HITSCIR System in NTCIR-9 Subtopic Mining Task

Wei Song, Yu Zhang, Handong Gao, Ting Liu, Sheng Li
School of Computer Science and Technology
Harbin Institute of Technology, China
{wsong, yzhang, hdgao, tliu, lisheng}@ir.hit.edu.cn

ABSTRACT

Web queries tend to have multiple user intents. Automatically identifying query intents will benefit search result navigation, search result diversity and personalized search. This paper presents the HITSCIR system in NTCIR-9 subtopic mining task. Firstly, the system collects query intent candidates from multiple resources. Secondly, Affinity Propagation algorithm is applied for clustering these query intent candidates. It could decide the number of clusters automatically. Each cluster has a representative intent candidate called exemplar. Prior preference and heuristic pair-wise preferences could be incorporated in the clustering framework. Finally, the exemplars are ranked by considering each own quality and the popularity of the clusters they represent. The NTCIR-9 evaluation results show that our system could effectively mine query intents with good relevance, diversity and readability.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval

General Terms

Algorithm, Experimentation, Performance

Keywords

Query intent, Affinity Propagation, Clustering, Query log

1. INTRODUCTION

The web queries are usually in the form of key words that leads to ambiguity. Song et al. divided queries into 3 categories [11]: ambiguous queries, which have more than one meaning; broad queries, which covers a variety of subtopics and clear queries which have a specific meaning or narrow topics. In their study, users feel difficult to distinguish between broad queries and clear queries. It indicates that a large percent of queries cover multiple subtopics (also called intents or aspects in this paper). Different users submitting the same query may have different information need, because they care about different aspects of the query. As a result, it is necessary to mine query subtopics automatically. Such techniques may provide opportunities for understanding user intents, diversifying search results and improving user experience.

Web search engines today have been complemented the search results with query aspects. For example, Bing presents a list of related searches on the left navigational bar; Google and Yahoo present them at the top or bottom of the search result page. These related queries help users to specify their search goals, formulate better queries and find satisfied results more effectively. Many existing work has been addressed for mining related queries based on search log sessions [13]. However, a query usually has too many related queries. These related queries are not enough to distinguish user intents. It is necessary to select a set of representative related queries with distinct information need. [9] combines both document click and session co-occurrence information for grouping query refinements into clusters. Though search log based methods have been widely studied, they still can't deal with new queries which never occur in past. In this condition, other resources should be used. Zeng et al. adopted term clustering techniques for grouping search result documents into clusters [14]. But the representative terms may lack of readability. Bing Liu et al. made use of html tags for extracting key concepts from web documents [8]. Dou et al. mined subtopics from anchor texts [5]. These resource based approaches can deal with any query. However, the mined topics mostly are static aspects and user behaviors are ignored.

In this paper, we aim to find query subtopics from all available resources. Generally, the task could be divided into 3 stages: query intent candidates extraction, query intent candidates clustering and query intent generation. In first stage, we extracted query intent candidates from multiple resources including search logs, anchor texts, search results and a web knowledge base. The query intent candidates are text spans which contain the original query as a substring. As these query intent candidates are either submitted by users or edited by web site editors, they have good readability. In certain degree, these query intent candidates also indicate what users care about. In the second stage, we apply a clustering algorithm for clustering these query intent candidates. The reason of clustering query intent candidates is to diversify the generated subtopics, because many query intent candidates have similar goals. By clustering, related query intent candidates will be grouped together. We view each cluster as a distinct user intent. A representative exemplar is selected to represent such intent. Considering the nature of the task, we apply Affinity Propagation algorithm [6] for clustering. This algorithm could find the exemplar naturally and decide the number of subtopics automatically. In addition, both semantic information and pair-wise preference are incorporated in this framework as a prior. The experimental results show that such additional information help for generating general and readable exemplars.

2. NTCIR SUBTOPIC MINING TASK

The NTCIR-9 subtopic mining task is motivated to encourage research on developing and evaluating algorithms to find query intents. The task is defined as follows:

- for a given query, the system should return a ranked list of subtopic strings, each of which corresponds to a specific interpretation of an ambiguous query or an aspect of a faceted query.

