Clustering User Queries into Conceptual Space

Li-Chin Lee* and Yi-Shin Chen†
*Institute of Information Systems and Applications,
National Tsing Hua University, Hsin-Chu, Taiwan
E-mail: lichinisme@gmail.com
†Institute of Information Systems and Applications,
Department of Computer Science, National Tsing Hua University, Hsin-Chu, Taiwan
E-mail: yishin@gmail.com , Corresponding Author

***Abstract*—The gap between user search intent and search results is an important issue. Grouping terms according to semantics seems to be a good way of bridging the semantic gap between the user and the search engine. We propose a framework to extract semantic concepts by grouping queries using a clustering technique. To represent and discover the semantics of a query, we utilize a web directory and social annotations (tags). In addition, we build hierarchies among concepts by splitting and merging clusters iteratively. Exploiting expert wisdom from web taxonomy and crowd wisdom from collaborative folksonomy, the experiment results show that our framework is effective.**

I. INTRODUCTION

With the enormous increase in the amount of information on the Internet, search engines have become an indispensable tool for people looking for information. People describe the information they need in the form of queries, and the search engine returns matched web objects on the results page. However, analysis of search logs shows the proportion of queries without follow-up click-through data is as high as 42%, which indicates that users are not satisfied with the search results [13]. This could be due to the search engine or the user. These two reasons are discussed below.

- **The Search Engine.** Search engines may fail to return useful results when they cannot understand what users want from their queries. According to previous research, the average query consists of only two terms [12], which causes ambiguity for the search engine, and makes it difficult to detect specific user intentions. Currently, if a search engine receives an ambiguous query, it will just return all documents that match the query without organizing them. Thus, users have to filter through the information by themselves. In response to this problem, Google recently released a search tool called "Wonder Wheel" [2], which is intended to help users simplify and arrange search results. Google's Wonder Wheel shows search terms related to the current searched query.

- **The User.** Users have difficulty describing their information needs. A concept can usually be described from many standpoints and related to many terms. People may use different terms to express the same information need, or use the same term to find different things. For instance:
  1) "The eagles" can refer to a rock band or a kind of bird.
  2) "Bald eagle" and "American Eagle" share the word "eagle", but are not related. The former is a kind of bird, while the latter is a clothing brand.
  3) "American Eagle" and "Abercrombie and Fitch" are related, even though they are not syntactically related.

  The first example describes query ambiguity. The second example shows the limitation of using syntax to judge semantic relatedness. Two semantically related but syntactically unrelated terms are shown in the last example. Search engines now face human semantic problems. The Google Wonder Wheel helps users discover their search intentions only by providing syntactically related terms.

We believe the search engine can bridge the gap between user intentions and query terms by grouping semantically related terms based on semantic concepts. In this way, word diversity can be discovered, and more information from terms related by a semantic concept can be returned, as illustrated in Figure 1.

We propose a framework that applies a clustering technique to group semantically related terms. The best way to understand user behavior is to analyze the search log collected from users. We consider user queries and clicked results to be crowd wisdom. Moreover, we want to make use of existing achievements and resources on the web.

We make the following contributions:

- We integrate clicked URLs, web directories, and social annotations to represent user queries and group them into semantic concepts.
- We propose an effective framework based on K-means to extract the hierarchies among semantic concepts.
- We automatically extract and label semantic concepts through clustering.

The rest of this paper is organized as follows. We review related work in Section II. Section III briefly introduces related web resources utilized in our research. Section IV provides an overview of the system framework and then elaborates on

