Dynamic Facet Hierarchy Constructing for Browsing Web Search Results Efficiently

Wei Chang and Jia-Ling Koh(✉)

Department of Information Science and Computer Engineering,
National Taiwan Normal University, Taipei, Taiwan, Republic of China
jlkoh@csie.ntnu.edu.tw

Abstract. In this paper, a method is proposed to dynamically construct a faceted interface to help users navigate web search results for finding required data efficiently. The proposed method consists of two processing steps: 1) candidate facets extraction, and 2) facet hierarchy construction. At first, the category information of entities in Wikipedia and a learning model are used to select the query-dependent facet terms for constructing the facet hierarchy. Then an objective function is designed to estimate the average browsing cost of users when accessing the search results by a given facet hierarchy. Accordingly, two greedy based algorithms, one is a bottom-up approach and another one is a top-down approach, are proposed to construct a facet hierarchy for optimizing the objective function. A systematic performance study is performed to verify the effectiveness and the efficiency of the proposed algorithms.

Keywords: Faceted search · Wikipedia · Search result organization

1 Introduction

Keyword based search is a popular way for discovering required data of interest from a huge collection of resources. The effectiveness of data retrieval mainly depends on whether the given queries properly describe the information needs of users. However, it is not easy to give a precise query because most queries are short (less than two words on average) and many query words are ambiguous. Using a general keyword with broad semantics as a query usually causes a huge amount of data returned. Most of the search services return results as a ranked list, but it is difficult for users to explore and find objects satisfying their search needs from a long list of results. Accordingly, how to automatically group search results into meaningful "topics" has become a significant issue to improve the usability of search results.

Faceted interface is a common feature of e-commerce sites, where the objects are structured data with attributes. Users can select an attribute and specify a constraint on the attribute value, such as the price or category of products, to filter the search results. Recently, several search engines also provide a set of facets to the search

This work was partially supported by the R.O.C. M.O.S.T. under Contract No. 103-2221-E-003-033.

M. Ali et al. (Eds.): IEA/AIE 2015, LNAI 9101, pp. 293–304, 2015.
DOI: 10.1007/978-3-319-19066-2_29

interface, which are usually the structural attribute of data such as location, time, size of document, etc. Faceted interfaces provide a convenient and efficient way to navigate the search results. However, most of the systems have to define the facets of their search interfaces in advance or assume that a prior taxonomy exists. Inspired by the recent works on automatic facets generation [7], in this paper, we would like to dynamically find keyword sets from the search results and construct a facet hierarchy of the search results in order to help users find the required data more efficiently.

Imagine that a user is exploring the news about "iPhone" and give a keyword query. The search engine returns a ranked list of search results with snippets. The snippet of a search result is a short text segment, which summarizes the important content of the search result. Our goal is to dynamically create a faceted interface for covering the top *k* search results, where the interface consists of a hierarchy of categories for the topic words in the search results. For example, a search result of the news of iPhone talks about "BlackBerry Messenger coming to iPhone and Android: … ." Accordingly, the facets include "BlackBerry(company)_mobile_phones", "information applications", and so on. The user can navigate the category path "information applications/Instant_ messaging/BlackBerry Messenger" to find the related results.

In order to eliminate the network transmission time of downloading the entire webpages of the search results, our proposed method extracts the topic terms from the snippets of search results as the candidates of facet terms on the lowest level of the facet hierarchy. Besides, the categories of the topic terms are looked up from Wikipedia to be the candidate facets on the higher levels. However, a facet term may belong to multiple categories in Wikipedia but some categories are not semantically related to the search results. To deal with this problem, we perform a filtering step which applies a learning-to-rank strategy to select the categories which are highly semantics related to the search results. Furthermore, we define an objective function of a facet hierarchy to estimate the browsing cost of users for finding a search result from the facet hierarchy. Then we propose two greedy-based algorithms, a top-down approach and a bottom-up approach, to construct a facet hierarchy from the candidate facet terms for optimizing the objective function. A systematic performance study is performed to verify the effectiveness and the efficiency of the proposed algorithms.

The rest of the paper is organized as follows. In the next section, a brief overview of the related works is introduced. The formal problem definition of constructing a facet hierarchy of query results is given in Section 3. Section 4 introduces the proposed method of facet terms extraction and two algorithms of constructing a facet hierarchy. The performance evaluation on the proposed algorithms is reported in Section 5. Finally, we conclude this paper in Section 6.