The Chinese document collection provided by organizer is the SogouT corpus crawled and released in 2008 by the Tsinghua-Sohu Joint Laboratory on Search Technology[1]. The Chinese query log data, called SogouQ, was also made available to participants. SogouQ was also constructed by the Tsinghua-Sohu Joint Laboratory on Search Technology[2]. It contains about 30 million clicks collected in June 2008. Its size is about 1.9 gigabytes. If the participants use external resources, it should be noted in system descriptions.

The organizer created a set of 100 Chinese topics from the June 2008 query log of Sogou. The time periods were chosen to coincide with the crawl periods of the document collections.

Having created the topic sets, the intents for each topic were identified. The top 20 subtopic strings from each run were pooled. Then, using a dedicated graphical interface, 10 assessors manually clustered subtopics with similar search intents. The assessors also provided a short description for each intent. Subsequently, the assessors vote whether each intent is important or not for each topic. Thus the number of votes for each intent could vary between 0 and 10. The probability of each intent given a query was computed according the the votes.

3. PROPOSED APPROACH

3.1 Main Idea

The task aims to find subtopics (we may also use *intents* in following sessions which refer to subtopics as well.) for a given query. In our opinion, there are 4 main factors that should be considered.

- relevance: A subtopic must truly discuss a topic about the query.
- diversity: The mined subtopics should be as complete as possible to cover all related aspects of the query.
- readability: The mined subtopics should be easy to understand and close to the way users express as much as possible.
- general: A more general concept or description is preferred to a specific one when they are talking about the same topic.

The task judgement is mainly based on relevance and diversity but the other two factors may also affect accessors' judgement. According to the above standards, we set up our framework. Our approach is based on user generated content such as query logs, anchor texts, web pages and web knowledge base. All the content is generated by human so that it ensures that the subtopics are readable. We constrain the subtopic candidates must contain key words in original query that attempts to ensure the relevance. Clustering method is used for grouping intent candidates into clusters. This is to reduce the duplicates and help for diversifying the subtopics. Semantic resources such as thesaurus [1] are used to measure the general level of concepts. Figure 1 illustrates the basic framework of our approach.

[1]http://www.sogou.com/labs/dl/t.html
[2]http://www.sogou.com/labs/dl/q.html

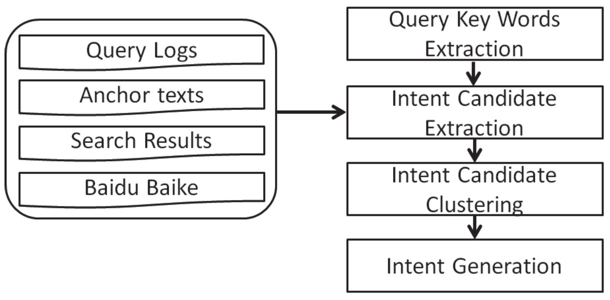
Query Logs
Anchor texts
Search Results
Baidu Baike
Query Key Words Extraction
Intent Candidate Extraction
Intent Candidate Clustering
Intent Generation

Figure 1: The framework of proposed approach.

3.2 Notions

We first introduce some notions which would be used in following sections. Given the query q, we define:

Query Keywords. Since there are queries in the form of natural language, we only use the key words in a query to represent it. We use the open source LTP package [3] for word segmentation and part of speech tagging. Heuristically, we remove all stop words and extract noun phrases and verb phrases from original query. For simplicity and without ambiguity, we still use $\boldsymbol{q}$ to represent the query key words.

Intent Phrase. An intent phrase is a phrase that cooccurs with the original query key words in certain context, noted as $\boldsymbol{p}$. For example, suppose the original query is "apple", and a query in query log is "apple mp3 price", as a result "mp3 price" is viewed as an intent phrase for query "apple". For this case, the queries in query log are considered as the context.

Intent Words. An intent phrase may contain multiple words. Each of these words is called a query intent word, noted as $\boldsymbol{w}$.