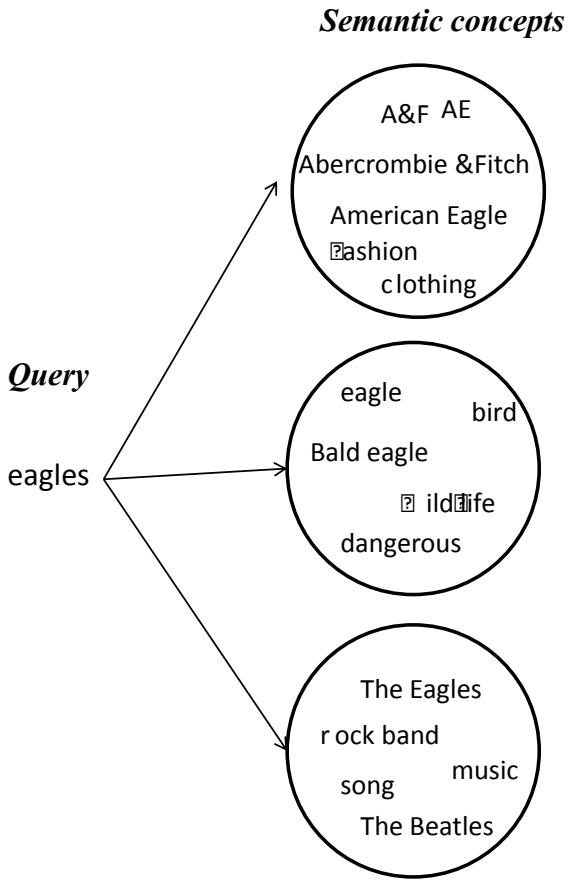


Fig. 1. Example of a query and related semantic concepts.

each component respectively. The empirical study is reported in Section V. Results are shown in Section **??**. Section VI concludes the paper and discusses future work.

II. RELATED WORK

The most important issue in query clustering is how to compare query similarity. Most of the numerous previous researchers have aimed to use query clustering to discover similar queries by exploiting clicked URLs. Clicked URLs represent the user's information needs and assist in clarifying the meaning of the user's queries. Queries shared among many clicked documents will be in the same clusters. Beeferman and Berger [5] applied an iterative agglomerative clustering algorithm to a bipartite graph built from queries and clicked URLs. As this method is content-ignorant, researchers started to consider how to summarize the content of clicked URLs. Wen et al. [15], [7] applied a similarity measure to the content of the query and the hierarchy of clicked URLs. Other recent research applied a clustering technique to mine the hierarchy among groups of user queries [11], [14]. Several researchers have identified query similarity and clustered queries based on refinements from the user session [10] [6]. However, no method has yet been described of representing a query and comparing semantic similarity between queries in order to discover semantically related query groups.

The present study use the web directory and social annotations to understand clicked URLs and improve query clustering.

III. DATA

Three types of web resources are utilized in our research, each containing either crowd wisdom or expert wisdom. The following briefly introduces the characteristics of such data.

- **Click-through Data.** Click-through data contains user queries and clicks, and has been seen as implicit feedback from users, because user intention and the meaning of the queries can be clarified by the content of clicked URLs. In order to understand the content of clicked URLs, we consider the web directory and social annotations. In this research, clicked URLs play an important role in mapping the taxonomy (web directory) and folksonomy (social annotations) to user queries.

- **Social Annotation.** Social annotation, also known as tag, has attracted much attention from researchers recently. Collaborative tagging is a popular web service that allows users to annotate or classify web objects with arbitrary terms. As a result, other users can capture the concept of a web object through its tags. Del.icio.us [1] is a popular web 2.0 service, which provides users with an online bookmarking service. Users can tag any URL with descriptive terms of their choice to classify their own bookmarks. The non-hierarchical classification is known as folksonomy.

- **Web Directory.** Web directories classify web resources with hierarchical structure according to different topics. The Open Directory Project(ODP) [4] is a well-known human-edited directory of the web. It classifies web pages into one location category World and 15 general categories: Adult, Arts, Business, Computers, Games, Health, Home, Kids and Teens, News, Recreation, Reference, Science, Shopping, Society and Sports. These 15 general categories are used in our work.

IV. METHODOLOGY

A. System framework

Figure 2 presents our research framework, which consists of three main components. *Data Mapping* preprocesses and integrates all the web resources we utilize. To represent queries, tags will be mapped to queries by a voting mechanism. The output will be a query-tag matrix, which is the input of concept clustering. *Concept Clustering* and *Splitting and Merging* are iterative processes. In order to discover the hierarchy among clusters, the latter analyzes the clusters gained from the former phase to check if the clusters should be split or merged. The detailed mechanism for each component is illustrated below.