2 Related Works

Many existing works [2][3][4][9][13] studied how to separate the search results into semantics related groups, which are called search results clustering. Topic modeling is a technique for extracting latent topics from a set of unlabeled documents. LDA [2][4] is a popular approach to construct a topic model, whose goal is to assign the documents representing the same topic concept into a cluster. After performing the unsupervised training phase, each obtained cluster is labeled manually. Then the constructed LDA model is used to assign the search results to the corresponding topic clusters. Many researches [1][8][11][12] applied the LDA approach to perform topic discovery from text data for various applications. However, the disadvantage of LDA is that it is required to give the number of clusters manually. It is not easy to determine a proper number of topic clusters in advance.

On the other hand, some works [9][13] applied knowledge bases, such as Wikipedia, to find topic clusters of search results. These works extracted the topic terms from the search results and find the corresponding entities in the knowledge bases. Then the categories of the entities in the knowledge bases are used to perform topic clustering. However, in [9], it provided only single level of clusters for the search results.

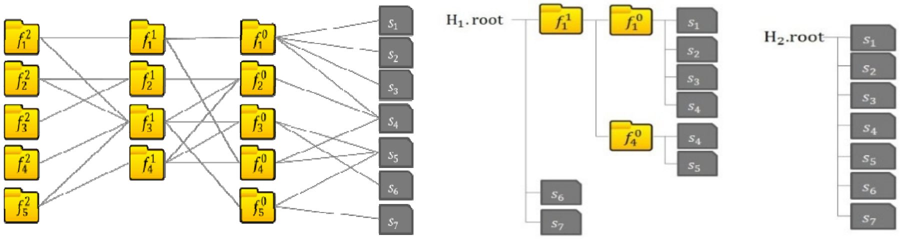
In e-commerce systems and digital libraries, faceted search is a popular way for searching objects with different attributes. The faceted search system provides a simple interface for users to filter and browse objects easily according to the selected faceted-value pair. However, most of the faceted search interface is constructed for the structured data where the faceted-value pairs are predefined. In recent years, some works [6][7][11] studied how to extract facets and facet terms from data dynamically from unstructured or semi-structured data. In [6], a supervised approach was proposed to recognize query facets from noisy candidates. This approach applied a directed graphical model, which learns how to recognize the facet terms in search results and how to group the terms into a query facet together. However, this approach didn't provide the semantic concept of the corresponding query facet. [11] extended LDA topic model to find facets from the messages posted in Twitter. However, this approach only extracts five specific types of facet terms, i.e. locations, organizations, persons, time distributions and general terms.

The Facetedpedia system proposed in [7] was a faceted retrieval system designed for information discovery and exploration in Wikipedia. The system automatically and dynamically constructs facet hierarchy for navigating the set of Wikipedia articles resulting from a keyword query. This work used the hierarchical categories of the Wikipedia articles as the candidate facets and designed a ranking algorithm for these facets. The proposed algorithm aims to select diverse facets for minimizing the navigation cost of search results. Although our goal is similar to the one of Facetedpedia, it has more challenges to generate a facet hierarchy for organizing the web search results than generating a facet hierarchy of the Wikipedia search results. The Facetedpedia system downloaded all the articles in the Wikipedia search results and built a category index for each article by performing an offline process. However, for generating a facet hierarchy of the web search results, it is not feasible to download and process the whole web pages of the search results. Accordingly, our approach processes the snippets instead of the web pages of the search results. The snippet of a search result is usually short. How to extract facet terms from the snippets and the knowledge base for effectively organize search results is the main issue.

3 Problem Definition

Given a query q, let $R_q = \langle s_1, s_2, \ldots, s_k \rangle$ denote the ranked top-k list of the search results with snippets which are returned by a search engine. Given a search result snippet s_i, some terms which correspond to the named entities in Wikipedia are extracted as the candidate facet terms (the extraction method is introduced in Sec 4.1). Given R_q, let F^0 denote the set of candidate facet terms $\{f_1^0, f_2^0, \ldots, f_{|F^0|}^0\}$, where each f_j^0 appears in at least one snippet in R_q. For example, given a query "bull", $R_{bull} = \langle s_1, s_2, \ldots, s_7 \rangle$ denotes the top 7 snippets in the search result. The set of terms extracted from the snippets is F^0 ={"Bull(Company)", "Chicago Bull", "Red Bull(Drink)", "Derrick Rose", "Spanish Fighting Bull"}. If a candidate facet term f_j^0 appears in a snippets s_i in R_q, we call that f_j^0 *covers* s_i. Besides, E^0 is used to denote the cover relationship between F^0 and R_q.

