Learning to Mine Query Subtopics from Query Log

Zhenzhong Zhang, **Le Sun**, **Xianpei Han**
Institute of Software, Chinese Academy of Sciences, Beijing, China
{zhenzhong, sunle, xianpei}@nfs.iscas.ac.cn

Abstract

Many queries in web search are ambiguous or multifaceted. Identifying the major senses or facets of queries is very important for web search. In this paper, we represent the major senses or facets of queries as subtopics and refer to indentifying senses or facets of queries as query subtopic mining, where query subtopic are represented as a number of clusters of queries. Then the challenges of query subtopic mining are how to measure the similarity between queries and group them semantically. This paper proposes an approach for mining subtopics from query log, which jointly learns a similarity measure and groups queries by explicitly modeling the structure among them. Compared with previous approaches using manually defined similarity measures, our approach produces more desirable query subtopics by learning a similarity measure. Experimental results on real queries collected from a search engine log confirm the effectiveness of the proposed approach in mining query subtopics.

1 Introduction

Understanding the search intents of queries is essential for satisfying users' information needs and is very important for many search tasks such as personalized search, query suggestion, and search result presentation. However, it is not a trivial task because the underlying intents of the same query may be different for different users. Two well-known types of such queries are ambiguous queries and multifaceted queries. For example, the ambiguous query 'michael jordon' may refer to a basketball player or a professor of statistics in Berkeley. The multifaceted query 'harry potter' may refer to different search intents such as films, books, or games and so on.

Many approaches have been proposed to identify the search intents of a query which are represented by search goals, topics, or subtopics. For example, Broder (2002) classified query intents into three search goals: informational, navigational, and transactional. Broder et al. (2007) and Li et al. (2005) represented query intents by topics. Clarke et al. (2009) represented query intents by subtopics which denote different senses or multiple facets of queries.

Previous work on query subtopic mining is mostly based on clustering framework by manually defining a similarity measure with few factors. Hu et al. (2012) employed an agglomerative clustering algorithm with a similarity measure combining string similarities, click similarities, and keyword similarities linearly. Wang et al. (2013) applied affinity propagation algorithm (Frey and Dueck, 2009) with a sense-based similarity. Tsukuda et al. (2013) used a hierarchical clustering algorithm with the similarity measure based on search results.

In this paper, we argue that the similarity between queries is affected by many different factors and it could produce more desirable query subtopics by learning a similarity measure. To learn a similarity measure for query subtopic mining, a natural approach is to use a binary classifier, that is, the classifier targets pairs of queries and makes predictions about whether they belong to the same subtopic. However, because such pairwise classifiers assume that pairs are independent, they might make inconsistent predictions: e.g., predicting queries q_i and q_j, q_j and q_k to belong to the same subtopic, but q_i and q_k to belong to different subtopics. For example, given three queries, 'luxury car', 'sport car' and 'XJ sport', for the query 'jaguar', a lexicon-similarity-based classifier is easy to learn that 'luxury car' and 'sport car', and 'sport car' and 'XJ sport' belong to the same subtopic; but difficult to learn that 'luxury car' and 'XJ sport' belong to the same subtopic. From this example, we can see that a learner should exploit these transitive dependencies among queries to learn a more effective similarity measure. Hence, in this paper, our first contribution is that we learn a similarity measure by explicitly modeling the dependencies among queries in the same subtopic. The second contribution is that we analyze the performance of the proposed approach with different dependencies among queries. The third contribution is that we conduct experiments on

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers), pages 341–345,
Beijing, China, July 26-31, 2015.

real-world data and the experimental results confirm the effectiveness of the proposed approach in mining query subtopics.

2 Learning to Mine Query Subtopics

In this section, we present our approach in details. First, we collect queries as subtopic candidates from query log using a heuristic method. Then, we learn a similarity measure to mine query subtopics from these candidates.

