Faceted Search using Bootstrap List Extraction Method

Shery T. S.
M.Tech Student
Department of Information Technology
Cochin University of Science and Technology (CUSAT)
Kerala, India

Sreeja M. U.
Research Scholar
Department of Information Technology
Cochin University of Science and Technology (CUSAT)
Kerala, India

Binsu C. Kovoor
Assistant Professor
Department of Information Technology
Cochin University of Science and Technology (CUSAT)
Kerala, India

ABSTRACT

The traditional way of interaction between users and search engines has changed a lot by the invention of faceted search. Earlier, the user had to enter keywords on search engines and the search engine returns a set of web pages on the basis of input keyword. The user has to traverse through these web pages to identify the relevant information. Also, the search results are multifaceted which further reduces clarity in the results. The proposed system presents a systematic solution for finding query facets from top results on search engines by utilizing list extraction algorithm. This helps the users to find the right information without searching a large number of pages. The paper proposes Bootstrap technique in the list extraction phase which will add more results into the extracted list.

Keywords

Query facets, faceted search, clustering, multifacets, bootstrap method.

1. INTRODUCTION

In the last few years, user interactions with the web search engines have not undergone major changes. Queries are issued manually and resultant pages are reviewed. The maximum noticeable change has been the introduction of verticals (e.g. snap shots, films, and news), question auto complete, and question answering. The invention of faceted search helps the users to find right information in a short span of time [1]. Since faceted search is common in e-commerce sites like Amazon, eBay, etc. most online users are now familiar with the concept. However, it would be highly beneficial if properly utilized for general web search engines.

A query facet is a list of items that gives descriptions and details about a particular query. A facet can be a word, phrase or even a sentence. And a single query may have other related facets that describe the query in different point of views. It is a difficult task to automatically mine the facets for a query. Since queries may be multifaceted, users have to visit tens of pages to find the right information. For example, the query Jericho is an ambiguous query. It may refer to a place, a person or an American drama series. If the user is referring to place, lots of pages have to be traversed unwantedly to find the relevant information which is time consuming. Query facets give advanced internet browsing experience by providing fascinating information in convenient fashion. First, query facets are displayed collectively with the original search results in a suitable manner. Therefore, users can discern some critical aspects of a query without surfing tens of pages. Second, query facets may additionally provide direct data or instantaneous answers that users are looking for. As an example, for the query lost season, all episode titles are displayed in a single aspect and essential actors are shown in another. In this case, exhibiting query facets could save searching time. Third, query facets can also be used to improve the diversity of the ten blue hyperlinks. Search results can be re-ranked to avoid displaying duplicate query facets. Apart from normal search engines, structured information included in the query facets can effectively be utilized in search engines with different underlying technologies like semantic search engines or entity search engines [2].

Faceted interfaces constitute a new powerful paradigm which has been established to be a supplement to keyword searching. Until now, the technology of faceted interfaces depended on both the manual identification of the facets and on previous expertise of the facets that can probably appear inside the text series. Users who want to locate information on the web commonly rely on one of the following paradigms. An immediate, keyword based search is used or the contents of the internet are browsed through to discover items that are relevant.

In faceted interfaces, users can expand a particular facet in the hierarchy to a sure factor after which the web results can be sliced and browsing can be switched to some other hierarchy. Such multifaceted interfaces expose the contents of the underlying web and aids users to locate items of interest swiftly. Up to now, the systems that use faceted interfaces are constructed manually. One of the crucial tasks required to permit extensive deployment of faceted interfaces is to construct strategies for automated development of faceted interfaces. It is clear that the user experience for structured web search can advantage from facets. For a search engine to effectively utilize facets two challenges need to be addressed. (a)Given the restricted screen display property and the huge range of possible facets, it is necessary to select the top-k vital facets, where k is generally a small quantity [1]. Facet significance can be measured by using the application of a facet towards a person's predicted action like a pivot or refinement. Since an entity could have one hundred attributes, the task here is to find the most important attributes and values with maximum anticipated utility. (b) There is a huge quantity of structured data sources currently available to engines like Google or Amazon. If the information is summarized as de-normalized entity-type tables, there will be thousands of such tables to remember. So any resolution that finds major attributes from these tables need to be totally automatic. The proposed system describes a systematic solution for this issue.