As defined in [7], the Wikipedia *category hierarchy* is a connected and directed acyclic graph $\mathcal{H}(r_{\mathcal{H}}, C_{\mathcal{H}}, \mathcal{E}_{\mathcal{H}})$, where $r_{\mathcal{H}}$ denotes the root node, the node set $C_{\mathcal{H}}$consists of the categories in Wikipedia, and the edge set $\mathcal{E}_{\mathcal{H}}$ denotes the set of category-subcategory relationships in Wikipedia. Given R_q, the category hierarchy of the facet terms $FC(C_{\mathrm{F}}, \mathcal{E}_{\mathrm{F}})$ is a connected subgraph of the category hierarchy $\mathcal{H}(r_{\mathcal{H}}, C_{\mathcal{H}}, \mathcal{E}_{\mathcal{H}})$, where $C_{\mathrm{F}} \subseteq C_{\mathcal{H}}$ and $\mathcal{E}_{\mathrm{F}} \subseteq \mathcal{E}_{\mathcal{H}}$, and $\mathbf{c} \in C_{\mathrm{F}}$ if $\mathbf{c}$ is the category of a candidate facet term in F^0. For example, suppose R_q contains the top-7 query result snippets, Fig. 1(a) shows the covering relationships between the 5 candidate facet terms and R_q, and the category hierarchy of the facet terms up to two levels.



(a) (b) (c)

Fig. 1. Example of facet hierarchies

Definition 1 (A facet hierarchy of query results). A facet hierarchy $H(H.root, N, E)$ of the query result R_q is an ordered tree structure, where $N \subseteq R_q \cup F^0 \cup C_{\mathrm{F}}$, $E \subseteq E^0 \cup \mathcal{E}_{\mathrm{F}} \cup (\{H.root\} \times N)$. Besides, $R_q \subseteq N$.

[Example 1]. Fig. 1(b) and (c) show two examples of facet hierarchy. Fig. 1(c) shows the case that no facet term is selected to construct the facet hierarchy H_2, which corresponds to a ranked list of the search results. On the other hand, there are two facet terms and a category facet term selected to construct the facet hierarchy H_1, where s_6 and s_7 are the remained search results not covered by the selected facet terms.

A facet hierarchy shows an organized hierarchy of showing the search results, similar to the directory of a file system. A browsing path simulates a browsing behavior of users to retrieve the snippet of a search result. We assume that a user will browse the children nodes of each node one-by-one according to the sorted order. Moreover, after selecting a node, the user will recursively browse the children nodes until finding the required result.

Definition 2 (A browsing path of a snippet). A browsing path of a snippet s_i on the Hierarchy H is a node visiting sequence from *H*.root to the leaf node representing s_i. The set of all browsing paths of a snippet s_i on the hierarchy H is denoted as $path^H(s_i)$.

Definition 3 (The browsing cost of a snippet). The browsing cost of a snippet s_i in a facet hierarchy H is the number of nodes in the shortest browsing path of s_i in H except *H.root.*

$$cost_H(s_i) = \min\{len(path) | \, path \in path^H(s_i)\},$$

where *len*(*path*) denote the number of nodes in *path* except *H.root.*

[Example 2]. As shown in Fig. 1(b), there is only one browsing path to retrieve the snippet s_2, denoted as $path^{H_1}(s_2) = \{f_1^1 \to f_1^0 \to s_1 \to s_2\}$. The browsing path simulates the user behavior of selecting facet term f_1^1, selecting facet term f_1^0, denying s_1, and then confirming s_2. The browsing cost of s_1 is 4. For the snippet s_4, $path^{H_1}(s_4)$ contains two browsing paths: $f_1^1 \to f_1^0 \to s_1 \to s_2 \to s_3 \to s_4$ and $f_1^1 \to f_1^0 \to f_4^0 \to s_4$. Therefore, $cost_{H_1}(s_4) = 4$. In Fig. 1(c), the browsing cost of s_k equals to k for *k*=1, …, 7, respectively.

