CUSTOM SEARCH ENGINE WITH EXTRACTION TECHNIQUE FOR E-COMMERCE WEBSITES

Dr. Chitrakala S[1], Lakshmi Narayanan R[2], Subramanian C[3], Abika Packia Chitra S[4]

[1,2,3,4] *Computer Science and Engineering, College of Engineering, Guindy, (India)*

ABSTRACT

Search results are being excessively used for many applications these days. Search results are obtained as a result of a query on a database. When a user wants to search the database for a particular keyword, he gives the keyword as input based on which a query is generated. The query on the database returns the search results in the form of a web page. In most applications there is a need to extract the records in the search result page. In this paper, we present a search and extraction technique which gets a keyword as input and returns the search result records from the resultant webpage. This paper only considers e-Commerce websites like Flipkart, eBay and Amazon. However it can also be scaled to other such websites. Our experiment indicates that the proposed approach is highly effective and accurate.

Keywords: Extraction, Result Records, Search, Search Result, Webpage.

I. INTRODUCTION

A search engine is one which searches webpages for certain keywords and returns a set of results. A custom search engine searches only a particular set of webpages. In our paper, the custom search engine searches only Flipkart, eBay and Amazon. Given a keyword and a website to search, the search engine must return the search records alone trimming out the unwanted contents like advertisements, suggestions etc. A typical result page consists of several search result records. A search result record is a unit of text corresponding to the keyword given as input. A search result record may contain any number of data units. A data unit is a piece of text that represents a concept in the search result record. For eg, If the user gives a keyword "Sony Laptop" and a website "Amazon", the result page consists of various search result records like the one showed in Fig.4.These records have to be extracted out as individual units for further processing.

In most search and information retrieval papers, search result records have been extracted with the help of tools like ViNT. These tools get a webpage i.e HTML file as input and return the search result records as output. However these tools do not prove to be accurate for result pages from Flipkart, eBay and Amazon. In order to facilitate the extraction of records for these websites this paper was implemented. The increase in the use of search results for applications has been the major reason behind the development of this paper. Search results are being used increasingly and there is a need for a tool that can extract result records out of webpages.

The paper has the following contributions:

1. The paper has an interactive interface that accepts input from the user and searches Flipkart, eBay or Amazon databases and returns the accurate search result page.
2. Unlike other papers, we have a visually appealing and user friendly interface design.
3. While most existing approaches use a tool for extraction, we extract search result records using DOM parser .All the records present in the webpage are extracted irrespective of the count.
4. We also find the tag path for each search result record and data units within a search result record. The tag path can be used for many applications like clustering, data alignment etc.

II. RELATED WORK

Search result extraction has been widely discussed in many papers. DeLa [3] splits the data region into more subparts and they consider only the data region with largest subparts and discard the others and it entirely depends on tag structure. It calculates the similarity before the data records are aligned in a manner, this leads to irregularity in making optional attributes. Moreover, this has good precision value when wrapping the records. Viper [2] (Visual Perception-based Extraction of Records) is one of the methods which is able to extract and separate relevance of different repetitive information contents or patterns with respect to the user's visual perception along with tags that are embedded with corresponding web page. It uses both human visual data perception value and the HTML tag structure to find rank and weight the patterns. Although Viper offers good results for single page, it does not handle when pages with nested structured data. CTVS handles nested structured data more efficiently than the Viper.

ViNT [4] is another tool like Viper for extracting the data, it uses visual as well as non-visual features to rank and weight the relevance of different extraction rules but has some drawbacks. First, it depends only on major data regions where the data records are highly reported than the other pages. Second, web users needed to collect the training pages or labels from the websites. Third, it needs continuous navigating for periodically updating dynamic changes. In contrast, CTVS [1] (Combining Tag and Value Similarity for data extraction and alignment) requires neither training page nor a pre-learned wrapper.

