Extraction of Top k Data from Web Pages

Darshana Dabhi[1] Ms. Jasmin Jha[2]
[1]M.E Student [2]Assistant Professor
[1,2]Department of Computer Engineering
[1,2]L. J. Institute of Engineering & Technology, Ahmedabad, Gujarat, India

Abstract— The web contains data in huge amounts. This data is a large source of information. All this information is in the form of structured or unstructured data. List is a crucial source of structured data on the web. Ranking the list data is generously important for information retrieval. Tremendous efforts have been done for extracting information from the structured data, especially from web tables, which contain quality information. Instead of focusing on context- free structured data, we aim to focus on context that we can spot, and then using the context to render less controlled information and proceed to its extraction. Here we highlight expensive as well as, rich source of information on the web, those are top-k web pages. Top-k web pages contain rich and quality information. They aim to identity the top attribute values for the entities of interest. Extraction of such lists can help answering engines to generate different fact and can act as a pre-processing step.

Key words: Rank Search, Time Sensitive Queries, binning

I. INTRODUCTION

Top k data on web is large and rich. The scale of this data is much larger than any manually or automatically extracted lists before. The top-k data is also rich in terms of the content acquired for each item in the list Top-k data is of high quality. It is generally cleaner than other forms of data on the web. Most data on the web is in free text, which is hard to interpret. Top- k pages have a common style. Top-k data is ranked. Ranking is extremely important in information retrieval.[4] Knowing that a term ranks 1st or among the top 3 based on a certain criterion is extremely useful in search, advertisement, and general purpose Q/A systems. Top-k data has interesting semantics. One of the reasons why top-k data is valuable is because each list has a context we can interpret, and the context is usually very interesting. Top-k lists are usually made up by domain experts for the general public, because people find such information interesting and useful.

Earlier the work performed in this field included extraction of only structured tabular data from the web pages in 2008. Thereafter the research continued for mining contigious and non contigious records, but had less accuracy. Further a hybrid approach was developed to mine general lists from web by studying the similarity of the elements using CSS box model. The approach developed after this was applicable to only html pages, and includes only table and list elements for extraction.

To overcome the above issue we focus on extracting the data from tags too. Instead of ignoring the web pages whose title match with the query, but does not contain the information in table or list tags, we can utilize such pages to extract data from tags. Usually most of the information in web is given in tags. The tags having the necessary information may have similar class applied to them. This can help us identify the tags of our interest.

This paper comprises of multiple sections: next section introduces the concept of top-k lists with proper examples, then we discuss the problem definition of the existing system and then propose a new method for overcoming it.

II. TOP K DATA

Top-k list is a list which contains k number of ranked elements. Where, k is any integer value. As compared to web tables top-k list contain rich and high quality data. Below are some examples of top-k titles:

1) Top 10 Mobile Phones in India
2) 12 Most Popular Books in USA
3) 5 Most Interesting Hollywood Movies
4) Top 15 Banks of 2014

Every top-k page title page contains at least these three pieces of information:

1) K: for example, 10, 12, 5, 15 in above example, which tells how many items are in top-k page.
2) Concept or topic: it tells which kind of item is retrieved. For example, mobile phones, books, Hollywood Movies, Banks etc.
3) Criteria for ranking: it decides on which basis ranking is provided to provide.

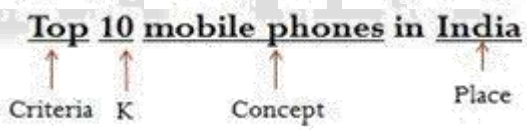

Fig. 2.1: Structure of Top k Title[4]

This fig is an example of top-k title. Top-k title can have many segments. The above example is showing a title with only two segments, first segment is main the segment and second segment may contain other modifiers. Segment 1 contains criteria- top, k-10 and concept-mobile phones. Second section gives information about place i.e. India. In many top-k title additional information about place or time is provide.[4] Compared to other structured data, top-k lists are cleaner, easier to understand and more interesting for human consumption, and therefore are an important source for data mining and knowledge discovery.

III. IMPORTANCE OF TOP K DATA

The basic reasons for extracting the top-k lists for the extraction of information are:

1) Top k data are large and rich on the web.[4] It is rich in terms of content required for each item in the list.
2) It is of high quality and usually cleaner than the other forms of data that are present on the web.[4] Mostly the data on the web is hard to interpret because of its free text format. Top k lists have a common style: Page title contains a number and concept of the items in the list.

Items can be seen as instances of the title, and number of items in the list should be equal to the number in the title.

3) Top k data is ranked.[4] Ranking holds a huge place of importance in the information retrieval. Knowing the rank of a certain item in a top-k list is extremely useful in search, advertisement, and general purpose Q/A systems.
4) Top k data has very interesting semantics. People find such information interesting and useful. Hence, information of this sort is more likely to find great audience.