According to the definition of the browsing cost of a snippet on a facet hierarchy, our goal is to find a facet hierarchy which provides the least expected browsing cost of the query result.

Definition 4 (Facet hierarchy construction with minimum browsing cost problem). Given the query result R_q and $P(s_k)$, which denote the access probability of query result s_k, where $P(s_n) \geq P(s_m)$ for $n < m$ and $\sum_{i=1}^{k} P(s_i) = 1$. The facet hierarchy construction with minimum browsing cost problem is to find a facet hierarchy H of $R_q$such that *H*=arg min ($\sum_{s_i \in R_q} P(s_i) \cdot cost_{H_j}(s_i)$).

4 Topic Term Extraction

4.1 Entity Mention Annotation

Based on our observation, most important topic words belong to nouns or noun phrases. Accordingly, for each snippet s_i, we use Stanford POS tagging tool and the NER library to parse the title and description of the snippet. Besides, we use the TAGME API provided on the official website [14] for finding the entity mentions.

Let $s_i.title$ and $s_i.description$ denote the set of Wikipedia entities extracted from the title and the description of s_i, respectively. From the terms in $s_i.title$ and $s_i.description$, a term t_j in $s_i.title$ is selected to be a *topic term* of s_i and inserted into $s_i.facets$ if t_j is a noun or t_j is recognized as a named entity by the Stanford NER library. According to the same rule, a term t_j in $s_i.description$ is selected to be a *context term* of s_i and inserted into $s_i.context,$ where the context terms are used to provide more semantic information for selecting the categories of facet terms.

4.2 Candidate Facet Terms and Categories Generation

After extracting the topic terms of each snippet, those topic terms are the candidate facet terms on the lowest level of the hierarchy. Accordingly, F^0 is set to be the union of $s_i.topic$ for each s_i in R_q. Besides, the categories of the topic terms in Wikipedia are looked up to find the candidate category facet terms.

For each facet term f_j^0, let $C^1(f_i^0)$ denote the set of categories of f_i^0 in Wikipedia. By recursively looking up the directory structure of Wikipedia, $C^l(f_i^0)$ denote the union of the parent categories of $C^{l-1}(f_i^0)$ for $l > 1$. For example, for a candidate facet term f_1^0 ="BlackBerry Messenger", $C^1(f_1^0)$ consists of "BlackBerry(company)", "BlackBerry_software", "Instant_messaging" and "Instant_messaging_clients". Besides, $C^2(f_1^0)$ consists of "BlackBerry(company)_mobilephone", "Canadian brands", etc." Among the categories of the topic terms, only some categories are highly semantics related to the search result. For example, "BlackBerry", "Instant messaging" and "Cloud clients" are highly semantics related to the search result, but "2007 introduction" and "Digital audio players" are not. In order to address this problem, we perform a learning-to-rank approach to select the categories with highly semantic relatedness to the search result.

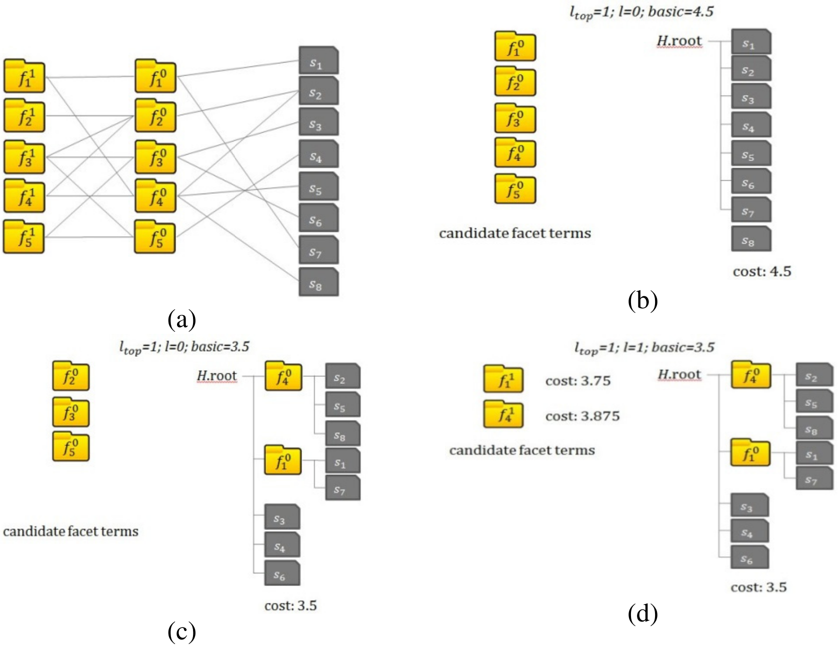
Let C^l(s) denote the union of $C^l(f_i^0)$ for each f_i^0 extracted from snippet *s*. We extract two groups of features: the *context aware features* and *popularity features* for each category in C^l(s). The context aware features of a category aim to evaluate the semantics related degree between the category and the query result *s*. Moreover, the popularity features of a category represent the popularity of the category.