This paper is part of the project "Search result annotation for e-Commerce databases". The search and extraction approach has been used to get the result records based on a keyword and the records will be annotated with labels corresponding to them.

III. PROPOSED SYSTEM

The proposed system consists of two parts:

1. Keyword-based search
2. Record extraction.

The architecture diagram in Fig. 1 shows the process involved in searching and extracting the records.

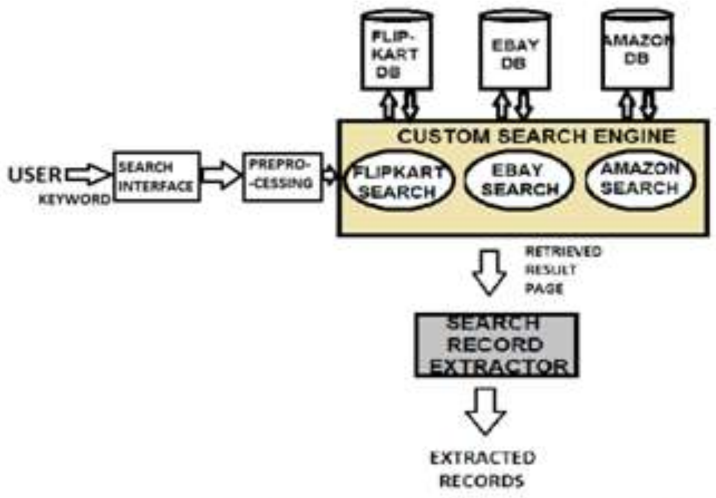


Fig.1 Architecture of Proposed System

3.1 Keyword-Based Search

Given a keyword and the e-Commerce site to search, the custom search engine must return the corresponding webpage. For eg, If the user searches for Sony headphones in Flipkart the corresponding result page will contain records about sony headphones. This functionality is provided by our custom search engine which can be used to search Flipkart, eBay and Amazon. However this can be extended to support other e-Commerce sites as well.

Keyword-based search consists of three modules:

1. Search interface
2. Preprocessing
3. Custom Search Engine

3.1.1 Search Interface

We get the keyword and e-Commerce site from the user via a search interface. The search interface as shown in Fig. 2, has been designed so that it is user-friendly and has a good user interface design. It consists of a form with two fields for entering the keyword and website. The user enters a keyword and a website. This information is then passed to the preprocessing step.

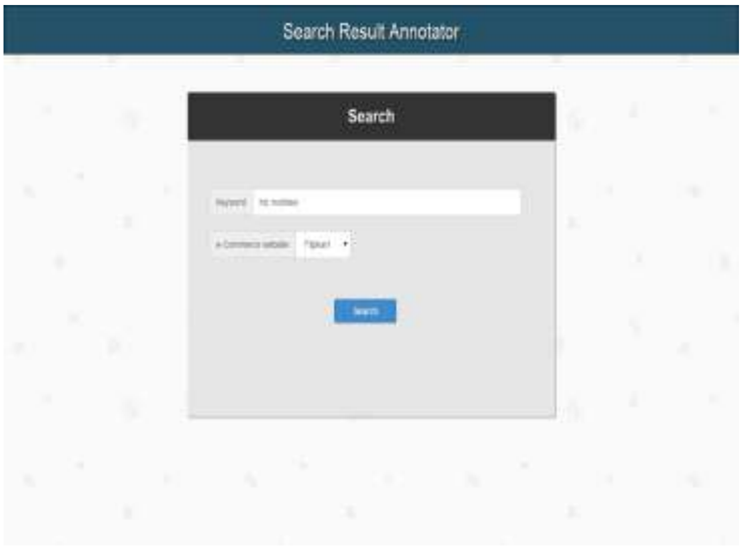
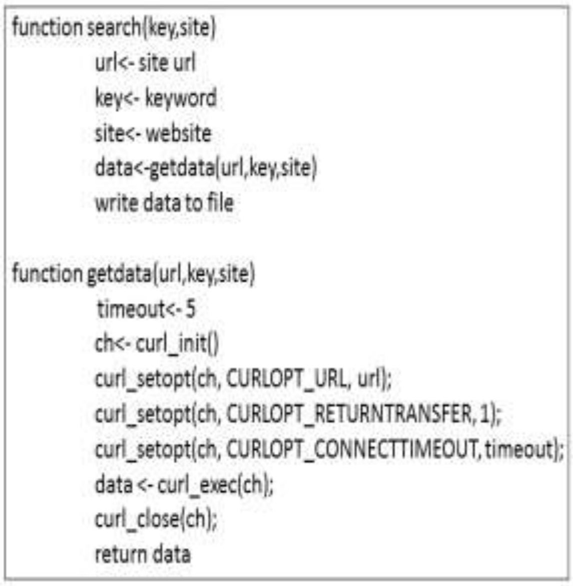