The proposed system works on manually developed dataset. The dataset is built from queries formulated by a set of users. Related lists for each query are found out and the dataset is built based on these queries. When a user gives a query, the system will automatically extract the corresponding lists from the dataset. This approach makes use of the model proposed by Dou et al. [2]. The proposed system introduces an additional step in the list extraction phase.

A summary of related work is described here. Wei et al [3] details the main features of existing faceted search systems. In addition, the performance of related faceted search methods and techniques in all phases is evaluated and described. Dou et al [4] developed QDMiner system to automatically extract facet hierarchies for keywords by aggregating frequent lists from free text, HTML tags, and reiterated regions within top search results. Anju G R and Karthik M [5] proposed a system implementing graphical model which is a supervised method. This paper implements pattern- based semantic extraction on top ranked web documents. The algorithms used are QF-I and QF-J. Li et al [6] developed a system named Facetedpedia that exploits internal hyperlinks of Wikipedia and folksonomy for automatic extraction of facet terms. Stoica et al [7] implements Castanet set of rules to select facet terms based on term frequency distribution. Ling et al [8] details a two-level probabilistic technique to extract facet phrases based on topic version. Andrew.C et al [9] describes a bootstrapping technique for semi-supervised learning to extract categories and relations from web pages.

The remaining paper is organized as follows. In section 2, the implementation of QDMiner along with the implementation of the proposed system is described. This is followed by results and discussions in section 3. The conclusion and future scope of the proposed system is included in section 4.

2. METHODOLOGY

The existing system, QDMiner [2] developed by Dou et al. gives a systematic solution for finding query facets. It is based on aggregating top search results to mine related facets for a particular query. Here, a query facet is a set of objects which describe and summarize one vital thing of a query. A facet item is typically a word or a phrase. A query may have more than one facet that summarizes the information about the query from specific perspectives. For example, the query ‘watch’ includes the knowledge about watches in five particular elements, which includes brands, gender categories, supporting features, styles, and colorations.

The proposed system is a modified version of the system described in [2]. The process flow of the proposed system is shown in Figure 1 with the four steps: list extraction, list weighting, list clustering, facet and item ranking.

The difference in implementation of the proposed system from the existing system is the introduction of Bootstrap list extraction technique in the list extraction phase. The top results for a query are retrieved and all documents as a set are input to the system followed by the following steps.

- List extraction using bootstrap method
- List weighting
- List clustering
- Facet ranking

2.1 List Extraction:

For each document in the input set, their context is extracted based on three patterns which are: free text, HTML tag and repeat region patterns. It is saved as a list. Later, useless symbol characters are removed and uppercase letters are converted to lowercase as part of normalization of all items. Lists with less than two unique items or greater than 200 unique terms are removed. But this process only extracts low grade lists and the number of duplicate elements is on the higher side. If there is any trade within the internal structure of the internet page along with the web page hyperlink or the addition of items to the internet site may additionally produce low quality lists. Generally, search engines display results as lists that may contain duplicate content. Hence, the extraction of useful facets from these lists affects the time complexity of the system. If the extraction algorithm used is efficient enough to eliminate duplicate content it aids in saving time. Hence, in the proposed system bootstrap list extraction technique is introduced to extract high quality lists from the manually developed data. The system uses a semi supervised boot-strapping approach that automatically extracts and classifies the entities. When a user search a query on the search engine the bootstrap algorithm will work automatically and it extracts more lists than the existing approach from the dataset.

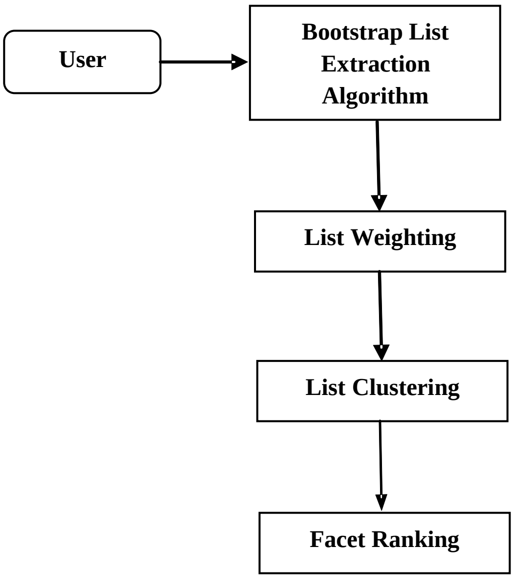

Fig 1: Process Flow for Facet Ranking

2.2 List Weighting:

The extracted lists may contain several information that may be useful or useless. Useless lists include navigational links, extraction error list, etc. These are not relevant to the query. Hence, a method should be implemented that calculates the importance of each list in the current scenario. It is proportional to document matching weight (calculated from percentage of items in the documents and importance of document depending on rank) and average invert document frequency.