We apply the method proposed in [5] to select two sets of children articles and split articles to represent the semantics of a category *c*. Besides, we apply the formula provided by [5] to compute the semantic relatedness between two Wikipedia articles. For each category c in C^l(s), the 9 features shown in Table 1 are extracted and used to rank the categories at level *l*. A parameter *max_l* can be given to control the number of levels in the constructed facet hierarchy.

Table 1. The features for ranking the categories of facet terms

Features	Description	Notation
Context Aware Features	Average semantic relatedness between children articles of the category and the topic terms of the search result	*CT*
	Average semantic relatedness between children articles of the category and the context terms of the search result	*CC*
	Average semantic relatedness between the split articles of the category and the topic terms of the search result	*ST*
	Average semantic relatedness between the split article of the category and the context terms of the search result	*SC*
Popularity Features	Number of children articles of the category	*NC*
	Word length of the category title	*WL*
	Character length of the category title	*CL*
	Number of split articles / word length of the category title	*SP*
	Have the corresponding main article	*MA*

In the training phase, for a snippet s, a manually ordered list of the categories in $C^l(s)$ for $l \geq 1$ is given according to their relevance with s. Then we use rankSVM to learn a model denoted $model^l$ for ranking the categories at level l. For each snippet s_i in R_q, the $model^1$ is used to select the top-K categories from $C^1(s_i)$ for each candidate facet term of s_i. Recursively, the $model^l$ is applied to select the top-K categories of a category facet term in F^{l-1} as F^l for $l > 1$, which are used to be the candidate category facet terms for constructing the facet hierarchy.



(a)

(b)

(c)

(d)

Fig. 2. An example of the FH-BU algorithm

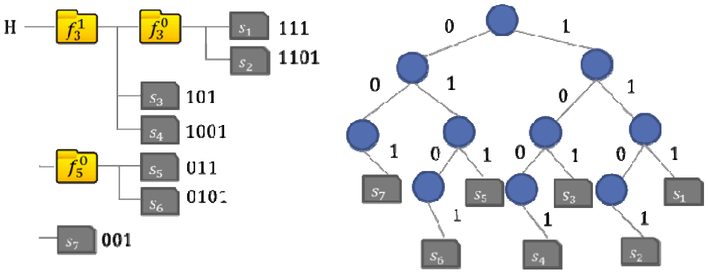


Fig. 3. The correspondence between a facet hierarchy and a binary encoding tree

5 Facet Hierarchy Construction

5.1 The Bottom-Up Approach

The bottom-up approach, which is named the FH-BU algorithm, applies a greedy approach to incrementally select the facet terms and categories into the facet hierarchy. Initially, the facet hierarchy *H*(*H.root*, *N*, *E*) has *N* = $R_q$and *E* =$\{H.root\}\times R_q$. Let*f.cover* denote the search results in R_q covered by a facet term *f* in F^0. For each facet term *f* in F^0, the browsing cost of the resulting facet hierarchy *H'* by inserting *f* as the parent node of *f.cover* is computed. The facet term whose resulting facet hierarchy has the lowest browsing cost, say *f**, is selected. After the facet term *f** is inserted into the facet hierarchy *H*, the process is repeated to select the next facet term from F^0 to be inserted into the facet hierarchy until the browsing cost couldn't be further reduced by adding another facet term in F^0. Then the set of categories of the selected facet terms is denoted by F^1. The similar process is performed recursively to select the category facet term in F^l for *l* ≥1 to be inserted into the facet hierarchy until the browsing cost couldn't be further reduced.