2.1 Collecting Subtopic Candidates from Query Log

In web search, users usually add additional words to clarify the underlying intents of a query (Hu et al., 2012). For example, if the ambiguous query 'jaguar' does not satisfy a user's information need, he/she may submit 'jaguar sport car' as an expanded query to specify the subtopic. Therefore, for a given query q, we collect its reformulations with additional words from query log as query subtopic candidates, e.g., we collect {'jaguar sports car', 'jaguar XJ sport', 'jaguar diet', …} for query 'jaguar'. We say query q' is a subtopic candidate of q if (1) q' is superset of q (e.g. q' = 'jaguar sports car' and q = 'jaguar'), and (2) q' occurred at least five times in query log. In this way, we collect a series of subtopic candidates for each query. Many subtopic candidates, however, belong to the same subtopic, e.g., 'jaguar sports car' and 'jaguar XJ sport'. Thus, to obtain the subtopics of a query, we need to group its subtopic candidates into clusters, each of which corresponds to an individual subtopic.

2.2 Mining Query Subtopics

As we described above, we need to group the subtopic candidates of a query into clusters to obtain its subtopics. The key to producing desirable subtopics is how to measure the similarity between subtopic candidates. In this paper, we learn a similarity measure by exploiting the dependencies among subtopic candidates in the same subtopic.

We represent each pair of subtopic candidates q_i and q_j as a feature vector $\phi(q_i, q_j)$, each dimension of which describes a factor. The similarity measure Sim_w parameterized by w is defined as $Sim_w(q_i, q_j) = w^T \cdot \phi(q_i, q_j)$, which maps pairs of subtopic candidates to a real number indicating how similar the pair is: positive for similar and negative for dissimilar. As argued in the introduction, the dependencies among subtopic candidates within the same subtopic are useful for learning an effective similarity measure. We denote the dependencies among subtopic candidates as a graph h, whose vertices are subtopic candidates and edges connect two vertices belonging to the same subtopic. In this paper, we employ two different graphs. The first one is the all-connection structure, where all subtopic candidates belonging to the same subtopic associate with each other. Figure 1 gives an example of the all-connection structure. The second one is the strong-connection structure, where each subtopic candidate only associates with its 'most similar' subtopic candidate within the same subtopic. Figure 2 gives an example.

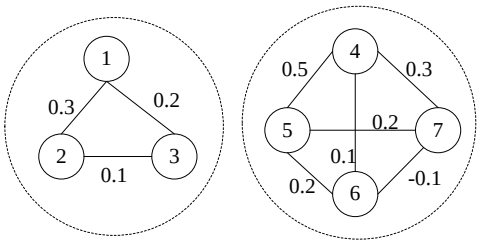

Figure 1. An example of the all-connection structure. The dashed circles denote the subtopics. The subtopic candidates (small solid circles) in the same dashed circle belong to the same subtopic. The weights indicate how similar the pair of two vertices is.

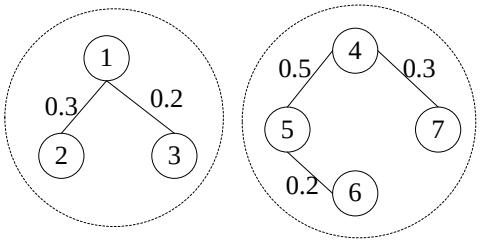

Figure 2. An example of the strong-connection structure.

Formally, we denote the set of subtopic candidates for a given query q as $S = \{q_1, q_2, \ldots, q_N\}$. The label y is a partition of the N subtopic candidates into subtopic clusters. h is the corresponding graph that is consistent with y. h is consistent with a clustering y if every cluster in y is a connected component in h, and there are no edges in h that connect two distinct clusters in y. Given S, our approach makes predictions by maximizing the sum of similarities for subtopic candidate pairs that are adjacent in h, that is,

$$\arg\max_{(y,h)\in Y\times H} \sum_{(i,j)\in h} Sim_w(q_i, q_j) = \arg\max_{(y,h)\in Y\times H} \sum_{(i,j)\in h} w^T \cdot \phi(q_i, q_j) \quad (1)$$

where Y and H are the sets of possible y and h respectively. (i, j) $\in h$ denotes q_i and q_j are directly connected in h.

To predict a partition y with the all-connection structure, we use the algorithm in (Bansal et al., 2002) with the objective function Eq (1). To predict a partition y with the strong-connection structure, we run Kruskal's algorithm on h and each tree corresponds to a subtopic, as shown in Algorithm 1.

