International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:5, No:2, 2011

Deep iCrawl: An Intelligent Vision-Based Deep Web Crawler

[1]R.Anita, [2]V.Ganga Bharani, [3]N.Nityanandam, [4]Pradeep Kumar Sahoo

Abstract**—**The explosive growth of World Wide Web has posed a challenging problem in extracting relevant data. Traditional web crawlers focus only on the surface web while the deep web keeps expanding behind the scene. Deep web pages are created dynamically as a result of queries posed to specific web databases. The structure of the deep web pages makes it impossible for traditional web crawlers to access deep web contents. This paper, Deep iCrawl, gives a novel and vision-based approach for extracting data from the deep web. Deep iCrawl splits the process into two phases. The first phase includes Query analysis and Query translation and the second covers vision-based extraction of data from the dynamically created deep web pages. There are several established approaches for the extraction of deep web pages but the proposed method aims at overcoming the inherent limitations of the former. This paper also aims at comparing the data items and presenting them in the required order.

Keywords**—**Crawler, Deep web, Web Database

I. Introduction

THE web database keeps expanding every day, which drives the focus on researches towards deep web mining. The information in a web database can be fetched only through its web query interface. These Web databases are queried for particular information and the query result is enwrapped to form a dynamic web page called the deep web page. It is almost impossible for the search engines to retrieve this information and hence this is called deep web or hidden web. The result objects obtained from the query submitted, is displayed in the form of data records. For example, the Wavetel site (a mobile sales company) has its own personal database for which it has a search interface on its webpage.

R.Anita is with the Department of Computer Science Engineering, Sri Sai Ram Engineering College, Sai Leo Nagar, West Tambaram, Chennai - 600 044.Tamil Nadu. India (phone: 919789009934;
e-mail:anitaramanath@gmail.com).
V.Ganga Bharani is with the Department of Computer Science Engineering, Sri Sai Ram Engineering College, Sai Leo Nagar, West Tambaram, Chennai - 600 044.Tamil Nadu. India (phone: 919840692470;
e-mail:vgangabharani@gmail.com).
N. Nityanandam is with the Department of Computer Science Engineering, Sri Sai Ram Engineering College, Sai Leo Nagar, West Tambaram, Chennai - 600 044.Tamil Nadu. India (phone: 919840043240;
e-mail:nity5550@gmail.com).
Pradeep Kumar Sahoo is with the Department of Computer Science Engineering, Sri Sai Ram Engineering College, Sai Leo Nagar, West Tambaram, Chennai - 600 044.Tamil Nadu. India
(e-mail:pradeep313_mtech@rediffmail.com).

When the user submits a query in their search interface, a page is created dynamically which has a list of mobiles that matches the query. This dynamically created page is an example of deep web page. Each mobile detail is displayed in the form of structured data records; each data record contains data items like price, discount, features, color, etc.

Data records are structured not only for the ease of humans but also for many applications like deep web crawling were data items need to be extracted from the deep web page. Recently the deep web crawling has gained a lot of attention and many methods have already been proposed for data record extraction from deep web pages. But these proposed methods are structure-based; either based on analyzing HTML codes or the tag types of the web pages. The inherent limitations of these methods are:

1. They are dependent on the programming language of the web page. Most of these methods are meant for HTML. Even if we assume that only HTML is used to write all the web pages, the previously proposed methods are not fully efficient and fool proof. The evolution of HTML is non-stop and hence the addition of any new tag will require amendment in the previous works in order to adapt to the new version.
2. In reality, HTML is not the only known web page programming language. Many languages like xml, xhtml have evolved. So the previous works should either be amended to consider these new additions or be abandoned.
3. The existing works does not consider the complexities like embedded java scripts and VB scripts. The underlying structure is drastically different from their Web Browser layout. This makes it difficult to analyze the structural regularities and hence extraction becomes difficult.

The deep web page is designed in such a way that data records and data items are arranged with visual regularities for easy understandability. The web page shown in Fig1 aptly describes the kind of visual regularities in arranging the data records.