[Example 3]. Fig. 2(a) shows an example of 8 search results with 5 candidate facet terms in F^0 and their corresponding 5 categories. We assume that the access probability of each search result is uniformly distributed, i.e. the access probability of each search result is 0.125. Initially, the facet hierarchy is shown as Fig. 2(b). Then the facet terms are selected from F^0 to be inserted into the facet hierarchy incrementally, that leads to the facet hierarchy shown in Fig. 2(c). After that, the categories of the facet terms f_1^0 and f_4^0, i.e. f_1^1 and f_4^1, are considered to be inserted into the facet hierarchy. However, the browsing cost of the constructed facet hierarchy could not be reduced by adding any category facet term. Accordingly, the constructed facet hierarchy is shown as Fig 2(d).

5.2 The Top-Down Approach

For each browsing path, we can label the user behavior of choosing a facet term and confirming a search result as "1", and "skipping a facet" or "skipping a search result" as "0". Therefore, each browsing path can be encoded as a bit sequence. The browsing cost of each search result is equal to the length of the bit sequence. In other words, the encoding of each browsing path of a facet hierarchy has a corresponding binary encoding tree as shown in Fig. 3.

Accordingly, given $R_q = \langle s_1, s_2, \dots, s_k \rangle$ and their access probabilities, the facet hierarchy construction with minimum browsing cost problem is similar to construct an encoding tree which generates the minimum expected length of codes. Besides, according to the definition of a facet hierarchy, the corresponding encoding tree T_q must satisfy the following two requirements:

(1) The encode bit sequence of each leaf node in T_q is ended with "1".
(2) Each internal node in T_q has a corresponding facet term in $F^0 \cup C_F$.

We apply the concept of Shannon–Fano coding method [10] to measure the information entropy of the access probabilities for the snippets covered and not-covered by a facet term *f* as following:

$$Split(f,S) = -(f.p_r(S) \times log_2(f.p_r(S)) + f.p_u(S) \times log_2(f.p_u(S))),$$
$$f.p_r(S) = \frac{f.p\_cover(S)}{f.p\_sum(S)}, \quad f.p_u(S) = \frac{f.p\_uncover(S)}{f.p\_sum(S)},$$

where $f.p\_cover(S) = \sum_{s_i \in S \wedge s_i \in f.cover} p(s_i)$, $f.p\_uncover(S) = \sum_{s_i \in S \wedge s_i \notin f.cover} p(s_i)$, and $f.p\_sum(S) = f.p\_cover(S) + f.p\_uncover(S)$.

A higher *Split* value of *f* implies that the access probabilities of the covered and not-covered search results are more balanced. In other words, it is more possible to get a lower expected browsing cost by using *f* as a facet term for *S*.

Accordingly, the top-down approach for constructing the facet hierarchy, named the FH-TD algorithm, selects the facet term which has the highest *Split* function value to split the search results in R_q into two subsets. The same procedure is performed recursively to split the two subsets of the search results. However, the *Split* function only estimates the distribution of the access probabilities of the covered and not-covered search results of one level. In order to improve the effectiveness of estimation, a positive integer parameter *d* can be given to further compute the *Split* function values on the following d levels of facet terms under the current facet term.

Initially, S is equal to R_q. For each facet term *f* in $F^{\max-1}$, *f* and the following d levels of facet terms under *f* are evaluated by the Split function and compute the average value. The facet term, say *f'*, and its decedent facet terms with the highest average *Split* value is selected into the facet hierarchy. Then S is separated into *f'.cover* and (S- *f'.cover*). Accordingly, the same procedure is performed on *f'.cover* and (S- *f'.cover*) recursively to find the other facet terms to be inserted into the facet hierarchy until finishing the insertion of the facet terms in F^0.

6 Performance Evaluation

In this section, we compare the two proposed methods with the method used in Facetedpedia [7]. The algorithms were implemented using JAVA in the Eclipse platform and performed on a personal computer under the Microsoft Windows 7 environment with 8 GB RAM. The test query set is provided by "Web Track 2012 and 2013" in the TREC competition, where there are 50 queries in both "Web Track 2012" and in "Web Track 2013", respectively. Given a query *q*, the snippets of the search results are collected by the Google search engine for each test query. Moreover, the access probability p_i of each snippet s_i is simulated according to a uniform distribution and an exponential distribution, respectively.