2.3 List Clustering:

The individual weighted lists retrieved from the above step cannot be considered as query facets. This is because it contains noise, only small number of facets and duplicate elements. So, in order to make the results more accurate and useful, clustering is performed. Clustering is grouping of similar lists together. If majority of items are shared among two lists, they can be clustered.

2.4 Facet and Item Ranking:

After possible query facets are extracted from the above step, the significance of facets and items are calculated and ranked accordingly. To calculate the importance of facets, two models, Unique Website Model and Context Similarity Model are utilized. Unique Website Model considers the fact that information from a single website will be similar. So, only one weight is considered for calculating importance of facet from a unique site. In Context Similarity model, the presence of duplicate information is owing to the presence of mirror websites, republished content and same publishing software. The above factors end up in generating duplicate lists. To solve these issues, list duplication estimation or list grouping method is adopted. Item ranking is the final step where the rank of an item depends on the number of lists that contain the item. If the item is better, the rank will be higher and vice versa.

The proposed approach starts with the trained dataset. The training set of documents is manually annotated with the named entity categories. The annotated training set is pre-processed by identifying the features of the word. The context of the word is analyzed to define the pattern. The patterns associated with each category of named entity are identified and used as seed patterns. The test data is processed by matching the features of the word with the pattern. If exact match occurs then the named entity category is identified. Up to this point named entities have been identified and categorized that have features exactly similar to the seed set initially given [8].

3. RESULTS AND DISCUSSION

The experiment was conducted on manually created dataset since offline dataset was not readily available. The dataset was built by collecting queries from a set of users. From this, the user selects most frequent queries. Related items for the selected query are manually created and stored in the dataset. Every time the system is executed, it goes to the dataset location and automatically extracts the facets for the particular query.

The quality of query facet depends primarily on the time taken for list extraction and accuracy of the query facets. Accuracy of the extracted facets depends on the number of relevant facets extracted. Hence, time taken for list extraction and the number of facets extracted can be used to evaluate the system performance.

The comparison of the number of facets extracted in faceted navigation systems with and without bootstrap method is shown in Figure 2. The graph is plotted with the number of clusters on X-axis and number of facets extracted per cluster on Y-axis.

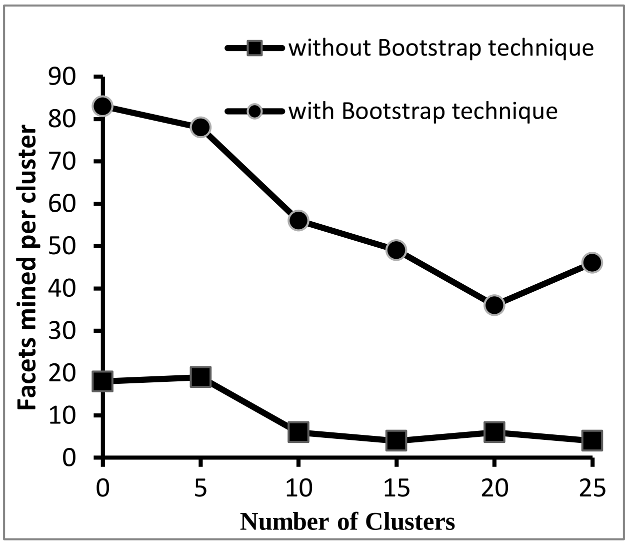

Fig 2: Comparison based on number of facets extracted

From the graph, it can be inferred that the bootstrap technique contains fewer clusters with query facets having less noise. The number of facets extracted per cluster is comparatively high. Ideally, the number of facets shouldn't be too few or too large. If it is too few there won't be any reduction in the search space and if it is too large, it will be a hectic task for the users to find the relevant facets and apply them. But in the proposed system, the extracted facets will be just the highly relevant ones. It can also be seen that the number of facets is comparatively high. But since the number of clusters is less, the number of facets won't be too high. Hence, it can be said that the accuracy and relevance of the extracted facets in the proposed system is higher than that of the existing system.