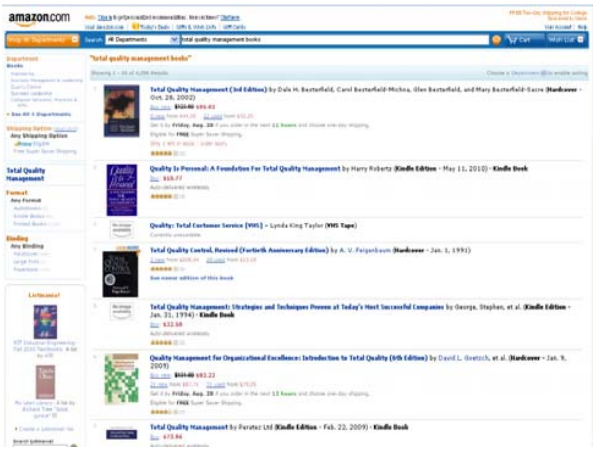

Fig. 1 Example of Deep Web Page Amazon.com

The data records and data items of this page are arranged in one particular order with visual demarcations between every data record. Similar data items in each data record have the same relative arrangement and font.

In this paper we exploit these visual similarities of the data items to make the extraction of data records efficient and generic (independent of webpage programming language). We propose Deep iCrawl, an intelligent deep web page crawler that is completely vision based.

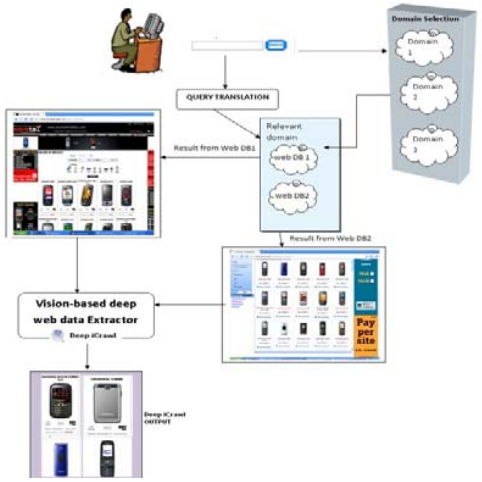

Fig. 2 An overview of Deep Web Crawling

Deep iCrawl involves 3 phases: Creating an integrated interface, query analysis and translation and extracting data from the deep web page.

For this process we need a data base which will have all domains grouped together. Since deep web pages cannot be indexed by normal crawlers, we need another method to access the web databases. We consider two methods for indexing deep web content. In the first method, we can create an integrated interface, with a mediator form and establish semantic mappings between individual web data sources and the mediator form. But it involves several drawbacks, like cost of building and maintaining the mediator forms and the mappings is high, identifying which queries are relevant to each domain is extremely challenging. And also there are thousands of web databases, which have to be connected to the mediator form. This task is hence tedious and difficult. In the second method, called Surfacing [7], we pre-compute the most relevant form submissions for all interesting HTML forms of the web databases from their respective web sites. The URLs resulting from these submissions are generated off-line and indexed like any other HTML page. This approach enables us to use the existing search engine infrastructure and seamlessly include the Deep-Web pages. Once the personal database containing these deep web result pages is generated, we can start extraction of the data records and data items. The 3 phases for extraction is as follows.

A. Creating an integrated interface:

In this phase the Deep Web Crawler identifies the websites that have Web database and clusters the [7] surfaced deep web pages based on their similarities and puts them into various databases. Each of the clustered domains in the databases can be queried using the integrated interface.

B. Query translation:

When the user submits a query in the integrated interface, the crawler analyses the query and selects the relevant domain from its list of domains (clusters of database).

C. *Extraction of results:*

These queries are then submitted to the databases and the relevant deep web pages are taken from the database. The deep web pages, i.e. the result of different websites, are analyzed and the data records are extracted using various rules. Then, these results, comprising of data records and data items, are merged and presented in an integrated result page.

Earlier methods like XWRAP [1], W4F [2], which are semi-automatic methods of wrapper generation, were used. Then to reduce the manual effort and to improve efficiency automatic approached like ROADRUNNER [3], MDR [4] were used for deep web data extraction. But all the previous methods are structure based and also they mainly concentrate on data record extraction only.

II. DEEP ICRAWL(DIC):

Deep iCrawl has two components, the Deep iCrawl data record extractor (DiCRE) and the Deep iCrawl data item extractor (DiCIE). Since DiC makes use of the visual features, it overcomes the limitations of existing works.