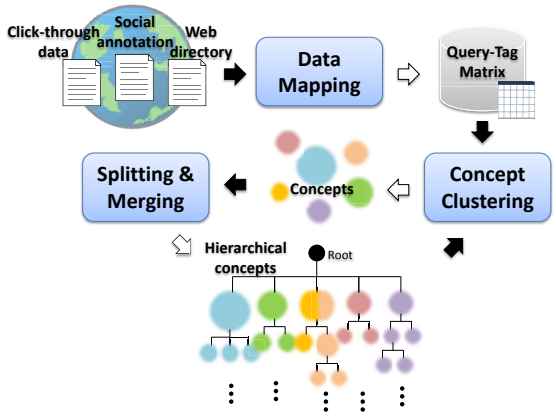
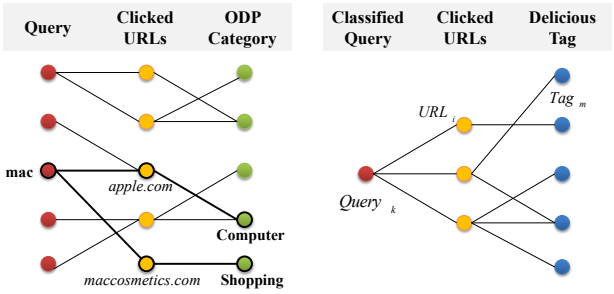


Fig. 2. System Framework.



(a).Classify queries (b).Map tags to queries

Fig. 3. Two-phase mapping.

B. Data mapping

To represent queries semantically, we exploit the wisdom of the web directory and social tags in two phases.
In the first phase, queries are classified into the 15 ODP categories through clicked URLs. Figure 3(a) illustrates a mapping example using query "mac", which is multiple-classified to the Computer and Shopping categories, because some of the users who issued the query "mac" clicked the Apple company's website, classified to the Computer category in ODP, and others clicked the MAC cosmetics company's website, which is under the Shopping category in ODP.

In the second phase, for each category, tags crawled from Del.icio.us are mapped to queries through clicked URLs (Figure 3(b)). Tags are the semantic features used to represent queries. For each query, different features have different semantic weights. We design a scoring function based on a cumulative voting mechanism to compute the weighting score between queries and their features. Let $C_{x|y}$ denote the cumulative votes to x by y, Q_k denote the kth query, T_m denote the mth tag; $Score_{Q_kT_m}$ is then computed by Equation 1.

$$Score_{Q_kT_m} = \sum_{j,l} \frac{C_{U_j|Q_k}}{\sum_i C_{U_i|Q_k}} \times \frac{C_{T_l|U_j}}{\sum_i C_{T_m|U_i}} \quad (1)$$

Subsequently, each category obtains a query-tag scoring matrix for clustering.

$$Score_{k,m} = \begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,n} \\ s_{2,1} & a_{2,2} & \cdots & s_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m,1} & a_{m,2} & \cdots & s_{m,n} \end{pmatrix}$$

C. Clustering

The search log updates quickly and is usually extremely large, so the required characteristics of the clustering algorithm are scalability and speed. K-means is a partition-based clustering algorithm, which is simple and fast. We choose it as the clustering method, and perform refinements by splitting and merging the clusters from K-means. A notable drawback of K-means is that k is needed in advance. Instead of manually assigning the k, we automatically estimate the initial k using the ODP structure. If a set of clicked URLs corresponds to a query set, the k can be estimated by considering how many categories ODP uses to classify the set of URLs.

D. Splitting and Merging

In order to refine the clusters created by K-means and extract the hierarchy among concepts, clusters start to be split or merged after the initial clustering. The following illustrates how we choose clusters to split and merge.

1) Cluster Splitting.: We measure whether a cluster should be split based on the intra-distance between the cluster centroid and the query. We assume that if the distance distribution in a cluster is a perfect normal distribution, then it is a good cluster. The distance distribution in a cluster can be described by the coefficient of variation, skewness, and kurtosis. The statistics are used to determine whether a cluster needs to be split.

For the clusters that need to be split, the k, which indicates the number of possible splits, is also estimated automatically in accordance with the distance distribution graph. Figure 4visualizes the relationship between distance distribution and the query distribution of a cluster. We analyze the distribution graph and set a different k for clustering. Then we measure the result using the Davies-Bouldin index [8], and find that the number of k is consistent with the number of peaks on the distance distribution graph.

Algorithm 1 shows the flow of automatically detecting whether a cluster should be split and automatically estimating the number of possible splits. Clusters will split continuously until they meet the end condition.

2) Cluster Merging: We find clusters should be merged based on the inter-distance between clusters. We compute the distances between cluster pairs and pick the pairs that are extremely close. For example, if a cluster pair A,B is picked and the other cluster pair B,C is also picked, then cluster

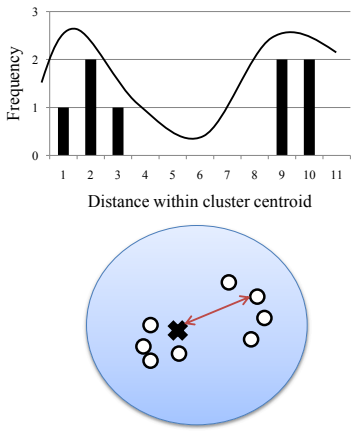


Fig. 4. Distance distribution in a cluster

A,B,C will be merged to form a new cluster.