Algorithm 1: Mining Query Subtopic with Strong-connection Structure

Input: the set of query subtopic candidates $S = \{q_1, q_2, \ldots, q_N\}$, feature vectors $\phi(q_i, q_j)$ ($1 \le i, j \le N$, $i \ne j$) and the weight w
Output: the partition y

```
//search for the strong-connection structure h, MST-KRUSKAL(G) denotes the Minimum Spanning Tree algorithm- Kruskal's algorithm

for i = 1…N-1 do
    for j = i+1…N do
        sim = w^T· φ(q_i, q_j);
        G(i, j)=−sim;
    end
end
h' = MST-KRUSKAL(G);
for i = 1…N-1 do
    for j = i+1…N do
        if h'(i, j)<0 then
          h(i, j) = 1;
        end
    end
end

// construct the partition y
t = 0;
y(1)=0;
for i = 2…N do
     j = 1;
    while j ≤ i-1 do
        if h(j, i) = 1 then
          y(i)= y(j);
          break;
        end
        j = j+1;
    end
    if j ≥ i then
      t = t + 1;
      y(i) = t;
    end
end
return y
```

2.3 Solving the Proposed Approach

For a given set of subtopic candidates with annotated subtopics, $\{(S_n, y_n)\}$ ($1 \le n \le N$), we need to estimate the optimal weight w. Empirically, the optimal weight w should minimize the error between the predicted partition y' and the true partition y, and it should also have a good generalization capability. Therefore, it is learnt by solving the following optimization problem (Yu and Joachims, 2009):

$$\begin{aligned} &\min_{w,\xi} \frac{1}{2} \| w \|^2 + C \sum_{n=1}^{N} \xi_n \\ &\text{s.t. } \forall n,\ \max_{h \in H} w^T \cdot \sum_{(i,j) \in h} \phi(q_i, q_j) \ge \\ &\qquad \max_{(y',h') \in Y \times H} [w^T \cdot \sum_{(i,j) \in h'} \phi(q_i, q_j) + \Delta(y_n, y', h')] - \xi_n \end{aligned} \quad (2)$$

where $\Delta(y_n, y', h')$ indicates a loss between a true partition y_n and the predicted partition y' specified by h', ξ_n ($1 \le n \le N$) is a set of slack variables to allow errors in the training data, and C controls the trade-off between empirical loss and model complexity.

Intuitively, the loss function $\Delta(y_n, y', h')$ should satisfy that $\Delta(y_n, y', h') = 0$ if $y_n = y'$, and rises as y_n and y' become more dissimilar. Because the all-connection structure is observable in the training data while the strong-connection structure is hidden, we define different loss functions for them. For the all-connection structure, we define the loss function as,

$$\Delta(y_n, y', h') = 10 \frac{D}{T} \quad (3)$$

where T is the total number of pairs of subtopic candidates in the set partitioned by y_n and y', and D is the total number of pairs where y_n and y' disagree about their cluster membership.

Since the strong-connection structure h_n for y_n is hidden in the training data, we cannot measure the loss between (y_n, h_n) and (y', h'). According to (Yu and Joachims, 2009), we define the loss function based on the inferred structure h' as,

$$\Delta(y_n, y', h') = n(y_n) - k(y_n) - \sum_{(i,j) \in h'} l(y_n, (i, j)) \quad (4)$$

where $n(y_n)$ and $k(y_n)$ are the number of subtopic candidates and the number of clusters in the correct clustering y_n. $l(y_n, (i, j)) = 1$ if q_i and q_j are in the same cluster in y_n, otherwise $l(y_n, (i, j)) = -1$. Then the optimization problem introduced in Eq. (2) can be solved by the Concave-Convex Procedure (CCCP) (Yuille and Rangarajan, 2003).

2.4 Pairwise Similarity Features

The proposed approach requires a set of features to measure the similarity between two subtopic candidates. Table 1 lists the features employed in our approach. These features are categorized into two types: lexicon-based similarity and URL-based similarity. The lexicon-based similarity features are employed to measure the string similarity between two subtopic candidates. And the URL-based similarity features are used to measure the semantic similarity between two subtopic candidates. The basic idea is that if two queries share many clicked URLs, they have similar search intent to each other (Li et al., 2008). To

make the features comparable with each other, we normalize them into range of [0, 1] accordingly.

