Generating Facets Using Website Wrapper

Vaishakhi VK
Computer Science and Engineering
Malabar Institute of Technology
vaishakhivk@gmail.com

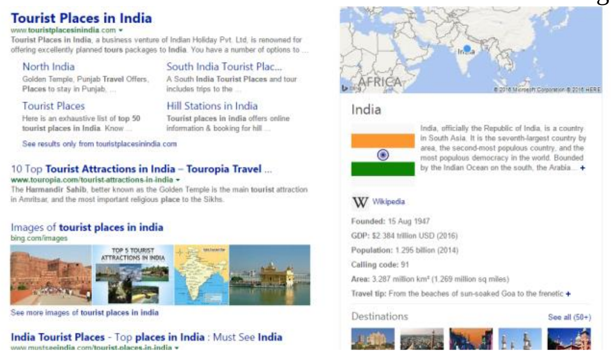
Neethu T Regi
Computer Science and Engineering
Malabar Institute of Technology
neethu.t.regi@gmail.com

Abstract

The query facets are multiple groups of words or phrases that explain and summarize the content covered by a query. When a query is given then from the search results the lists are extracted using the list extraction algorithm, all extracted lists are given weights, unimportant or noisy lists that occasionally occurs in a page are assigned by low weights. Similar lists are grouped into clusters. Facets are then evaluated and ranked. High rank is given for lists extracted from similar context and lists having higher weight. Links are collected and given to the user for detailed result.We address the problem of finding query facets and the navigation to the desired web page. In order to increase the quality of the extracted list we introduce a website wrapper called Anchor based data extraction system. The facets are generated based on the search interest of the user. The experiment is performed real time in Bing using the Bing search API. From the generated facets user can have detailed result by clicking the links.

1. Introduction

Data mining also called data or knowledge discovery is the process of analyzing data from different perspectives and summarizing it into useful information. The patterns, associations, or relationships among all this data can provide information. Information can be converted into knowledge about historical patterns and future trends. Data mining deals with the kind of patterns that can be mined We address the problem of finding query facets and the navigation to the desired web page. A query facet is a set of items which describe and summarize one important aspect of a query. Here a facet is a word . A query can have many faces that summarize the information about the query from different aspects. We use the data mining techniques to for generating facets and to navigate to the desired web page. Query facets provide interesting and useful knowledge about a query and thus can be used to improve search experiences. Users can understand some important aspects of a query without browsing lots of pages. Query facets may provide direct information or instant answers that users are seeking.



Tourist Places in India
www.touristplacesinindia.com
Tourist Places in India, a business venture of Indian Holiday Pvt. Ltd, is renowned for offering excellently planned tours packages to India. You have a number of options to ...

North India
Golden Temple, Punjab Travel Offers, Places to stay in Punjab, ...

South India Tourist Plac...
A South India Tourist Places and tour includes trips to the ...

Tourist Places
Here is an exhaustive list of top 50 tourist places in India. Know ...

Hill Stations in India
Tourist places in india offers online information & booking for hill ...

See results only from touristplacesinindia.com

10 Top **Tourist Attractions in India – Touropia Travel** ...
www.touropia.com/tourist-attractions-in-india
The Harmandir Sahib, better known as the Golden Temple is the main tourist attraction in Amritsar, and the most important religious place to the Sikhs.

Images of **tourist places in india**
bing.com/images
See more images of tourist places in india

India Tourist Places - Top **places in India** : Must See **India**
www.mustseeindia.com/tourist-places-in-india

India
India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, the second-most populous country, and the most populous democracy in the world. Bounded by the Indian Ocean on the south, the Arabia... +

Wikipedia
Founded: 15 Aug 1947
GDP: $2.384 trillion USD (2016)
Population: 1.295 billion (2014)
Calling code: 91
Area: 3.287 million km² (1.269 million sq miles)
Travel tip: From the beaches of sun-soaked Goa to the frenetic +

Destinations
See all (50+)

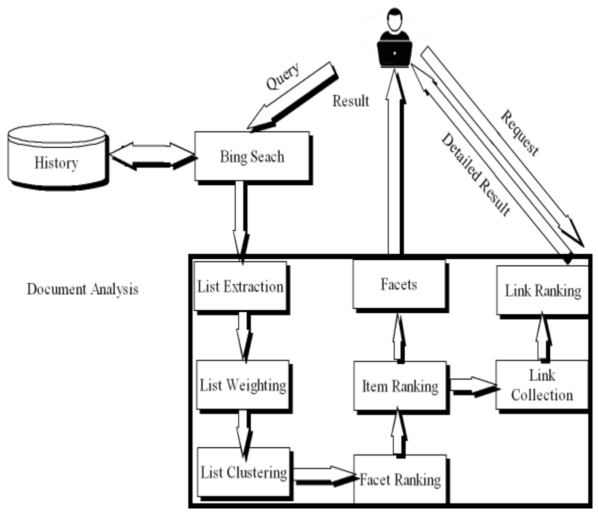
AFRICA
India
TOP 5 TOURIST ATTRACTIONS IN INDIA
© 2016 Microsoft Corporation © 2016 HERE