Algorithm 2 shows the flow of picking clusters to merge. The process will stop if none of the cluster pairs meets the condition for merging.

Algorithm 1 Cluster splitting algorithm

```
for each cluster do
  for each node in cluster do
    ComputeDistance(cluster centroid, query)
  end for
  Dist = ComputeDistanceDistribution()
  CV = ComputeCoefficientOfVariation(Dist)
  skew = Absolute(ComputeSkewness(Dist))
  kurt = Absolute(ComputeKurtosis(Dist))
  if CV> 5 and (skew> 1 or kurt> 1) then
    k = EstimateK(Dist)
    KmeansClustering(k)
  end if
end for
```

Algorithm 2 Cluster merging algorithm

```
for each cluster A do
  for each cluster B do
    if A≠B then
      ComputeDistance(A,B)
    end if
    Dist = ComputeDistanceDistribution()
    candidates = FindClustersShouldBeMerged(Dist)
    pickedclusters = SingleLink(candidates)
    MergeClusters(pickedclusters)
  end for
end for
```

3) Cluster Labeling: Each cluster will be automatically labeled during clustering. The cluster centroid is considered as the representative of a cluster, and the cluster labels are the top-scored attributes (tags) of the cluster centroid.

V. EXPERIMENTAL EVALUATION

In this section, we first briefly describe the source query log, followed by the experimental setup of data set preparation and associated use. Our approach was evaluated according to its ability to group semantically related queries, and it was compared with a density-based clustering algorithm and a hierarchical clustering algorithm.

A. Experiment Setup

Our data set was extracted from the MSN search log released by Microsoft Live Labs. The data were collected from United States users sampled in May 2006. Besides click-through data, there were two types of web resources we needed: the ODP classification structure and content, and the social annotations of clicked URLs. ODP permits any user to download its structure and content data. For the social annotations, we crawled the tags of a set of URLs from Del.icoi.us between February and May 2010. We performed some preprocess operations on the tags, including:

- Removing the tags annotated by only one URL.
- Removing the tags composed only of symbols and the tags containing non-ASCII characters.
- Removing meaningless symbols in tags.
- Integrating the tags with the same meaning but in different forms into a uniform tag, e.g., data@mining and datamining become data_mining.

After gathering and preprocessing relevant data sources, we exploited the click log data containing query string and search result click-through and selected a query set containing $88,302$ queries, by the following steps:

1. Rank all clicked URLs by their clicked times and leave the URLs for which click times are bigger than or equal to 20. By this means we collected 38,973 distinct URLs.
2. Find the URLs that intersect with ODP collection and Del.icio.us bookmarks. The URL intersection consisted of 10,538 URLs.
3. According to the URLs, we gathered all corresponding queries in click-through data. We removed queries that consisted only of symbols or contained non-ASCII characters. This yielded 88,302 distinct queries to be used in the experiment.

Through corresponding clicked URLs, queries were mutiple-classified to 15 sub-data sets according to the first level of the ODP structure. The classification of queries was the first level of our hierarchical conceptual space. Then, for each query, tags were mapped, also through clicked URLs. Figure 5 shows the number of queries and the features of each category. It also reflects the general search interest of crowds.

Finally, we used K-means with the Euclidean distance measure. By iteratively clustering, splitting, and merging the conceptual clusters, we established a conceptual space with a 35-level hierarchy.