DiC employs four steps for data extraction from deep web page. First, it takes sample deep web page from a particular web database and obtains its visual representation. Later this is converted to a Visual Block Tree. Second, data records are extracted from visual block tree. Third, this extracted data records are further partitioned into data items and the similar data items (semantically related) are clustered together. Fourth, the most important step is the generation of visual wrappers, which will extract the data items and data records from the other deep web pages which are dynamically created by the same web database.

III. Visual Features

A webpage may contain text, Image, audio, video, etc. But this paper focuses mainly on Web page layout and fonts.

A. Web page layout:

Each and every point of a web page can be uniquely defined if it is marked under the coordinate system, where the x coordinate represents horizontal left-right and y coordinate represents vertical top-down. The x coordinate increases as we go down the web page and y increases as we go rightwards in the web page. So each and every object (text/image) can bonded within a minimum rectangle whose sides are parallel to the axes. The objects will have (x, y) coordinate, x is the horizontal distance and y is the vertical distance of the rectangular bound, that will describe it. The size and the coordinates of the objects make the web page layout.

B. Font:

The font attribute is analyzed in order to find the similarity between data items. Attributes of fonts are Size, Face, Color, Bold, Italic, Underlined, Hyper linked, etc. Only if the values of all these attributes are same the fonts are considered to be same.

IV. Visual Block Tree

A deep web page can be transformed into visual block tree and the visual features mentioned above can then be easily extracted. This paper makes use of the VIPS algorithm [5] to convert deep web page into visual block tree. The whole deep web page is divided into segments of rectangular regions and each of these regions corresponds to one block of the visual block tree. The web page itself is considered a block which makes the root of the visual block tree. Leaf blocks are those which cannot be further segmented. The leaf block is the basic unit of the web page which can be an image or continuous text.

Each internal block a is represented as a = (CS,P,S,FS,I) where CS is the set containing its child blocks (note that the order of blocks is also kept), P is the position of a (its coordinates on the Web page), S is its size (height and width), FS is the set of the fonts appearing in a, and IS is the number of images in a. Each leaf block b is represented b=(P,S,F,L,I,C), where the meanings of P and S are the same as those of an inner block, F is the font it uses, L denotes whether it is a hyperlink text, I denotes whether it is an image, and C is its content if it is a text. Fig3 shows the overall layout of a deep web page whose Visual Block Tree is shown in Fig4.

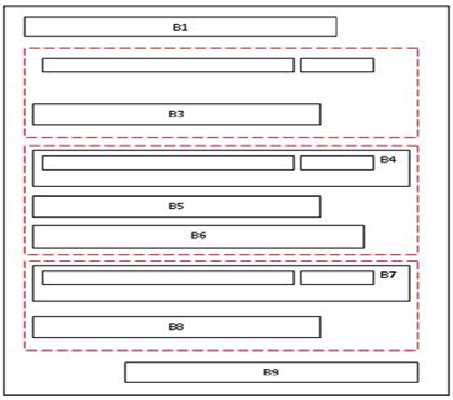

Fig. 3 A Layout of Deep Web Page

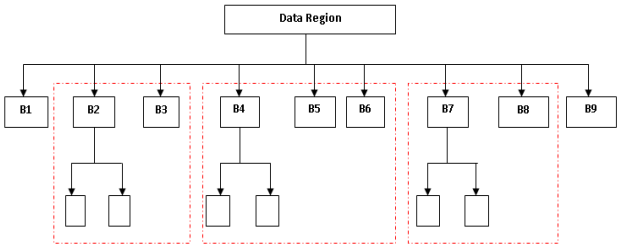

Fig. 4 Visual Block Tree

V. Vision-Based Rules

This paper employs certain vision-based rules to analyse the visual structure of a deep web page.

R1: The data in a webpage is always displayed with visual regularities.
R2: One particular data item in every data record is mandatory in a specific deep web page. This can be found by sampling the deep web pages of that particular website.
R3: Data regions in the deep web pages are always centered horizontally.
R4: The size of the data region is usually large relative to the area size of the whole page.
R5: The data records are usually aligned flush left in the data region and present one after another.

VI. Data Record Extraction

To extract a data record from a deep web page, first, the boundary of the data records must be discovered. The record extraction process must follow two rules: 1) for every data region that is considered, all the data records in it must be extracted 2) every valid data item in an extracted record must be included and not single incorrect data item should be included.