Intent Candidate. An intent candidate is a text span containing both the query key words and intent phrases, noted as $\boldsymbol{c}$ which is a combination of $\boldsymbol{q}$ and $\boldsymbol{p}$.

3.3 Extracting Intent Candidates

In this paper, we use multiple sources of context for extracting query intent candidates. These sources include query log, anchor text, text spans in search results and a knowledge base Baidu Baike[3].

Extraction from query log. Search log contains all past queries users have submitted to a search engine. During interaction with search engine, users may formulate more specific queries in order to find desired information quickly. Web queries reflect user intents in certain degree. We mainly exploit two kinds of queries:

- queries containing original query key words as a substring

[3]http://baike.baidu.com/

Table 1: A summarization of the types of test queries

Types	Entity	Description	Task	Event
Cases	金素妍 减肥粥	越南旅游 糖尿病症状	QQ下载 祖玛游戏下载	宋美龄去世 日俄战争
Count	70	23	5	2

- query reformulations that occur following the original query within 15 mins.

By excluding the original query key words from these queries, intent phrases are extracted.

Extraction from anchor text. The anchor text is a highlighted and clickable text span containing a hyperlink to a web page. The anchor texts are all edited by human. They are usually with good readability. We adopt similar method to extract query intent phrases from anchor texts. In detail, we used all anchor texts in the SogouT corpus and extracted the anchor texts containing original query key words. Some anchor texts are sentences which contain too much noisy. So we discard the ones longer than a certain length (10 words in our experiment).

For efficiency, we indexed the query logs and anchor texts in SogouT using Intri toolkit[12]. When a query comes, we can search the indexed content, and select the top ones to ensure the quality.

Extraction from search results. Previous work has paid much attention for mining query reformulations from search sessions and search logs. However, these approaches only work for old queries that have been seen before. For new queries that never occur in past query logs, search results are one of the main resources for mining. We used Indri to index the SogouT corpus. For each query, we collected relevant documents by searching SogouT using Intri with default setup. The search results doesn't provide snippets, so we extracted query intent phrases according to the html tags in documents largely based on [8].

Extraction from Knowledge Base We analyze the query types of test queries, as show in table 1. We manually classify the queries into 4 types: Entity, Description, Task or Event. An entity query is defined as the query itself is an entity. For other types of queries, most of them still contain entities, however, they have already talked about some aspects of the queries. In fact, the subtopics form a hierarchical structure. We can see a large ratio of queries are entity queries. In fact, we also have sampled another 1000 queries from SogouQ query logs. And similar conclusion is got that more than 42% queries are entity queries. As a result, mining subtopics for entity queries is a core part for mining query subtopics.

In this case, web knowledge base could be made use of because many entity related information is contained in such knowledge base. In detail, we use the query key words as a query to search in Baidu Baike, if it matches one of the entry in Baidu Baike or matches one of the subheading of an entry, we extract the first level subheadings of these entry as the query intent candidates. However, such method can't work for all entity queries. It can't serve for non entity query, new entities and complex queries. Besides, the extracted query intent candidates tend to be attributes of entities but not aligned to user information need.

For all intent phrases mined from different sources, we label the part of speech of every word in intent phrases. Only Norn and verb words are selected as the intent phrase.

3.4 Ranking Intent Candidates

The direct way to get query intents is to rank the mined intent phrases and return the top ones as the intent. We use the frequency to sort these intent phrases. First, we segment all the intent phrases into words. Then for every word, we weight it according to the LPF (log phrase frequency) score among across multiple sources. The LPF score of a term w is computed as $LPF(w) = log(pf(w) + 1)$, where $pf(w)$ refers to the number of intent phrases containing word w. The assumption is that the more times a word occurs in query intent phrases, the more important this word is for the original query. Finally, we rank query intent candidates $\boldsymbol{c}$ according to

$$Score(c) \equiv Score(p) = \frac{1}{|p|}\sum_{w\in p} LPF(w) + \max_{w\in p} LPF(w) \tag{1}$$

Where $\boldsymbol{p}$ refers to a query intent phrase and $|\boldsymbol{p}|$ indicates the number of words contained in $\boldsymbol{p}$. Since an intent candidate $\boldsymbol{c}$ could be represented by $\{\boldsymbol{q}, \boldsymbol{p}\}$, and $\boldsymbol{q}$ is the same for all candidates so that scoring $\boldsymbol{c}$ is equivalent to scoring $\boldsymbol{p}$. By this way, we could get a list of ranked intent candidates. Considering the issue of efficiency, we only used top query intent phrases as the input for next phase.