Fig.2 Search Interface

3.1.2 Preprocessing

The search interface passes on the input given by the user for preprocessing. In a URL, the space character is not allowed, instead of the space character the symbol '%20' should be used. In this preprocessing step, we replace the space character in the keyword with '%20'. To perform the search on the keyword corresponding URL must be created. Hence, the modified keyword is appended to the URL based on the website.



```
function search(key,site)
        url<- site url
        key<- keyword
        site<- website
        data<-getdata(url,key,site)
        write data to file

function getdata(url,key,site)
        timeout<- 5
        ch<- curl_init()
        curl_setopt(ch, CURLOPT_URL, url);
        curl_setopt(ch, CURLOPT_RETURNTRANSFER, 1);
        curl_setopt(ch, CURLOPT_CONNECTTIMEOUT, timeout);
        data <- curl_exec(ch);
        curl_close(ch);
        return data
```

Fig. 3 Algorithm for Keyword-Based Search

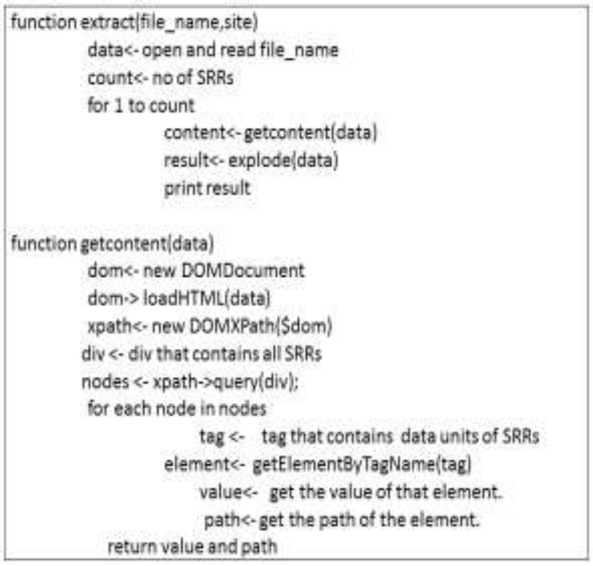
3.1.3 Custom Search Engine

Now this modified URL is given to the curl command. The curl command is used to retrieve the contents of the webpage specified by the URL. This retrieved content is stored in a HTML file with the name in the format "Keyword-website.html" for offline access and for extracting the search result records from the resultant webpage. A sample search result page is shown in Fig. 4.

Fig.4 HTML Result Page For The Keyword

3.2 Record Extraction

Extraction of records from web pages is done using the extraction algorithm specified in Fig. 5.The HTML file that has the retrieved content is parsed by using a DOM parser. The DOM parser is used to extract particular nodes or divisions in a HTML or XML file. The e-Commerce websites Flipkart, eBay and Amazon are analysed to study the pattern of the HTML template used in the result page. It is found that all result pages from a particular website say Flipkart follows the same template to display its contents. Only the data in it is dynamic. This template is deeply analysed so that we get a good knowledge of which divisions to omit and which to extract. Based on this analysis, we get a set of divisions whose contents need to be extracted so that they form the search result records. Once we identify the set of divisions to extract, we use DOM parser to get the content of the division and also the contents of the sub-divisions within that division.