To extract data records from the deep web page, first the data region is located then the data records from the region are extracted. Data region is a rectangular region that includes data records of the web page. Data region corresponds to a block in the visual block tree. To identify the data region, rules R3 and R4 are employed. Rule R4 can be implemented

as a formula that is given by-[6] $(area_{block}/area_{page}) > T_{region}$, where T_{region} threshold which is trained from sample deep web pages. In case there is more than one block in the visual block tree that satisfies these two rules, the one with smallest area is chosen. The data regions in the visual block tree can be found efficiently and accurately by this method.

Data records are the child blocks of data regions, so it is enough that we concentrate only on the child blocks of data region. To accurately extract data records from a data region two facts must be considered. One, noise blocks and two, the number of blocks that makes one data region. Noise blocks are blocks that do not belong to any data record but they are contained in data region. Noise blocks can be statistical information (e.g. 1000 matching results for "user's search *keyword"*) or they can be annotation (e.g. 1, 2, 3, (prev), (next)). The second fact that must be considered is that the number of blocks that makes a data record is not fixed. Many blocks in the visual block tree might correspond to a single data record. For example, in Fig4 block B2 and B3 belong to same data record while B4, B5 and B6 correspond to another data record.

Once, the data region is chosen using the already mentioned rules, the boundary of all the data records in the region must be identified. Boundary discovery means identifying 'which blocks in the visual block tree corresponds to which data record'. This can be done in three steps.

Step 1: Filtering the Noise
Step 2: Appearance-based clustering
Step 3: Blocks regrouping

A. Filtering the Noise:

Noise blocks do not appear in between the data records, they appear either at the top or bottom of the data region (inference of rule R5). According to R5, data records are usually aligned flush left in the data region, so all the blocks that are not aligned flush left are considered as noise and are filtered. But this step does not ensure the filtering of all the noise blocks. For example, in Fig3 blocks B1 and B9 are noise blocks but in this step only B9 will be removed as B1 is aligned flush left.

B. Appearance-based clustering:

The rest of the blocks are considered useful blocks and are clustered based on their appearance. Items in data records can be primarily classified into two: text and image. Images of two data records can be considered similar if they are of the same size and text similarity is based on same font attributes. Text can be further divided into plain text and link text.

Algorithm for block clustering:
Input: B, List of blocks say $b_1,b_2...b_m$
Output: C, List of Clusters $C_1, C_{2...}\ C_n$
Begin

1. Take the first block and make a cluster, say C_1
2. Add C_1 to list of clusters C
3. For each block b_i in list B
   Compute $S(b_iC_i)$
   If ($S(b_iC_i) > T_a$ then
   //T_a *is threshold value trained by //sample pages generally set as 0.8*
   Add b_i to the cluster C_i
   Else
   From new cluster C_q based on b_i a
   Add C_q to the list C
4. End

$S(b_iC_i)$ is the average of Similarities between b_i and all the other blocks in cluster C_i. Applying this one pass clustering algorithm in the example shown in Fig.1, the images in data records are clustered together; the titles of all the data records are made into one cluster and so on.

C. Blocks Regrouping:

The clusters in the list C from previous step do not correspond to data record. Each cluster has a collection of blocks from different data records. The blocks in the same cluster are same type of contents from different data regions, for example titles of all data regions. Now blocks must be regrouped to form groups that will have blocks that belong to one data record only.