3.5 Clustering Intent Candidates

Though the ranked intent phrases emphasize the most important query intents, there are some problems. First, some intent phrases have similar semantics which results in redundance. Second, there is no overview about the query intents, such as how many aspects a query may have. As a result, we apply clustering algorithm for organizing the mined intent phrases. The task could be summarized as clustering these query intent phrases into clusters, for each cluster select one best intent phrase to represent this cluster. The selected representative intent phrases are used as query intents.

3.5.1 Affinity Propagation Algorithm

Considering the characteristic of the task, we apply Affinity Propagation (AP) [6] as the clustering method. AP is an exemplar-based clustering method. It takes as input similarities between data points and outputs a set of data points (exemplars) that best represent the data, and assign each non-exemplar point to its most appropriate exemplar, thereby the data points are grouped into clusters.

In detail, this algorithm has two initial inputs. First is a similarity matrix M. $M_{i,j}$ represents how points i prefers point j to be the exemplar. Second is a preference list for each point. It represents how likely the point should be an exemplar. Then the real-value messages are exchanged between data points until a high quality set of exemplars and clusters emerge. These exemplars are representative to their corresponding clusters. AP was reported to be able to cluster with lower error compared with other state of the art clustering approaches.

The reason we use AP algorithm for clustering is because it has several advantages: First, it decides the number of topics automatically. It is obvious that it is difficult to tell how many subtopics a query should have. Second, after grouping intent candidates into clusters, it is necessary to select one best to represent the whole cluster. The AP algorithm is an exemplar based method, points within the same cluster prefer the same exemplar, so that it decides the representative one naturally. The third, the AP algorithm allows asymmetric similarities between data points that provides us more convenience to integrate some semantic information to bias the exemplar selection.

3.5.2 Similarities between Intent Candidates

We adopt three main strategies for computing similarity between query intent candidates. Lexical similarity is based on the surface lexical features of strings. Search log features include co-occurrence in query logs and user sessions. Due to the data we use is not big enough, these features are sparse. Semantic similarities are computed using knowledge resources. Here, we use Hownet and a synonym thesaurus. For simplicity, we represent each query intent candidate $c = \{q, p\}$, where c is an intent candidate for a given query q and p refers to an intent phrase of q.

(1) Lexical Similarity

Two strategies are used for measuring lexical similarity. Similar methods were also used in [2] for query clustering.

Similarity based on Keywords. The similarity function is defined as

$$Sim_{keywords}(c_1, c_2) = \frac{KN(p_1, p_2)}{Min(kn(p_1), kn(p_2))} \tag{2}$$

Where kn(.) is the number of keywords in an intent phrase. $KN(p_1, p_2)$ is the number of common keywords of two intent phrases.

Similarity based on Edit-Distance. The similarity function is defined as

$$Sim_{edit\_dist}(c_1, c_2) = 1 - \frac{edit\_distance(c_1, c_2)}{Max(wn(c_1), wn(c_2))} \tag{3}$$

Where wn(.) is the number of words in a query variation. *edit_distance* is a measure based on the number of edit operations (insertion, deletion, or substitution of a word) necessary to unify two strings [7].

(2) Search log based Similarity

$$Sim_{co-click}(c_1, c_2) = \frac{CL(c_1, c_2)}{Max(cl(c_1), cl(c_2))} \tag{4}$$

Where $cl(.)$ is the number of clicked documents corresponding for a query in query log, and $CL(c_1, c_2)$ refers to the number of common clicked documents for query c_1 and c_2.

$$Sim_{co-session}(c_1, c_2) = \frac{CS(c_1, c_2)}{Max(cs(c_1), cs(c_2))} \tag{5}$$

Where $cs.$ is the number of occurrence of a query, and $CS(c_1, c_2)$ is the times that two queries co-occurs in the same session.