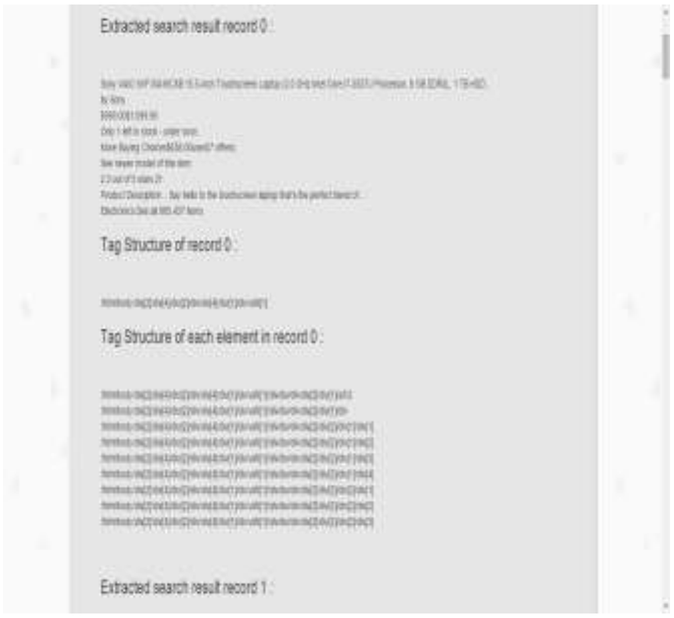


```
function extract(file_name,site)
        data<- open and read file_name
        count<- no of SRRs
        for 1 to count
                content<- getcontent(data)
                result<- explode(data)
                print result

function getcontent(data)
        dom<- new DOMDocument
        dom-> loadHTML(data)
        xpath<- new DOMXPath($dom)
       div <- div that contains all SRRs
       nodes <- xpath->query(div);
        for each node in nodes
                    tag <-   tag that contains  data units of SRRs
                element<-  getElementByTagName(tag)
                    value<-   get the value of that element.
                     path<- get the path of the element.
          return value and path
```

Fig. 5 Algorithm for Search Record Extraction

We also get the tag path of each division so that it can be used for other applications like data alignment. Thus we get a list of search result records along with the tag path. These results are printed out as in Fig. 6



```
Extracted search result record 0 :

[illegible]

Tag Structure of record 0 :

[illegible]

Tag Structure of each element in record 0 :

[illegible]

Extracted search result record 1 :
```

Fig. 6 Extracted Search Result Records and Their Tag Structures

IV. EXPERIMENTS

4.1 Data Sets and Performance Measure

Our experiments involve data sets in the following domain: Electronics, Book, Movie and music. For each domain, keywords from three e-Commerce websites Flipkart, eBay and Amazon are selected for performance evaluation. Datasets DS1, DS2, DS3 consist of keywords that will result in webpages from Amazon, Flipkart and eBay in a particular domain. We adopt the precision and recall method to evaluate the performance of our approach.

For search, the precision is defined as the percentage of the correctly returned result pages over all the result pages returned by the system; recall is the percentage of the webpages that are correctly returned by the system over all manually found resultant webpages by the expert. For extraction, the precision is defined as the percentage of the correctly extracted units over all the units returned by the system; recall is the percentage of the search result records that are correctly extracted by the system over all manually extracted result records by the expert.