The basic concept or topic of the web pages is extracted from the lists of instances.

IV. RELATED WORK

Paper [1] takes up machine learning approach to address the problem of automatically extracting titles from HTML documents (web pages). It has 2 phases: Training and Extraction.

The input is a document for pre-processing. It parses the body of the HTML document and constructs a DOM tree. And then it extracts all the leaf nodes as units from the DOM tree [10].

In learning, the input is a sequence of units from one document, and each sequence corresponds to one document. Labelled units are taken as training data and a model is constructed for identifying whether a unit is a title. In extraction, the input is a sequence of units from one document. The model is used to identify each unit in the sequence to find whether it is a title and it assigns a score to each unit.

The output is the extracted titles of the document. The consecutive units with the highest scores are chosen first for title and then consecutive units with second highest scores as second titles[1]. This paper tackles the issue of extracting the titles from the bodies of HTML documents. The main drawback being that, it only uses HTML pages and uses the titles extracted from the bodies of the HTML pages.

Paper [2] aims to return the top-k values of the attribute for the entity according to a scoring function for extracted attribute values. This scoring function depends on extraction confidence and importance. More often each document is accessed by users when searching for information related to an entity, the more likely it contains important information[2].By analysing query click-through data, search engines can identify the web documents that people refer to for information. For each entity in dataset, a frequency measure is computed on the basis of how many users have searched for the entity and how many pages matching a particular pattern have been clicked as a result of the search [2].

It follows the following algorithm:

- Document Selection: Select a batch of unprocessed documents
- Extraction : Process each document in batch with extraction system
- Top-k Calculation : Update rank of extracted attribute values for each entity
- Stopping Condition : If top-k values for each entity have been identified, stop, otherwise go to step 1.[2]

This paper addresses both quality and efficiency challenges and gives more popular documents in results by focusing on the importance of data. But this method may ignore the new and fresh web pages, which may be containing important data. Popular data may get more and more popular and new web pages will take some time to come into the result set.

Paper [3] introduces a Hybrid approach for automatic list discovery and extraction on the web (HyLiEn). HyLiEn uses the CSS2 visual box model to segment a web page into a number of boxes, each having a position and size. It recursively considers inner boxes and then extracts list boxes which are visually aligned and structurally similar to other boxes. Visual clues in the web page are utilised to generate candidate lists, which are subsequently pruned with a test for structural similarity in the DOM tree [3].

The paper considers the visual and structural features of the web lists and produces a general list, but not a ranked one. This method is applicable to only those web pages where CSS box model can be applied and does not have a notion of element distance that could be used to separate aligned but separated lists.

Paper [4] uses tag paths, which is a path from the root to the arbitrary node in the DOM tree. It improves the result quality by optimization heuristics:

- Visual Area: The total visual area of the candidate list versus the total area of the page is considered, by calculating the combination of image sizes, font sizes and potential white spaces
- Interleaving Lists: Top-k lists may have list items with alternate visual styles such as background colours or fonts. A special heuristic is used to detect such interleaving patterns and reconstruct the whole list
- K+1 problem: Top-k pages may have additional header or footer that looks almost same with same tag paths. In another case, the first or the last item of a top-k list may have slightly different style and gets excluded from the class. Special attention is paid to such items by analysing text content[4]

The paper has a basic algorithm running in four steps:

- Compute the tag path for every node in the DOM tree of the input page
- Group nodes with identical tag paths into one class & select those classes having exactly k items as Candidate classes
- Merge the candidate classes on whom the grow-up operation can be applied, item components that
- belong to the same list item are grouped together
- The candidate list is ranked by their importance to the page and returned as result[4]

This paper gives improved result quality and better performance by using the optimization heuristics, and also gives ranked results. As the focus of the paper is on the visual area and patterns, smaller lists may not get noticed.

The system introduced here consists of the following components:

1) Title Classifiers : It attempts to recognize the page title of the web page
2) Candidate Picker: It extracts all the candidate lists from the input page. It is structurally a list of HTML tag paths which are identical. A tag path is a sequence of tag names, from the root node to a certain tag node.
3) Top-k Ranker: It scores the candidate list and picks the best one by scoring function which is weighted sum of two features: P-score and V score.
4) P score measure the correlation between the list and title. V score calculates the visual area occupied by a list, because usually the main list of a web page tends to occupy larger area than other lists.
5) Content Processor : Processes the extracted list to produce attribute value pairs by inferring the structure of text nodes, conceptualizing the list attributes, using the tables heads or the attribute/value pairs.[5]

This method gives performance by providing domain-specific lists and focussing more on the content. It doesn't focus only on the visual area of the lists.

But a list, if divided into more than one pages, may not get included completely.