The default setting of the parameters in the experiments is as follows. The level of facet categories *max_l* is 2. The maximum number of facet terms at the top-level, i.e. level 2, is set to be 8. The look-ahead parameter *d* of the top-down method is set to 2.

[Exp. 1]. The effect of changing the number of search results
In this experiment, the expected browsing costs of the constructed facet hierarchies are compared by changing the number of search results. The baseline is the browsing cost of the ranked list of search results without using a facet hierarchy. The results performed on the query results with uniformly distributed probabilities and exponentially distributed probabilities are shown in Fig. 4(a) and 4(b), respectively.

From the results, it shows that the facet hierarchies constructed by the proposed methods have lower expected browsing costs than the ones constructed by the Facetedpedia. When the access probabilities of the search results are uniform distribution, the facet hierarchies constructed by FH-BU and FH-TD have similar browsing costs. However, when performed on the access probabilities with exponential distribution, the performance of FH-TD is better than FH-BU. When the number of search results is more than 50, the constructed facet hierarchies can save more than half of the browsing costs of the baseline method. The gain of browsing cost saving is more significant when the number of search results increases.

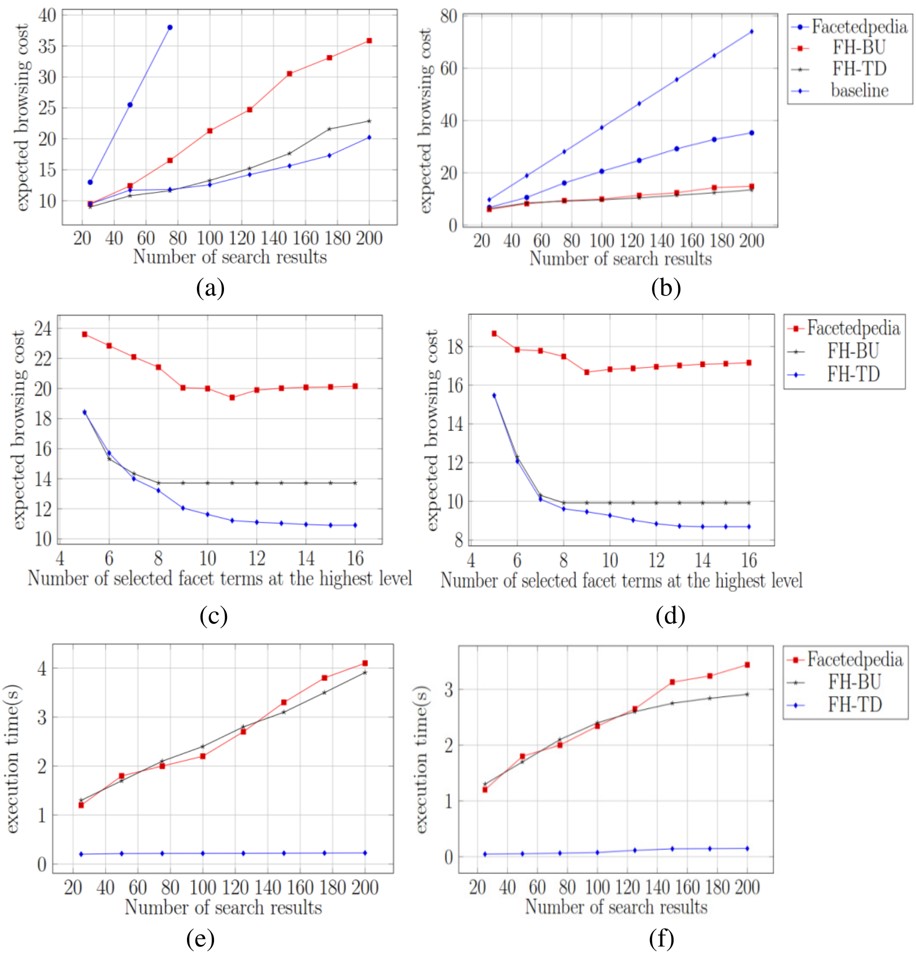
[Exp. 2]. The effects of changing the maximum number of categories selected at the highest-level of the facet hierarchy.