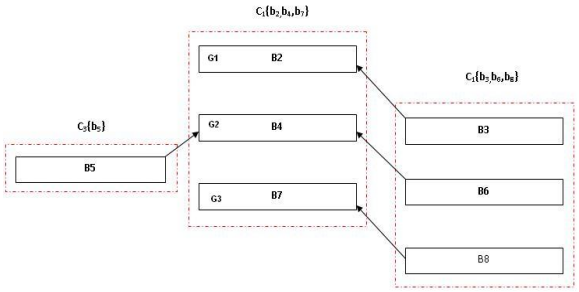

Fig. 5 Regrouping of blocks

R2 gives the basic idea of blocks regrouping. According to R2 there will be a mandatory data item in every data record. Therefore one of the clusters with maximum number of data items is considered to be mandatory data item cluster. This cluster is C_{max}. This C_{max} must be identified and the blocks of C_{max} become the seeds for forming Groups of blocks that belong to same data record. The data items of data records are presented in a fixed order. Fig5 is drawn with reference to the deep web page layout shown in Fig3. C_1{B2,B4,B7} contains blocks of different data records and are the first items of their corresponding record. So, all the blocks in other clusters are grouped by linking them with the blocks in this cluster. For example, B5, B6 are linked to B4 and they form group G2. For each cluster on the page, a minimum rectangle is formed. The order of the different semantic blocks can be found by comparing the position of these rectangles on the page. For example if the rectangle enclosing all the titles is higher than the rectangle enclosing the ID number block, then title must be ahead of its corresponding ID. This is done for all rectangles and hence the blocks are regrouped.

VII. Data Item Extraction

A data record consists of a group of data items. The data items mainly have three types of data:

1. Mandatory data items-i.e. author name, title etc. In a deep web page of books site.
2. Optional data items –i.e. discount, availability etc.
3. Static data items- annotations to the data.

These data items are extracted and are classified under columns, according to their semantics. This task of extracting the data items from the data records and segregating them is called the Data Item Extraction process. This process is performed in two phases:

A. Data record segmentation:

In the visual block tree, consider the leaf nodes, which are the segments of the data records, in a data region. The segments of the data records consist of composite data items, which cannot be further segmented, are initially considered as regular data items. These can be segmented using some heuristic rules. The output of this phase is that all the data records are converted to a sequence of data items.

B. Data item extraction:

The data items in data record can contain optional items, such as discount price for a book. So it becomes difficult to align them directly. Therefore the main aim of the data item alignment process is:

1. Put the data items of same semantics together.
2. Preserve the order of each item in the data record.

(i) *Data item matching:*

It is apparent that data items of similar semantics in different data records have the same font and position. The position of the data items can be classified into two types: absolute and relative. The absolute position refers to the distance between the left side of the data record and the data item respectively. For example, the title of the book will always have a fixed absolute position in the data records as shown in the figure. The based on the absolute positions of the data items they can be matched. There will be items that do not have an absolute position, like the author name, as shown in the figure1. These items can be matched based on their relative position, i.e. if the title of the record is matched then the author of that data record is matched. The algorithm is presented in figure2. In the given algorithm the $item_{p1}$ and $item_{p2}$ represents the data items just after $item_1$ and $item_2$, which compares the data items in relative position, like the author name, to be matched.

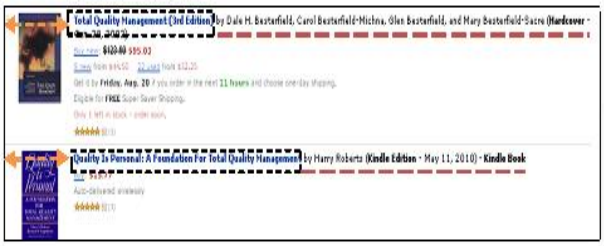

Fig. 6 Data Item Matching

// Algorithm for data item matching
Input: $item_1$, $item_2$
Output: data item matched or unmatched
Begin
1. if (font($item_1$)!= font($item_2$))
2. return unmatched;
3. if(position($item_1$)==position($item_2$))
4. return matched;
5. if($item_{p1}$ and $item_{p2}$ are matched)
6. return matched;
7. else
8. return unmatched;
End

(ii) Data item alignment:

All the data items in the data records are in a fixed order. Each data record is considered as a sequence of data items. At the end of data alignment process, all the data items must be aligned according to their semantics. Considering the possibility of having optional data item in the records, we must have dedicated attributes under which these data items can be classified. When data records without the optional data items are classified, this column must be filled with a predefined blank item, in order to maintain the same number of data items for each record at the end of classification. The data records are given as input, whose data items are represented as $\{item^1i, item^2i, \ldots, item^mi\}$. At every stage, the unaligned data items of a data record are processed and its position is decided. The aligned output of the data items can be stored in a data base for analysis and other web based applications.