Feature	Description
COS	cosine similarity between q_i and q_j
EUC	Euclidean distance between q_i and q_j
JAC	Jaccard coeff between q_i and q_j
EDIT	norm edit distance between q_i and q_j
LEN	|length(q_i)-length(q_j)|
SUBSET	whether one is a subset of the other
UCOS	cosine similarity between the clicked URL sets of q_i and q_j
UJAC	Jaccard coeff between the clicked URL sets of q_i and q_j

Table 1: pairwise similarity features employed in our approach

3 Experiments

3.1 Data Set

To illustrate the effectiveness of our approach, we use 100 ambiguous/multifaceted queries provided by the NTCIR-9 intent task as original queries and collect their subtopic candidates from SogouQ dataset (http://www.sogou.com) using the method mentioned in section 2.1. For the 100 queries, we totally collect 2,280 query subtopic candidates. Three annotators manually label these candidates with their subtopics according to the content words of these candidates and their clicked web pages (if there are clicked URLs for the candidate in query log). A candidate belongs to a specific subtopic if at least two annotators agree with it. At last we obtain 1,086 subtopics. We randomly split the original queries into two parts: half used for training and the rest for testing.

3.2 Evaluation Metrics and Baselines

To evaluate the performance of our approach, we employ the measures in (Luo, 2005), which are computed as follows,

$$p = \frac{\sum_i \pi(R_i', g(R_i'))}{\sum_i \pi(R_i', R_i')}, \quad r = \frac{\sum_i \pi(R_i', g(R_i'))}{\sum_j \pi(R_j, R_j))}$$

where R' is the predicted partition and R is the ground-truth partition; $\pi(A, B)$ is a similarity measure between set A and B, which is Jaccard coefficient in this paper; and $g(.)$ is the optimal mapping between R' and R. Based on p and r, f-*measure* can be calculated as,

$$f-measure = \frac{2 \times p \times r}{p + r}$$

The higher the *f-measure* score is, the better performance an approach achieves.

We used the following approaches as baselines:

- K-means: we perform the standard k-means clustering algorithm with different manually defined similarity measures to mine query subtopics. COS, JAC, EUC, EDIT refer to cosine similarity, Jaccard similarity, Euclidean distance, and edit distance, respectively.
- Binary Classification Cluster with the all-connection structure (BCC-AC): BCC-AC uses a SVM classifier to learn the weight w and clusters with correlation clustering method.
- Binary Classification Cluster with the strong-connection structure (BCC-SC): BCC-SC uses a SVM classifier to learn the weight w and clusters with the method presented in Algorithm 1.

For the proposed methods, we denote the method with the all-connection structure as AC and the method with the strong-connection structure as SC. The parameter C in Eq. (2) is picked from10^{-2} to 10^{4} using a 10-fold cross validation procedure.

3.3 Experimental Results

Methods	p	r	f-measure
K-Means-COS	0.6885	0.6589	0.6734
K-Means-JAC	0.6872	0.6616	0.6742
K-Means-EUC	0.6899	0.6652	0.6774
K-Means-EDIT	0.6325	0.6275	0.6300
BCC-AC	0.7347	0.7263	0.7305
BCC-SC	0.7406	0.7258	0.7331
AC	0.8027	0.7911	0.7968
SC	**0.8213***	**0.8187***	**0.8200***

Table2: the performance of all methods. "*" indicates significant difference at 0.05 level using a paired t-test.

Table 2 presents the experimental results. Compared with K-Means methods with different manually defined similarity measures, SC achieves at least **13.14%** precision improvement, **15.35%** recall improvement, and **14.26%** F-Measure improvement. And AC achieves at least **11.28%** precision improvement, **12.59%** recall improvement, and **11.94%** F-Measure improvement. These results confirm that the similarity between two subtopic candidates is affected by many factors and our methods can achieve more desirable query subtopics by learning a similarity measure.