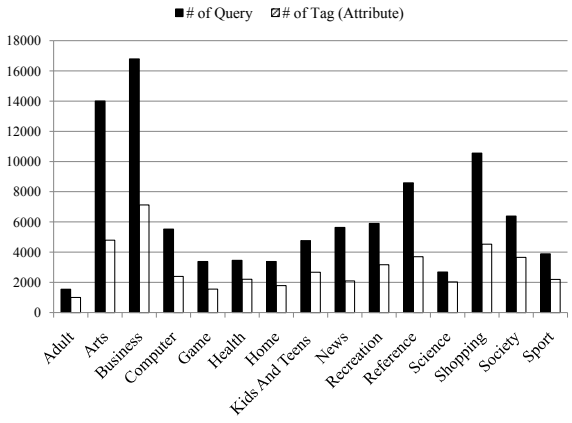


Fig. 5. Number of queries and tags in each category

Labeler 1	Labeler 2	Labeler 3
88.37%	82.5%	79.55%

TABLE I
ACCURACY OF COMPARING ODP-APPLIED CLASSIFICATION WITH KDD-CUP 2005 LABELED DATA

B. Experimental Results

1) Effectiveness of Applying ODP Structure: First we wanted to know whether applying the ODP structure is effective. We compared our query classification results with the manually labeled queries from KDD-cup 2005 [3]. The labels represent a general semantic concept, which is the same as our first level classification. We determined the accuracy of comparing query data from KDD-cup with ODP classification, and the results are shown in Table I. The average accuracy is higher than 80%, showing that using the ODP scheme to classify queries is effective and reliable.

2) Comparing with Other Clustering Algorithms: We applied two different clustering methods with the Euclidean distance measure on our data set and compared the above evaluation metrics with our approach. The first method was single-link agglomerative hierarchical clustering (AHC), which is a bottom-up hierarchical clustering. The second method was DBSCAN [9], which is a density-based clustering that can discover clusters with arbitrary shapes. We set epsilon to 1 and minimum points to 5.

The following metrics were proposed to measure the effectiveness of several query clustering approaches:

- *Efficiency:* The ability to handle data on different scales.
- *Accuracy:* The ability to discover natural clusters.

Figure 6 displays the efficiency of each clustering algorithm. The data scalability was measured as the number of queries multiplied by the number of feature dimensions. Our method was apparently the quickest. AHC and the DBSCAN are relatively unrobust for a large and high-dimensional data set.

Our results show that our method is both accurate and efficient in handling data sets in different scales.

We manually labeled 50 queries in several categories. We computed accuracy by seeing if the clusters matched the manually labeled groups. We compared our method with DBSCANm which is good at discovering natural clusters. Figure 8 presents the results. Apart from the Health category, our method performed better than DBSCAN.

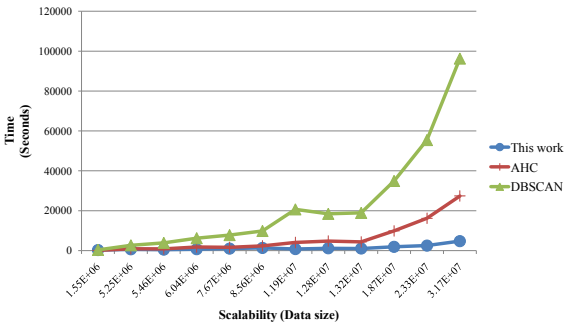


Fig. 6. Efficiency comparison with DBSCAN and AHC

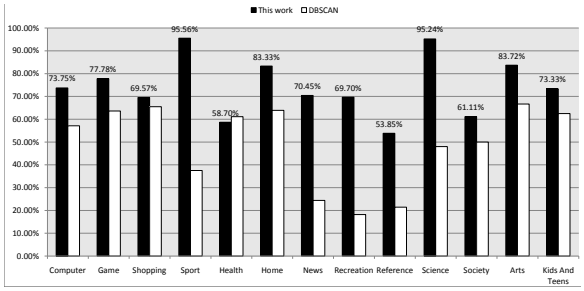


Fig. 7. Accuracy comparison with DBSCAN

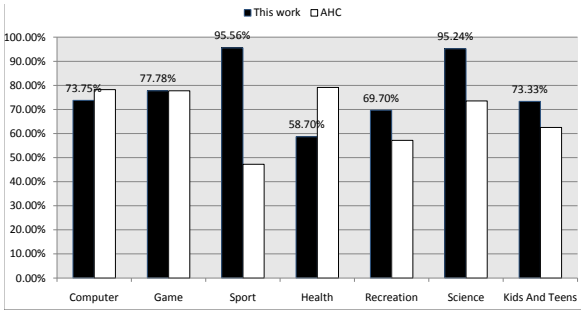


Fig. 8. Accuracy comparison with AHC

3) Query Clustering Results: In this section, we show the query clustering result by a simple example in Table II. By exploring the conceptual spaces we constructed, the term *eagle* can be related to the *Shopping*, *Arts*, and *Science* categories, which is the first level of our concept hierarchies. Then we can explore the clusters in lower levels under each category. Each cluster is labeled with tags.