(3) Semantic Similarity

We use two semantic resources to compute semantic similarities. First is Hownet which is a structured Chinese semantic dictionary [4]. The API of Hownet provides an interface *HowNet_Get_Concept_Similarity* to compute the semantic similarity between two concepts. We define the semantic similarity between two phrases as:

$$PhraseSim_{hownet}(p_1, p_2) = \frac{1}{|p_1|}\sum_{w \in p_1} \frac{1}{|p_2|}\sum_{w' \in p_2} HowNet\_Get\_Concept\_Similarity(w, w') \tag{6}$$

Then the semantic similarity between two query intent candidates is defined as:

$$CandidateSim_{hownet}(c_1, c_2) = \frac{PhraseSim_{hownet}(p_1, p_2) + PhraseSim_{hownet}(p_2, p_1)}{2} \tag{7}$$

We also use a Chinese synonym thesaurus[1] which has a tree structure. The concepts locate in higher levels are hypernyms of their children. Heuristically, if concept v and concept u are siblings at the same branch of the tree structure, the similarity between them is set to 1.

3.6 Algorithm Initiation

As introduced before, AP has two inputs: the preference and similarity matrix.

Initiating Preference. We can set the preference for all data points uniformly. We use the mean value of the similarity matrix as the preference value for all points, noted as **UniformPreference**. Also, if we emphasize the preference for some data points that meas these points have a higher prior to be an exemplar, we assign a higher preference to these points by multiple a coefficient larger than 1, noted as **HighlightPreference**. Here, we give intent candidates from Baidu Baike a higher preference where the coefficient is set to 1.5.

Initiating Similar Matrix. AP supports asymmetric similarities. Instead of using symmetric similarities (**Symm**), we can integrate pair-wise preferences, noted as **Asymm**. For a pair of intent candidates $c_i = \{q, p_i\}$ and $c_j = \{q, p_j\}$, we consider two types of preferences:

- if p_i and p_j have common text, and $|p_i| < |p_j|$, then c_i is preferred to c_j. To achieve this, we change equation 2 to

$$Similarity_{keywords}(c_i, c_j) = \frac{KN(p_i, p_j)}{kn(p_j)} \tag{8}$$

- if p_i is the hypernym of p_j, then c_i is preferred to c_j. To achieve this, we change the thesaurus based similarity heuristically by setting $M_{j,i} = 1.5 \times M_{i,j}$

3.7 Generating Query Intents

After achieving the clusters, we use the exemplar of each cluster as a subtopic. We rank these subtopics according to the importance of the cluster. We make two assumptions.

(i) The more important the intent candidate is, the more important the subtopic is.

(ii) The more important intent candidates the cluster contains, the more important the subtopic is.

Based on the above assumptions, given cluster $G = \{c_1, ..., c_n\}$ and its exemplar e, we score G according to

$$Score_{cluster}(G) = \lambda * Score(e) + (1-\lambda) * \sum_{c_i \in G} Score(c_i) \quad (9)$$

$Score(.)$ refers to equation 1 which indicates the importance of a candidate. λ is a parameter used to balance the two types of effects. The second factor also reveals users voting, since popular intents will have higher scores.

4. EXPERIMENTS

4.1 Parameter Setup

There are several parameters to be set.

In section 3.4, we got a list of intent candidates. In our experiement, we only used top 100 intent candidates for clustering. Since the candidates extracted from Baidu Baike are usually with high quality, we directly put these candidates into the final list. The intent candidates from other resources are ranked according to 1 and the top ones are added to the final list until its size reached 100.

We adopt several methods for computing similarities between query intent candidates. We used linear interpolation to combine them together. We assigned uniform weights to each similarity methods except that we gave higher weights to semantic similarities, since semantic similarities we used are knowledge based with higher confidence comparing to other methods.

4.2 Runs

We submitted 5 runs in all.