In this experiment, the expected browsing costs of the constructed facet hierarchies are compared by changing the maximum number of categories at the highest level. The number of search results is set to be 100.

Fig. 4(c) and 4(d) show the results performed on the query results with uniformly distributed probabilities and exponentially distributed probabilities, respectively. From the results, it demonstrates that the browsing costs of the constructed facet hierarchies will decrease as the maximum number of categories at the highest level increase. For the FU-BU, the browsing costs keep stable when the maximum number of categories at the highest level is 8. It implies that the FU-BU couldn't add another category facet term to further reduce the expected browsing cost of the constructed facet hierarchy. On the other hand, the expected browsing costs of the facet hierarchies constructed by FU-TD continue to decrease until the maximum number of categories at the highest level is larger than 12.

[Exp. 3]. Evaluation on the execution time
In this experiment, the execution time of the proposed algorithms is observed. The number of search result is fixed to 100 and the maximum number of categories at the highest level is set to be 8.

The FH-TD estimate the expected browsing cost by using the entropy function without tracing the hierarchy and calculating the expected browsing cost of the resultant facet hierarchy. Accordingly, as shown in Fig. 4(e) and 4(f), the execution speed of FH-TD is much faster than the other twos.



(a) (b)

(c) (d)

(e) (f)

Fig. 4. The experimental results

7 Conclusion

In this paper, we propose a method to dynamically construct a faceted interface for browsing the search results efficiently. A learning-to-rank approach is used to select the query-dependent facet terms for constructing the facet hierarchy. Moreover, two greedy based algorithms, one is a bottom-up approach and another one is a top-down approach, are proposed to construct a facet hierarchy for minimizing the expected browsing cost as possible. The experimental results show that the facet hierarchies constructed by the two proposed methods both save more than half of the browsing costs of the baseline method. Moreover, the top-down approach can achieve better performance on the constructed facet hierarchy and spend less computing time than the bottom-up approach.

References

1. Agarwal, D. B., Chen, C.: fLDA: matrix factorization through latent dirichlet allocation. In: The Third ACM International Conference on WSDM (2010)
2. Blei, D.M., Ng, A.Y., Jordanm, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research (2003)
3. Carpineto, C., Osiński, S., Romano, G., Weiss, D.: A survey of web clustering engines. ACM Computing Surveys (CSUR) **41**(3), July 2009
4. Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J.: Stochastic variational inference. The Journal of Machine Learning Research **14**(1), January 2013
5. Jiang, P., Hou, H., Chen, L., Chen, S., Yao, C., Li, C., Wang, M.: Wiki3C: exploiting wikipedia for context-aware concept categorization. In: The Sixth ACM International Conference on Web Search and Data Mining (WSDM) (2013)
6. Kong, W., Allan, J.: Extracting query facets from search results. In: The 36th International Conference on Research and Development in Information Retrieval (2013)
7. Li, C., Yan, N., Roy S. B., Lisham, L., Das, G.: Facetedpedia: dynamic generation of query-dependent faceted interfaces for wikipedia. In: The 19th International Conference on World Wide Web (WWW), pp. 651–660 (2010)
8. Mei, Q., Shen, X.C., Zhai, X.: Automatic labeling of multinomial topic models. In: The 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2007)
9. Scaiella, U., Ferragina, P., Marino, A., Ciaramita, M.: Topical clustering of search results. In: The Fifth ACM International Conference on Web Search and Data Mining (2012)
10. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal **27**, 379–423 (1948)
11. Vosecky, J., Jiang, D., Leung, K. W., Ng, W.: Dynamic multi-faceted topic discovery in twitter. In: The 22nd ACM International Conference on Information and Knowledge Management (2013)
12. Zhu, J., Ahmed, A., Xing, E.P.: MedLDA: maximum margin supervised topic models. The Journal of Machine Learning Research (2012)
13. Zhu, X., Ming, Z.Y., Zhu, X., Chua, T.S.: Topic hierarchy construction for the organization of multi-source user generated content. In: The 36th ACM International Conference on Research and Development in Information Retrieval (SIGIR) (2013)
14. http://tagme.di.unipi.it/tagme_help.html