- Precision:

$$\frac{d1}{d2} * 100$$

- Recall:

$$\frac{d3}{d4} * 100$$

Fig. 7 Formulas for Calculating Precision and Recall

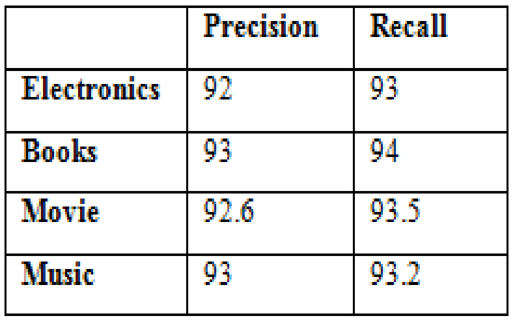
In Fig.7, d1 represents the number of correctly extracted units, d2 represents the total number of units returned by the system, d3 represents the number of search result records that are correctly extracted by the system and d4 represents the number of records manually extracted by the expert.

4.2 Experimental Results

For search, the precision and recall for all the datasets DS1, DS2, DS3 over all the domains: Electronics, Books, Movie, Music is 100% i.e the system is fully accurate when producing search results based on keywords. For extraction, precision and recall values for DS1 for all the domains are around 95%. For DS2, the precision value on an average is 93% and the recall value is around 93.5% on an average. For DS3, the precision values are around 94.5% and the recall values on an average are around 95%.The corresponding precision and recall values for the domains : Electronics, Books, Movie and Music for the datasets DS1 , DS2 and DS3 have been tabulated in Fig.8,9 and 10.

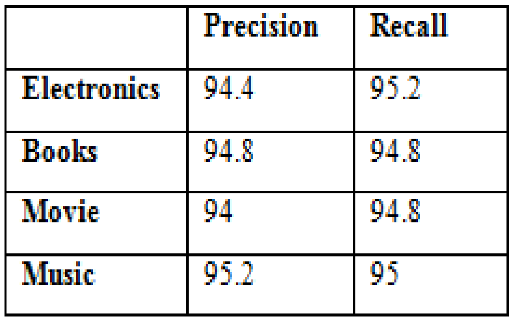
	Precision	**Recall**
Electronics	95.2	96
Books	94.8	95.6
Movie	95	95.2
Music	95.6	96

Fig. 8 Precision and recall values for DS1



	Precision	Recall
Electronics	92	93
Books	93	94
Movie	92.6	93.5
Music	93	93.2

Fig. 9 Precision and Recall Values for DS2



	Precision	Recall
Electronics	94.4	95.2
Books	94.8	94.8
Movie	94	94.8
Music	95.2	95

Fig. 10 Precision and Recall Values for DS3

V. CONCLUSION

In this paper, we developed a search and extraction technique that is used to extract search result records from web pages. We also developed a search interface that accepts a keyword and a website from the user. The search results obtained are very accurate. The search result page obtained is given to the extraction module. From the webpage, unwanted information like advertisements and filters are removed and the remaining search result records are printed. The tag path of the search results records are also extracted so that they can be used for data alignment etc. Our experimental results show that our search technique is 100% accurate over all the domains considered. Our extraction technique is 95% accurate for search result pages from Amazon and it is 93% accurate for result pages from Flipkart and 95% accurate for result pages from eBay. The accuracy values are calculated using precision and recall values. There is scope for improvement in extraction technique for certain domains .Further, the extracted records can be aligned and labels can be assigned to them using annotators.

REFERENCES

[1] Weifeng Su, Jiyang wang, Frederick H.Lochovsky, “Combining Tag and Value Similarity for Data Extraction and Alignment”, IEEE Transactions on Knowledge and Data Engineering, vol.24, No.7, July 2012

[2] K. Simon and G. Lausen, “ViPER: Augmenting Automatic Information Extraction with Visual Perceptions”, Proc. 14th ACM Int’l Conf .Information and Knowledge Management, pp. 381-388, 2005.

[3] J. Wang and F.H. Lochovsky, “Data Extraction and Label Assignment for Web Databases”, Proc. 12th World Wide Web Conf.,pp. 187-196, 2003

[4] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu, “Fully Automatic Wrapper Generation for Search Engines”, Proc. 14th World Wide Web Conf., pp. 66-75, 2005.