(1) We extracted intent candidates from all resources introduced in 3.3. According to section 3.6, the initiation settings are **Asymm** and **HighlightPreference**.

(2) We extracted intent candidates from SogouQ and SogouT, and ranked them according to equation 1. Unfortunately, when evaluating, due to we didn't add original query information, most of the candidates were judged as irrelevant. As a result, this run achieved a very low s-core. We think it is largely underestimated. In following sessions, we will not discuss this run.

(3) Similar to (1). But we did not use "Baidu Baike" to extract intent candidates. The initiation settings are **Asymm** and **HighlightPreference**.

(4) Similar to (3). However the settings are changed to **Symm** and **HighlightPreference**.

(5) Similar to (1). But the settings are changed to **Asymm** and **UniformPreference**

4.3 Evaluation Metrics

D♯-nDCG is used as primary evaluation metric. D♯-nDCG is a linear combination of intent recall (or "I-rec", which measures diversity) and D-nDCG (which measures overall relevance across intents) [10].

According the official report, D♯-nDCG is a simple average of I-rec and D-nDCG here. The gain values for the per-intent graded relevance were set linearly: 1, 2, 3 and 4 for L1, L2, L3 and L4,respectively. The measurement depths (i.e. number of top ranked items to be evaluated) were set to l = 10, 20 and 30.

4.4 Experimental Results

Table 2, table 3 and table 4 illustrate the experimental results of our runs for top 10, top20 and top 30 intent candidates respectively. We have some observations.

4.4.1 The role of web knowledge base

The run1 and run5 using Baidu Baike for extracting query intent candidates outperform run3 and run4 which didn't use Baidu Baike. It indicates that when web knowledge base is available for a query, it is good resource to infer query's subtopics. Especially it improves the relevance (D-nDCG). It is easy to explain. Though candidates extracted from Baidu Baike are unable to cover all user intents about corresponding query, it is mostly accurate enough, as the knowledge bases are maintained by human. Since a large ratio of web queries are entity queries, subtopic mining task is especially suitable for entity queries. The mined subtopics perhaps benefit search result personalization or diversity.

The run3 and run4 still have acceptable performance. They don't depend on web knowledge base, so could be applied to any types of queries. In this case, query candidate extraction is very important. According to our observations, the query candidates extracted from query logs and anchor texts with higher quality compared to the ones extracted from search results. One main reason is that we didn't use commercial search engines to get search results but used Indri toolkit without optimizing parameters for SogouT. The returned search results by Indri may be not good enough. The extraction from full texts of search results further involved noises.

Table 2: The performance of 4 runs measured using top 10 results

Runs	I-rec@10	D-nDCG@10	D♯-nDCG@10
Run1	0.4854	0.6453	0.5653
Run3	0.4807	0.6308	0.5558
Run4	0.4738	0.6291	0.5514
Run5	0.4936	0.6449	0.5693

Table 3: The performance of 4 runs measured using top 20 results

Runs	I-rec@10	D-nDCG@10	D♯-nDCG@10
Run1	0.6316	0.6213	0.6264
Run3	0.6235	0.6027	0.6131
Run4	0.6235	0.6027	0.6131
Run5	0.6421	0.6180	0.6300

Table 4: The performance of 4 runs measured using top 30 results

Runs	I-rec@10	D-nDCG@10	D♯-nDCG@10
Run1	0.6634	0.5531	0.6083
Run3	0.6479	0.5182	0.5830
Run4	0.6479	0.5182	0.5830
Run5	0.6730	0.5529	0.6130

4.4.2 The role of initial preference for AP

The run1 and run5 are similar. When applying AP, run1 sets a higher preference for intent candidates from Baidu

Baike. While run5 used the uniform preference value for all intent candidates. We can see that the relevance score D-nDCG slightly improves when using distinct preference. But the diversity score I-rec decreases. It shows that (1) changing initial preferences truly affects the exemplar selection. The ones with higher preference are more likely to be selected as the exemplar of each cluster. (2) the results extracted from web knowledge base are more accurate. (3) Emphasizing knowledge base effect may hurt the diversity.