Further, we provide an interface for exploring in the conceptual space we constructed (Figure 9). By inputting a keyword, the system will show the user the related concept according to that keyword. The user can check the concept in the hierarchy to see other queries classified in the same concept.

Query	Related Concept in Level 1	Related Concept in Lower Levels	Related Queries Example
eagle	Shopping	Clothing, Clothes, Fash- ion, Shop,Apparel	american eagle american eagle outfitters american eagle clothing old navy gap outlet banana republic
		Wheels, Rims, Covette,Tires, Cars	american eagle rims eagle alloys american racing vintage wheels black racing rims
	Arts	Music,Bands,Rock,Entertainment,Artist	eagles band hotel california eagles eagles desperado santana band jazz festival
	Science	Wildlife, Environment, Education, Science, Government	information on eagles eagle bird american bald eagle coyotes eat US fish and wildlife service
	Recreation	Travel, Airlines, Airline, Fights, Airfare	american eagle airlines us air airlines aa airlines phone northwest airline southwest airlines reservations

TABLE II
EXAMPLE OF QUERY *eagle*

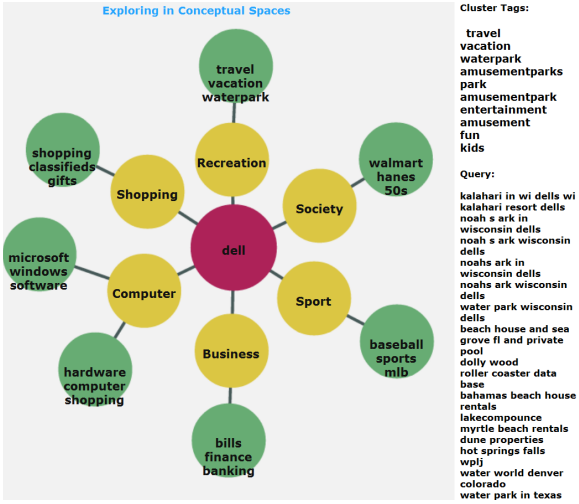


Fig. 9. Interface for exploring in conceptual spaces

VI. CONCLUSION AND FUTURE WORK

In this paper, we present a novel way to represent queries by aggregating a web directory and social annotations with click-through data. Our method successfully clusters semantically related queries and labels the clusters automatically. Moreover, we extract the hierarchies among semantic concepts by an effective splitting and merging mechanism, which is also helpful for refining the clusters from K-means.

To advance our approach, we could: (1) do more tag-related researcj to improve the features of queries and reduce the dimensionality, (2) investigate more search log data to enrich the current concepts. The approach could be refined with more considerations in future work.

ACKNOWLEDGMENT

This work is supported by the National Science Council, Taiwan, under grant 102-2221-E-007-093-, 101-2221-E-007-126, 100- 2221-E-007-109, 99-2221-E-007-092, and 98-2221-E-007-096.

REFERENCES

[1] Del.icio.us. http://delicious.com/.
[2] Google wonder wheel. http://www.googlewonderwheel.com.
[3] Kdd-cup 2005. http://www.sigkdd.org/kdd2005/kddcup.html.
[4] Open directory project. http://www.dmoz.org/.
[5] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 407–416, New York, NY, USA, 2000. ACM.
[6] I. Bordino, C. Castillo, D. Donato, and A. Gionis. Query similarity by projecting the query-flow graph. In *SIGIR '10: Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522, New York, NY, USA, 2010. ACM.
[7] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 875–883, New York, NY, USA, 2008. ACM.
[8] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:224–227, 1979.
[9] M. Ester, H. peter Kriegel, J. S, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. 2nd int. Conf. on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, 1996.
[10] E. Sadikov, J. Madhavan, L. Wang, and A. Halevy. Clustering query refinements by user intent. In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 841–850, New York, NY, USA, 2010. ACM.

[11] D. Shen, M. Qin, W. Chen, Q. Yang, and Z. Chen. Mining web query hierarchies from clickthrough data. In *AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence*, pages 341–346. AAAI Press, 2007.

[12] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, 1999.

[13] S.-E. Tsai, C.-Y. Tsai, S.-W. Tu, and Y.-S. Chen. Improving query suggestion by utilizing user intent. 2010.

[14] X. Wang, B. Tan, A. Shakery, and C. Zhai. Beyond hyperlinks: organizing information footprints in search logs to support effective browsing. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1237–1246, New York, NY, USA, 2009. ACM.

[15] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 162–168, New York, NY, USA, 2001. ACM.