V. METHODOLOGY

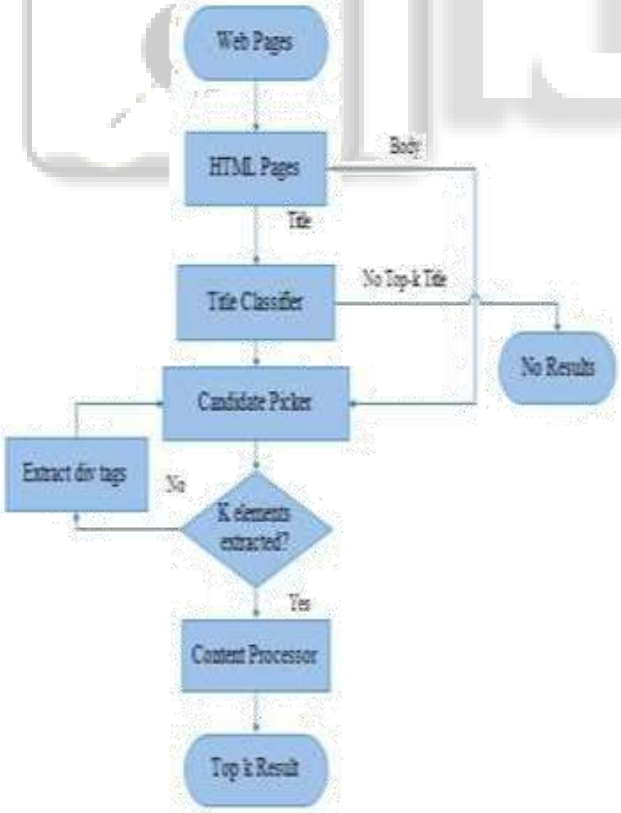

Fig. 2.2: Structure of Top k Title

The proposed system aims to extract the data from the lists as well as tables, of the web pages and the div tags too.

The basic flow of data will be as shown in the above diagram.

A. Title Classifier:

It aims to identify the title of the web page and helps us determine whether the page is useful for us or not.

1) Top k titles can be these types:

- Is a title contains the word "top" followed by a number, it is probably a top k title
- A title that satisfies the above rule is still a top k title if the word "top" is removed
- If a title contains a number followed by a superlative adjective, it can be a top k title, e.g. 5 costliest phones of the world
- If the title contains a number followed by a superlative adverb, followed by an adjective, it is likely to be a top k title, e.g. 10 most tallest buildings of India

B. Candidate Picker:

It extracts one or more list or tabular structures which appear to be top-k lists from the web page. The top k candidate items are structurally presented as a list of HTML nodes with similar or identical tag paths. A tag path is a path from root node to the certain tag node. It can be represented as a sequence of tag names, each followed by other till certain tag node is reached.

Tag path clustering method will compute the tag paths for each node and then cluster the similar or identical tag paths into one cluster. Following example can be taken for understanding tag path clustering:

Tag Path Clustering Algorithm in the candidate picker has following benefits:

1) Uses path similarities between data nodes
2) Can process a path query by accessing only small number of clusters & need not use all clusters
3) Enables path query to be processed efficiently by neglecting unnecessary data
4) Reduces the processing required for query processing

It is identified whether k number of items are extracted from the web page or not, if k <tr> or <li> tags are found, then the corresponding result is displayed otherwise, tags are further examined to find the results through tags.

C. Content Processor:

The content processor get the list from the candidate lists and tables, it and needs to produce the properly structured results in the form of attribute-value pairs.

D. Search Page

This is search page where users can fire a query and get the urls and its top k result. The result is shown in tabular format.

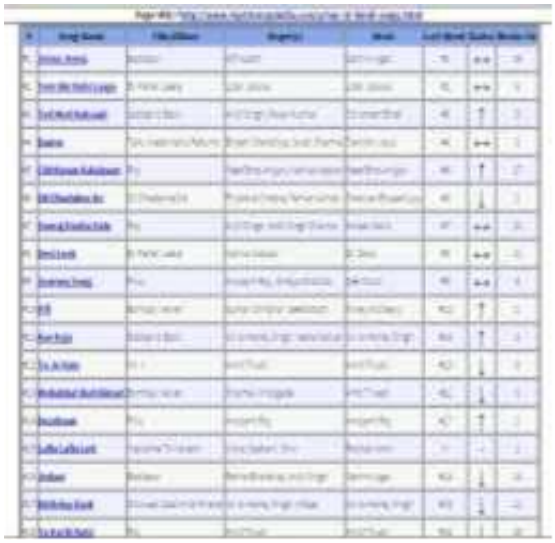

Fig. 4.5: Screenshot of Search Page Displaying Table

E. Result Analysis

This graph shows the comparison between existing systems and the proposed system, for the number of queries and the total number of URLs extracted for each query.