It is easy to understand that emphasizing query candidates from Baidu Baike leads the improvement of relevance. On the other hand, the language used in Baidu Baike is more formal compared to the language used in other resources. For example, "蟑螂防治 (cockroach control)" is a query subtopic candidate extracted from Baidu Baike for query "蟑螂 (cockroach)". However, the word "防治 (control)" doesn't appear frequently in query logs. Instead, users are used to using "杀死 (kill)" or "消灭 (eliminate)" to describe the same intent. As a result, these ones with high frequency in query logs and the one from the knowledge base tend to be ranked on the top even though they refer to similar intent. That is why the diversity score decreases.

4.4.3 The role of asymmetric similarity for AP

The run3 and run4 are quite similar, but differ in similarity matrix. The run3 uses asymmetric similarities. The results show that when evaluating the top 10 query subtopics, the algorithm using asymmetric similarity performs better on both relevance and diversity. However, for top 20 and top 30 candidates, the results are almost the same.

4.4.4 Number of mined subtopics

One advantage of AP is that it could decide the number of clusters automatically. The number of query subtopics also is a important factor for subtopic mining algorithms. It indicates how an algorithm could group similar intents together and separate different intents. The average number of subtopics labeled by the labelers are 16. When we use median value of the entries in similarity matrix, the average number of subtopics is 13. When we use mean value, the average number of subtopics is 23. Both are close to the labelers' judgement. The number of subtopics also relate to the number of query intent candidates. For some queries, there are not enough candidates in resources that results in less clusters.

5. CONCLUSIONS

In this paper, we present our system on NTCIR9 subtopic mining task. We set the subtopic mining task in a clustering framework. Multiple resources are used for extracting query intent candidates. We find that the Affinity Propagation algorithm is very suitable for this task. The semantic knowledge and pair-wise preference could be incorporated in this framework. We analyzed several factors affecting the system performance. The results are encouraging. In future, we plan to further improve our system by paying more attention on search results with a better retrieval system.

Acknowledgments

This research is supported by NSFC under Grant No. 60736044, 61073126, by HIT.KLOF.2009020, by 2011ZX01042-001-001.

6. REFERENCES

[1] 梅家驹等编. *同义词词林 第二版*. 上海辞书出版社, 上海, 中国, 1996.

[2] Query clustering using user logs. *ACM Trans. Inf. Syst.*, 20:59–81, January 2002.

[3] W. Che, Z. Li, and T. Liu. Ltp: a chinese language technology platform. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, COLING '10, pages 13–16, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[4] Z. Dong and Q. Dong. *Hownet And the Computation of Meaning*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2006.

[5] Z. Dou, S. Hu, K. Chen, R. Song, and J.-R. Wen. Multi-dimensional search result diversification. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, pages 475–484, New York, NY, USA, 2011. ACM.

[6] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[7] D. GUSFIELD. Inexact matching, sequence alignment, and dynamic programming. Technical report, Cambridge University Press, 1997.

[8] B. Liu, C. W. Chin, and H. T. Ng. Mining topic-specific concepts and definitions on the web. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 251–260, New York, NY, USA, 2003. ACM.

[9] E. Sadikov, J. Madhavan, L. Wang, and A. Halevy. Clustering query refinements by user intent. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 841–850, New York, NY, USA, 2010. ACM.

[10] T. Sakai and R. Song. Evaluating diversified search results using per-intent graded relevance. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, pages 1043–1052, New York, NY, USA, 2011. ACM.

[11] R. Song, Z. Luo, J.-R. Wen, Y. Yu, and H.-W. Hon. Identifying ambiguous queries in web search. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 1169–1170, New York, NY, USA, 2007. ACM.

[12] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language-model based search engine for complex queries (extended version). IR 407, University of Massachusetts, 2005.

[13] X. Wang, D. Chakrabarti, and K. Punera. Mining broad latent query aspects from search sessions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 867–876, New York, NY, USA, 2009. ACM.

[14] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma. Learning to cluster web search results. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 210–217, New York, NY, USA, 2004. ACM.