Time is a crucial factor in all web transactions. If the user can't find the right content quickly and easily, it is likely that the user won't get the right information at all. The search engine which takes the lowest time to retrieve relevant results will be considered the most efficient one. Hence, time taken by the system for list extraction and thereby facet creation can be considered as another evaluation criteria for assessing the performance of the system. The comparison of time taken to return results in faceted navigation systems with and without bootstrap technique is depicted in Figure 3. The range of time is taken in the order of Nano seconds. It can clearly be noted that the proposed system takes significantly less time when compared with the existing system.

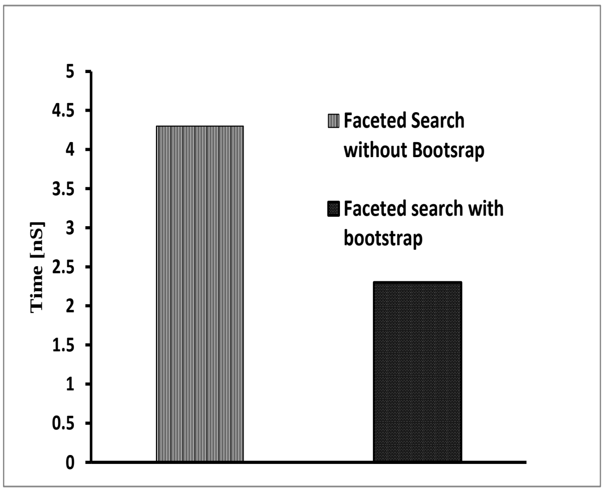

Fig 3: Comparison of time for faceted search

4. CONCLUSION AND FUTURE WORK

This proposed system provides a systematic solution for finding query facets from search engine. It is implemented based on the system proposed by Dou et al in [2]. Additionally, the proposed system uses a bootstrap list extraction method to avoid noisy lists during list extraction process. It can be seen that the proposed system shows improved performance and accuracy than QDMiner. The addition of Bootstrapping in the list extraction phase yields impressive results by retrieving better query facets and reducing duplicate facets. The only drawback of the proposed system is the manual creation of dataset which is time consuming. This disadvantage can be alleviated if online datasets are utilized. Further, the same technology can successfully be applied to different search engines like semantic search engines and entity search engines.

5. REFERENCES

[1] Friedrich, J., Lindemann, C. and Petrifke, M. 2015. Utilizing Query Facets for Search Result Navigation. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 271-275.

[2] Dou, Z., Jiang, Z., Hu, S., Wen, J. R. and Song, R . 2016. Automatically Mining Facets for Queries from Their Search Results IEEE Transactions on Knowledge and Data Engineering. 28(2), 385-397.

[3] Wei, B., Liu, J., Zheng, Q., Zhang, W., Fu, X., Feng, B.2013. A survey of faceted search. In: Journal of Web Engineering, 12(1-2), 41-64.

[4] Dou, Z., Hu, S., Luo, Y., Song, R., Wen, J. 2011. Finding dimensions for queries. In Proceedings of the ACM international conference on Information and knowledge management.

[5] Anju, G. R. and Karthik, M. 2016. Minning Queries From Search Results: A Survey. Imperial Journal of Interdisciplinary Research (IJIR). 2(12), 1840-1842.

[6] Li, C., Yan, N., Roy, S. B., Lisham,.L. and Das,.G. 2010. Facetedpedia: dynamic generation of query-dependent faceted interfaces for Wikipedia. . In Proceedings of the International conference on World Wide Web. 651- 660.

[7] Stoica, E., Hearst, M. A., and Richardson, M. 2007. Automating creation of hierarchical faceted metadata structures. In Proceedings of NAACL HLT. 244-251.

[8] Carlson, A., Betteridge, J. and Wang, R.C., Estevam, R. H., Mitchell, T. M. 2010. Coupled Semi-Supervised Learning for Information Extraction. In Proceedings of the ACM international conference on Web search and data mining.

[9] Daniel, N., Esenther, A. 2009. Semi-Supervised Information Extraction From Variable-Length Web Page Lists. International Conference on Enterprise information Systems.261-266.