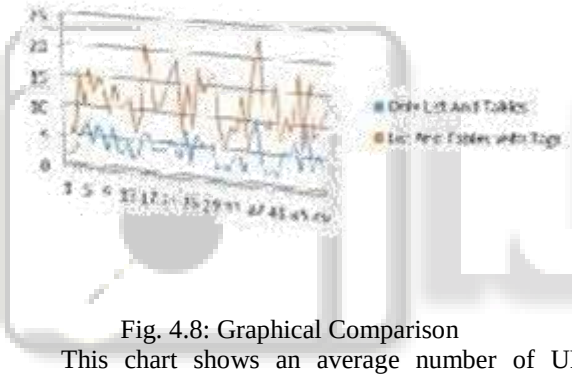

Fig. 4.8: Graphical Comparison

This chart shows an average number of URLs extracted for lists and tables, and another for lists, tables with tags extracted.

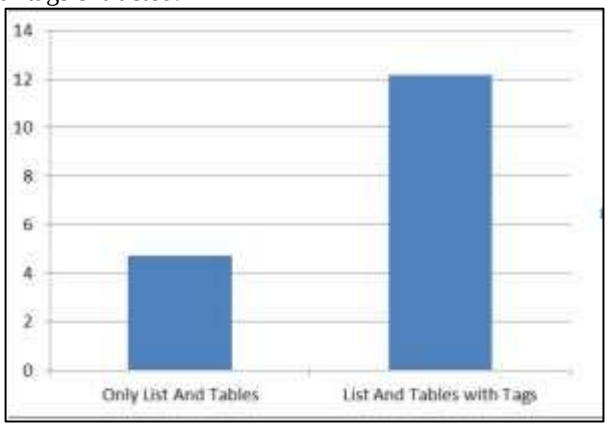

Fig. 4.9: Chart Showing Average URLs Extracted

VI. CONCLUSION

By, understanding the issues faced by the current systems, more improvements can be done in the field of extracting top-k lists from the web pages. The recent system only include the list structured data. We include the lists from the web pages and aim to extract the information from tags too, so that other web pages having the data does not get ignored. This helps to generate the ranked presentation of the results.

This work can further be extended to improve the extraction of top-k lists by overcoming the problems of the current system. New parameters can be added to identify and extract the content from the pages where the content may be given in other forms..

REFERENCES

[1] Matthew Solomon, Cong Yu, Luis Gravano, "Popularity Guided Top-k Extraction of Entity Attributes", Columbia University, ACM, 2010, ISBN: 978-1-4503-0186-2.

[2] Fabio Fumarola, Tim Weninger, Rick Barber, "Extracting General Lists from Web Documents: A Hybrid Approach", Computer Science Department, University of Illinois at Urbana-Champaign, Springer, Part -1, LNAI 6703, 2011, pp. 285-294.

[3] Zhixian Zhang, Zheng Wang, Haixun Wang, Kenny Q. Zhu, "Automatic Extraction of Top-k Lists from the Web", Shanghai Jiao Tong University, SJTU, 2012.

[4] Zhixian Zhang, Kenny Q. Zhu, Haixun Wang, Hongsong Li, "Automatic Extraction of Top-k Lists From the Web", IEEE, 2013, ISSN: 1063-6382.

[5] Gengxin Miao, Junichi Tatemura, Wang-Pin Hsiung, Arsany Sawires, Louise E. Moser, "Extracting Data Records from the Web Using Tag Path Clustering" USCB, ACM, April 20-24, 2009, ISBN: 978-1-60558-487-4.

[6] Donthi Raju, B. Rajani, T. Sahana Edwin, "Exclusion of Data Records From Documents of Web", IJRRECS, Sept 2014, Vol-2, Issue-9, 3252-3256, ISSN: 2321-5461.

[7] Yunhua Hu, Guomao Xin, Ruihua Song, Guoping Hu, Shuming Shi, Yunbo Cao, Hang Li, "Title Extraction from Bodies of HTML Documents and its Application to Web Page Retrieval", Microsoft Research Asia, ACM, August 15-19, 2005, ISBN: 1-59593-304-5.

[8] Mahesh Dabade, Shriniwas Gadage, "Methodology for Extraction of Information from Web Pages by Using Clustering Algorithm", IJSR, 2014, ISSN: 2319-7064.

[9] Lazaros Iliadis, Ilias Maglogiannis, Harris Papadopoulos, "Artificial Intelligence Applications and Innovations", Part II, Springer, Corfu, Greece, Sept 2011, pg- 241, ISBN: 978-1-4419-0221-4.

[10] Mario A. Nascimento, M. Tamer Ozsu, Donald Kossmann, Renee J. Miller, Jose A. Blakeley, Berni Schiefer, "Proceedings 30th International Conference on Very Large Data Bases", Toronto, Canada, Aug 31-Sept 3, 2004, pg- 430, ISBN: 0120884690.

