejected in the query expansion task.

After pre-processing, tweets' and websites' index was built with Indri.

2 Query Expansion

In the retrieval process for the retrieval of a query, it is needed to return the most relevant microblog results. There is a kind of words is very much associated with the query, the kind of word is a synonym of the query. The synonym of the query in order to find the most important step is to find a reliable comprehensive thesaurus. This time we choose WordNet, a thesaurus. WordNet is a large lexical database of English. Nouns, verbs, adverbs and adjectives are organized by semantic relations into synonym sets (synsets), each representing one concept.

We select all of the synonyms of each word in the query for each query depending on the part of speech. The result of the synonym expansion would be added to the former result of query expansion by other means.

3 Query Formulation

1 Analyzing tweets using electric resistance network

In this task, electric resistance network was applied to query expansion. We used the tweets retrieved from indri as the corpus and calculated the effective resistance and distance metrics. (1)To building an association network from related tweets, We built an undirected weighed graph G = (V;E;w) where nodes V represent terms in

tweets, edges E represent the associated pair, and weight w on the edges measures the strength of association between two connected nodes. If two terms co–occur in one tweet, a direct link is built between them. If these two terms co–occur in n (n>0) different tweets, the weight w is n. (2) The weight wjk between node j and node k was calculated according to the electric conductance cjk defined in the original weighted graph: rjk = 1/cjk= 1/wjk. For all possible pairs, we calculated the resistances with the help of Laplacian Graph L and L = A-D, where A is the adjacency matrix and D is the degree matrix of the graph. Then the effective resistance between node vj and node vk can be calculated as follows:
rjk = L+jj + L+kk – L+jk – L+kj (3)
where L+ represents the pseudo-inverse of L.
(3) The definition of the distance between a target term t and a term set S is as follows:

$$r_{S,t} = \frac{1}{|S|} \sum_{s_i \in S} r_{s_i,t}$$

where rij is the effective resistance of node i and node j.
As for the query, we define Q and X as query term set and corpus term set respectively. And for a target term x, its normalized distance to Q can be calculated as follows:

$$r_{Q,x}^{norm} = \frac{r_{Q,x}}{\frac{1}{|X-Q|} \sum_{y \in X-Q} r_{x,y}}$$

2 Analyzing links using latent Dirichlet allocation
There are 2 kinds of links we took into consideration, links of retrieved tweets and links retrieved from indri. We used LDA to analyze these 2 kinds of links separately, and then integrate the results.
3 Design of the adaptive threshold system
As to the filtering task, we designed an adaptive threshold system to simulate actual situation. First we manually set the initial threshold for each query with the same value. Those tweets with score higher than threshold were labeled with ‘yes’ and we checked relevance of this tweet to adjust the threshold. If the yes-labeled tweet was indeed relevant, we labeled it with ‘right’; on the other hand, we labeled it with ‘wrong’. If there were too many labeled-wrong tweets/too less labeled-right tweets, we raised/lowered the threshold and adjustment was determined by the average value of labeled-right/wrong tweets, threshold before adjusting and a constant coefficient which was set manually.

4 Scoring and Ranking

Last year we used a formula [1] to evaluate the relevance of a tweet to a certain topic, which combined five factors with manually tuned parameters for weighting. However, this year we used a learning to rank algorithm which computed an optimized ranking using SEMI-SUPERVISED BIPARTITE RANKING [2]. Therefore, it can help automatically tune parameters when combining multiple evidences (e.g., text relevance, local sensitivity, URL quality, spam score, etc.) and avoid over-fitting by means of regularization.

The overall tweet score was equal to $\sum_i^N FeatureWeight_i \times FeatureScore_i$, where $\mathrm{FeatureWeight_i}$ was a configurable parameter and $\sum_\mathrm{i}^\mathrm{N} \mathrm{FeatureWeight_i} = 1$. Each $\mathrm{FeatureScore_i}$ evaluated a field-specific relevance feature. N was the total number of features. There were five fields: keywords, expanded words, entity, URL, hash-tag. For each field, we used the following features:

1. keyword_score - the percentage of keywords which a tweet contained for the target topic.
2. expandword_score - the percentage of expanded words which a tweet contained for the target topic.
3. entity_score - Since a named entity (e.g., people, products, location, etc.) in a query indicated the main part of the topic, this feature was used to highlight the contribution of target entities in the queries.
4. URL_score - A URL represented the potential information of a tweet, and the relevance score of linked text would also be added to the score.
5. hashtag_score - The hash-tags indicated the key words of tweets. But a tweet made up of all hash-tags would be spam. So this feature was used for key-word emphasizing and spam detection.

Our approach trained learning to rank with 50 iterations on the TREC Micro-blog Ad-hoc Task Evaluation Data last year. Then we ranked tweets for each topic according to the score in the descending order. Finally, we chose top n tweets in the ranking list as the relevant tweets.

[1] Yan Li, Zhenhua Zhang, Wenlong Lv. PRIS at TREC2011 Micro-blog Track. Proceedings of Text REtrieval Conference 2011.

[2] Massih Reza Amini, Tuong Vinh Truong, Cyril Goutte. A boosting algorithm for learning bipartite ranking functions with partially labeled data. Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, July 20-24, 2008, Singapore, Singapore.