VIII. WRAPPER GENERATION

Wrappers are generated for web databases by analyzing their sample web pages. The visual schema of all the deep web pages generated by a web database will be the same. Therefore, once the data records and data items from one of these deep web pages is extracted, an extraction wrapper is generated for that web database, using which new web pages can be crawled without repeating the extraction process. Deep iCrawl uses two phase extraction process, the Deep iCrawl data record extractor (DiCRE) and the Deep iCrawl data item extractor (DiCIE). The previous works like XWRAP [1] makes use of the tag trees for extraction. While here we concentrate on the visual features of the web page. Some basic information like the frequent symbols and their data types are used in this method.

A. Vision-Based Data Record Wrapper

The data record region in a deep web page is identified using the visual block tree. For every data region R in the deep web page P from site S, we find out five parameters, (x, y) coordinates of R on P, width *w and* height *h* of R and the level *l* of R in the visual block tree. When performing the extraction on the new deep web page P*, we select the data region such that it overlaps with R, that was found for the sample web page. The coordinates, height and width information of these selected regions can be used to identify the overlap area.

Once the data region is selected, the data records must be identified. For this process we have to find the first and the last block (b_{last}) on every record of a deep web page. While training the extractor with a sample page, we find out the visual information of the first block of every record in that result page. The distance *d* between every record is also noted. The wrapper finds the first block of every record in the new web pages, using this information. The b_{last} of every record follows the these rule,

1. The vertical distance between any two blocks of the same record will be less than *d.*
2. The distance between the last block (b_{last}) of a record and the start of the next record will always be greater than d, i.e. the records in a deep web page will be placed one after another with a separation between the last block of the previous record and the start of the next record.

Using these two rules, we can find the last block (b_{last}) of every record, as that block whose distance with the next block is greater than *d.* Thus the record blocks can be separated and extracted.

B. Vision-Based Data Item Wrapper

The work of a data item wrapper is to segregate the data items of the data record, according to their semantics. To perform this task, the data alignment algorithm is used classify the data items of the sample deep web page, according to the different attributes identified from the deep web page. The following parameters of the attribute can be used to segregate the data items:

1. *f,* font used in the data items of that attribute.
2. *l,* a Boolean variable, to denote if the data item of this attribute is a link.
3. *d,* image, text, number, date, email, etc.

These parameters can be used to classify the data items of the new web pages. The wrapper performs the following task:

1. The wrapper gets a sequence of attribute $\{a_1,a_2,\ldots,a_n\}$ from the sample web page.
2. A set of data items $\{item_1, item_2, \ldots, item_m\}$ from the new deep web result pages are extracted.
3. The data item is compared with these attributes and classified accordingly based on the parameters *(f, l, d)* of each attribute, i.e. $item_i$ *and* a_i are compared with each other for *(f, l, d)* and if they are the same, $item_i$ is classified under a_i.

Similarly all the data items of every data record are classified.

IX. Conclusion

Deep web has abundant information in it. To tap these resources, we need an efficient method to get the desired information which is embedded in the deep web pages. The structured data that is extracted can be used for processing in web based applications in real time. The paper effectively extracts the deep web data records and data items using visual features. In this paper we create a database of deep web pages of different domains, which will have to be updated frequently. This process of updating will require an effective algorithm to maintain the efficiency of the system. The future works can be done in integrating this feature in this proposed method.

References

[1] XWRAP l.Liu, C.Pu and W.Han, "XWRAP: An XML-Enable Wrapper Construction System for Web Information Sources"
[2] A.Sahugent and F.Azavant "Building Intelligent Web Applications using Lightweighted Wrappers"
[3] V.Crezcenzi, G.Mecca and P.Merialdo "RoadRunner: Towards Automatic Data Extraction from Large Websites", Proceedings of the 27th VLDB conference, 2003
[4] B.Liu,R.L.Grossman and Y.Zhai "Mining Data Record in Web Pages" *SIGKDD .03*, August 24-27, 2003, Washington, DC, USA
[5] D.Cai , S.Yu, J.Wen and W.Ma,"Extracting Content Structure for Web Pages Based on Visual Representation"
[6] Wei Liu and X. Meng "ViDE: A Vision-Based Approach for Deep Web Data Extraction" IEEE Transactions On Knowledge And Data Engineering, Vol. 22, No. 3, March 2010
[7] Jayant Madhavan, David Ko, Łucja Kot, Vignesh Ganapathy, Alex Rasmussen and Alon Halevy. "Google's DeepWeb Crawl". PVLDB '08, August 23-28, 2008, Auckland, New Zealand