Compared with BCC-AC and BCC-SC, SC achieves at least **8.07%** precision improvement, **9.29%** recall improvement, and **8.69%** F-Measure improvement. And AC achieves at least **6.21%** precision improvement, **6.53%** recall im-

provement, and **6.37%** F-Measure improvement. These results confirm that the dependencies among the subtopic candidates within the same subtopic are useful for learning a similarity measure for query subtopic mining.
Compared with AC, SC achieves **1.86%** precision improvement, **2.76%** recall improvement, and **2.32%** F-Measure improvement. These results confirm that a subtopic candidate belonging to a given query subtopic does not need to similar with all subtopic candidates within the given subtopic.

In order to understand which pairwise similarity feature is important for the problem of query subtopic mining, we list the features and their weights learned by SC, AC, and BCC (Binary Classification Cluster) in Table 3.

Feature \ Method	SC	AC	BCC
COS	0.08	0.04	0.19
EUC	−1.74	−1.07	−0.73
JAC	4.44	4.73	4.90
EDIT	−1.60	−1.01	−0.48
LEN	−1.34	−0.91	−1.07
SUBSET	0.21	0.11	−0.05
UCOS	0.01	0.01	0.04
UJAC	0.06	0.07	0.09

Table 3: the features and their weights learned by the different methods.

As can be seen in Table 3, JAC has the largest importance weight for mining query subtopics in the three methods. The URL-based features (UCOS and UJAC) have small importance weight. The reason is that clicked URLs are sparse in our query log and many long-tail subtopic candidates in the same subtopic do not share any common URLs.

4 Conclusions

In this paper, we propose an approach for mining query subtopics from query log. Compared with previous approaches, our approach learns a similarity measure by explicitly modeling the dependencies among subtopic candidates within the same subtopic. Experimental results on real queries collected from a search engine log confirm our approach produces more desirable query subtopics by using the learned similarity measure.

Acknowledgments

The work is supported by the National Natural Science Foundation of China under Grants no. 61433015 and 61272324, and the National High Technology Development 863 Program of China under Grants no. 2015AA015405. Moreover, we sincerely thank the reviewers for their valuable comments.

References

N. Bansal, A. Blum, and S. Chawla. 2002. Correlation clustering. In *Machine Learning*, 56, 89-113.

A. Z. Broder. A taxonomy of web search. 2002. In *Sigir Forum*, 36:3-10.

A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. 2007. Robust classification of rare queries using web knowledge. In *SIGIR*, pp. 231-238.

C. L. A. Clarke, N. Craswell, and I. Soboroff. 2009. Overview of the trec 2009 web track. In *TREC'09*, pp. 1-9.

Y. Hu, Y. Qian, H. Li, D. Jiang, J.Pei, and Q. Zheng. 2012. Ming query subtopics from search log data. In *SIGIR'12*, pp. 305-314.

T. Finley and T. Joachims. 2005. Supervised clustering with support vector machines. In *ICML*, pp. 217-224.

B. J. Frey and D. Dueck. 2007. Clustering by passing messages between data points. In *science*, 315(5814):972-976.

Y. Li, Z. Zheng, and H. K. Dai. 2005. Kdd cup-2005 report: facing a great challenge. In *SIGKDD Explor. Newsl.*, 7:91-99.

L. Li, Z. Yang, L. Liu, and M. Kitsuregawa. 2008. Query-url bipartite based approach to personalized query recommendation. In AAAI'08, pp. 1189-1194.

X. Luo. 2005. On Coreference resolution performance metrics. In *HLT&EMNLP*, pp. 25-32.

F. Radlinski, M. Szummer, and N. Craswell. 2010. Inferring Query Intent from Reformulations and Clicks. In *WWW*, pp. 1171-1172.

R. Song et, al. 2011. Overview of the ntcir-9 intent task, In *NTCIR-9*, pp.82-105.

K. Tsukuda, Z. Dou, and T. Sakai. 2013. Microsoft research asia at the ntcir-10 intent task. In *NTCIR-10*, pp. 152-158.

J. Wang, G. Tang, Y. Xia, Q. Hu, S. Na, Y. Huang, Q. Zhou, and F. Zheng. 2013. Understanding the query: THCIB and THUIS at ntcir-10 intent task. In *NTCIR-10*, pp. 132-139.

C. J. Yu and T. Joachims. 2009. Learning Structural SVMs with Latent Variables. In *ICML*, pp. 1169-1176

A. Yuille, and A. Rangarajan. 2003. The concave-convex procedure. In *Neural Computation*, 15, 915.