The facets are generated based on the user's preference. The user can also go to the desired web page by selecting the item from the facets. Thus it saves the user's time wasted on searching for the data in tens or thousands of pages. website wrapper called Anchor based data extraction system for extracting high quality lists. These extracted lists are given weights by using the document matching weight and average invert document frequency. In the clustering the similar lists are then grouped togather. In the ranking step the lists are given higher ranks if it is extracted from unique context and having higher weight. After that the facets are generated. The user can then select the particular item for detailed result. In link ranking step the links are collected and only the top links are displayed to the user.

2. Related Work

.Luo et al. proposed Application of Internet Technology and Web Information extraction wrapper based on DOM for Agricultural Data Acquisition [3].It is the method of Web Information extraction wrapper based on DOM. Combining X-Path and pattern matching, it can deal with the two type of information at the same time under the guide of source and target knowledge library. Information extraction method is actually a text processing method. Rauch et al. proposed Knowminer Search - a Multi-Visualisation Collaborative Approach to Search Result Analysis[4].Since the information provided on the internet is large It becomes difficult for the user to get apt information. Here faceted search interface provides the possibility to coherantly reduce the search result set. Friedrich et al. proposed Utilizing Query Facets for Search Result Navigation[5]. Facets provide a way to examine and go through the search result space. Features that rank facets based on their usefulnes to partition the search result documents. A very successful idea to generate facets for HTML documents is based on the extraction of lists from HTML pages. Simonini et al. proposed Big Data Exploration with Faceted Browsing[6]. Big data analysis now manage nearly every point of modern society. One of the most valuable means through which to make meaning of big data, and thus make it more helpful to most people, is data visualization. The faceted search allows the user to detail a query progressively, seeing the effect of each choice inside one facet on the available choices in other facets.

3. System Architecture



4. Module Splitup

- **Search**

When a query is given to the Bing search engine the query is processed. After query processing the results are stored in the Bing search container.

- **Anchor**

The anchors are selected, which are textual elements that identify the start or end of a document. HTML tags such as the < head > , < b > etc are selected as anchors. After that region patterns for the anchors are selected. Regions are selected based on the tags such as the < select > < div > etc.

- **Facets**

The anchors and the regions act as the input to the data extraction algorithm. The extracted list is given weight. After that the similar lists are clustered. Then the lists are ranked. After that the items are ranked. Facets are generated.

- **Facets Link**

Facets contain items, user can select any item to get detailed information. Then the corresponding links are collected. After that the links are ranked and the top links are given to the user.

5. Algorithms

- **Algorithm** 1 BING SEARCH

```
Input : Query
Output: Documents
1: function search(query)
2: Enter the query
3: Process the Query
4: Store the results in Bing Search container
5: end function
```

- **Algorithm** 2 Anchors and Region Patterns

```
Input : Documents
Output: Anchors and Region Patterns
1: If the tag <head> or <b> or <mark> then
2: Set tag as anchor
3: If tag <Select> or <div>found
4: Then Extract all region between start and end of that tag.
5: If tag < a > found then
6: remove the region between< a > and< =a > from considering
```

- **Algorithm** 3 List Normalization

```
Input : Anchors and Region Patterns
Output: Normalized Lists
1: function Normalization(list)
2: Create an empty list liwords.
3: Collect the result from the previous session.
4: Store it in the datatable.
5: Perform normalization of the result.
6: Create a for loop with i=0,until i < no of rows in the datatable.
7: Remove the stop words and repeating words and store in an array.
8: Compare the result in the list and the array.
9: If any word not present then add it into the list liwords.
10: end for
11: Store the list in the session.
12: Store the whole result into session.
13: end function
```

- **Algorithm** 4 List Normalization

```
Input : Normalized Lists
Output: Preprocessed Lists
```

```
1: function Preprocessing(list)
2: In the preprocessing stage collect the result from the previous session.
3: Create a for loop with i=0,until i < no of rows in the datatable.
4: Remove null spaces and ing ion s es etc from the wrord.
5: end for
6: Collect the preprocessed result into a list
7: Store the result into the session.

8: end function
```

- **Algorithm 5** List Extraction

```
Input : Preprocessed Lists
Output: Extracted Lists
1: function Extraction(list)
2: Collect the result from the previous session.
3: Store it in a datatable.
4: Create a for loop with i=0,until i < no of rows in the datatable.
5: Clear the list liwords
6: Store the items in the document in an array.
7: Create a for loop with j=0,until j < array length.
8: Check if liwords contain the items in array,if no then add it.
9: end for
10: end for
11: Store the result into the session.
12: end function
```

- **Algorithm 6** List Weighting

```
Input : Extracted Lists
Output: Weighted Lists
1: function Weighting(list)
2: Create a list cmnelmnts,Set ttfrqAr1=0,ttfrqAr2=0,tf=0
3: Add common elements count of Arr1 to cont1 and Arr2 to con2.
4: Create a for loop i = 0,i < cntar1.Length
5: Calculate ttfrqAr1 = ttfrqAr1 + cntar1[i]*cntar2[i]
6: Create for loop j = 0, j <cntar2.Length
7: Calculate ttfrqAr2 = ttfrqAr2 + cntar2[j] *cntar2[j]
8: Calculate tf = tf + cntar1[j] * cntar1[j]
9: Calculate b = (ttfrqAr1 ) / (tf)
10: Calculate c = (ttfrqAr1) / ttfrqAr2
11: Calculate sm= Max(b, c)
12: if (sm > 1),sm=1
13: return sm;
14: end function
```

- **Algorithm 7** List Similarity

```
Input : Weighted Lists
Output: Similar lists
1: function Cluster(list)
2: Collect the result from the previous session.
3: Store it in a datatable.
4: Create a for loop with i=0,until i < no of rows in the datatable.
5: Create a for loop with j=i+1,until j < no of rows in the datatable.
6: If i != j then
7: Store the string in i to s1
8: Store the string in j to s2
9: Similarity sm = common elements in S1 and S2.
10: end If
11: end for
12: end for
13: Store the result into the session.
14: End function.
```

Algorithm 8 List Clustering and Ranking

```
Input : Similar Lists
Output: Facets
1: function Clustering(list)
2: Collect the result from the previous session.
3: Store the items in datatable into list cls
4: Create a new list frslst ,smn
5: Create a for loop with i=0,until i < no of rows in the datatable.
6: Check whether frslst contain item in doc 1 in datatable.
7: If not then add it.
8: Check whether frslst contain item in doc 2 in datatable.
9: If not then add it.
10: end for
11: end function
12: Create a new list ds
13: Create a for loop i = 0, i < smn.Count
14: Create a for loop j = 0, j < smn.Coun
15: If ds does not contain elements in smn then add it.
16: Create a list clsnew and ind.
17: Create a for loop i = 0,i < ds.Count
18: Check if ds contains the item in datatable
19: Check if ind contains the item in datatable if not then add it.
20: merge lists cls and ind.
21: end function
```

Algorithm 8 Link Collection and Ranking

```
Input : Links
Output: Detailed result
1: function collection(link)
2: Choose an item i
3: Collect all links corresponding to i
4: Store links
5: end function
6: Provide the top links to user
```

6. Conclusion

This work presents a method of generating meaningful facets from the user query search results. The facets here are generated using four steps. List extraction, list weighting, list clustering and list ranking. Here the list extraction performed using a website wrapper called Anchor based data extraction system and the patterns are generated for these anchors. Then from these generated patterns lists are extracted using the list extraction algorithm. This method can extract high quality lists from the top k query search results. Hence these high quality lists can be used to generate meaningful facets. These facets are generated based on the user's interest.

References

[1] Zhicheng Dou, Zhengbao Jiang, Sha Hu, Ji-Rong Wen, and Ruihua Song. Automatically mining facets for queries from their search results. IEEE Transactions on Knowledge and Data Engineering, 28(2):385– 397, 2016.

[2] Ahmad Pouramini and Shahram Nasiri. Web data extraction using textual anchors. In 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), pages 1124–1129. IEEE, 2015.

[3] Liming Luo,Wen Lu, BingWei, Ye Qin, and YeQing Xiong. Application of internet technology and web information extraction wrapper based on dom for agricultural data acquisition. In Network and Information Systems for Computers (ICNISC), 2015 International Conference on, pages 327–331. IEEE, 2015.

[4] Manuela Rauch, Werner Klieber, Ralph Wozelka, Santokh Singh, and Vedran Sabol. Knowminer search-a multi-visualisation collaborative approach to search result analysis. In 2015 19th International Conference on Information Visualisation, pages 379–385. IEEE, 2015.

[5] Jan Friedrich, Christoph Lindemann, and Michael Petrifke. Utilizing query facets for search result navigation. In 2015 26th International Workshop on Database and Expert Systems Applications (DEXA), pages 271–275. IEEE, 2015.

[6] Giovanni Simonini and Song Zhu. Big data exploration with faceted browsing. In High Performance Computing & Simulation (HPCS), 2015 International Conference on, pages 541–544. IEEE, 2015.A. Alpher, , J. P. N. Fotheringham-Smythe, and G. Gamow. Can a machine frobnicate? Journal of Foo, 14(1):234–778, 2004.