A Hybrid Approach for Extracting Web Information

R. Abarna1* and S. Pradeepa2

1School of Computing, SASTRA University, Thanjavur - 613401, Tamil Nadu, India; abarna.raj14@gmail.com
2SASTRA University, Thanjavur - 613401, Tamil Nadu, India; pradeepa@cse.sastra.edu

Abstract
Mining the webpage is the predominant technique to grab the data from the internet. It is the extracting job from the web pages in either supervised or unsupervised. Unsupervised extraction extracts more irrelevant data than the relevant and it fails to eliminate the data redundancy. The proposed hybrid approach separating the relevant content from the webpages and filter out the replication. The newly generated hybrid algorithm performs the region separation using tag tree and straining the repeated information. Hence the output contains only reliable data. This approach is the proficient way for extracting the relevant information from the webpages.

Keywords: Hybrid Approach for Extraction, Multi String Alignment, Region Separation, Web Content Mining

1. Introduction
Web mining deals with several areas in the web documents and it’s a challenge- when information is huge/diverse/redundant1. The business persons concern about this and make money by extracting the web information and structuring in relational tables hence third party clients can use2. Web mining is chiefly to extract facts from web by filtering at least the banner, advertisements, header and footer, navigation bars and search boxes etc. Scraping the website for extracting its content and presentation in various formats such as Microsoft Excel (.XLS), Document (.doc), Text (.txt), XML, Microsoft Access (.mdb), SQL, etc. This could be used in the fields like Real Estate Properties, Scraping Product details, Scraping Product details like reviews and ratings from E-commerce website for evaluation, Scraping business related details from websites like Yelp and Mantra. These ideas will boost the advertising of your product. This paper focus more on transforming the unstructured data to structured one. The progress of web content mining is discussed in recent literatures3.

Information extractor is used for information extraction. It is of two types either based on rules (supervised) or heuristics (unsupervised). Supervised techniques based on extraction rule will have so many limitations like rigidness of the rule when the structure of the document changes, these changes may invalidate the existing rules. It also requires maintenance and needs frequent human interaction to guide the extraction with what type of information should be extracted and how much of data that need to be extracted. Researchers then move on to unsupervised approach of heuristics based extraction because it does not need to use extraction rules rather it will rely on heuristics and hypotheses that have proven in most of the techniques6. It does not need much human intervention. Web data extraction has been done in so many ways like generating extractors, wrapper generation, generating transducer, and some more based on document formats like parser induction, regular expressions etc5. For efficient extraction usage of parsers will be the right choice that is learned from our studies8.

In 1 presented an approach for mining the text related data by filtering the shared token sequence. JTidy was used to convert the HTML code to XHTML code1, whereas

*Author for correspondence
these XHTML has many features like sustainability and compatibility; but it has problem in working with the browser. Here Tree matching algorithms used a particular pattern tree as a sample and matches with others and finds the distance between nodes. The Input documents are treated as whole leads in difficulty. Earlier a technique was presented to learn the regular expression with the algorithm to find the shared pattern. Then created trinity tree with the prefix, separators, and suffix of the pattern\[10\]. Capturing of the groups presented in the tree would provide the extracted data. Here the user has to assign the meaning to capturing the groups. It will not be affected by any HTML errors. It does not require creating DOM trees and well-formed HTML documents. It lags in effectiveness. In \[10\] algorithm was presented that compare the input documents for learning the expression. Backtracking process consumes more time here\[10\]. In 2010, Kayed presents an approach to derive the common template from inputs, then by using clustering algorithms that generate distance in the tree to DOM nodes of the inputs. But the time consumption decreases the efficiency\[5,17\]. Learning expression by using a multi string alignment algorithm provides efficient result\[11\].

In \[23\] provided learning the extraction rules from the inputs with its annotations. Transducer was used to indicate the attributes to extract and conditions like when to start and stop the extraction. The matching process starts before the extraction completes which seems to be a drawback\[6\]. In \[1\] provides ontology based approaches to form the linked data\[24\]. The input is a set of documents that are annotated already to find simple regular expression which contains delimiters for indicating the amount of information that is to be extracted. Information extraction is a time consuming process\[19\]. In \[12\] presents an approach in web service for text mining. It discovers new facts using the automatic extraction algorithms which is perfectly belongs to the particular category. KNN algorithm is used here to find the categorization but it lags in efficiency\[12\]. In 2014 Joshila grace presents an approach for determining the web usage using the bookmarks, utility, downloads etc. It also detects the frequency of the navigations in the websites\[13\].

2. Description of the Proposed Methodology

This focuses on region separation and extracts information from them. Most clients only prefer relevant data. The Idea is to classify the contents like advertisements, header and footer information, copyright related details and other unrelated links in the web documents\[14\]. It has a standard location for each above in a web document like copyright should be at the bottom etc. Based on that it can eliminate those unwanted information. It is not easy to classify local noise which is present in the main content, like search boxes, filtering panel, navigation links, advertisements, etc. The authors stated that there are Noise Reduction and Content Extraction from Web Pages Using DOM Based Page Segmentation\[16\]. There are some tools to get the relevant information from the web but it is not validated for all pages\[18\].

The Major portion in document provides some template related content for user friendly style. The job is to extract the information from that template\[22\]. For this, authors proposed approaches like hand crafted information extractor, rule learning, heuristic based applications. Crafting information extractor needs so much of the work; rather it will not convince the efficient extraction\[25\]. Rule learning has work of learning the patterns of the input documents. It is quite a difficult process when the website is updated, its structure with latest designs. Heuristic based proposals depend on the proven results with plenty of combinatorial solutions so it will provide finest results. This is widely used in unsupervised methods, hence it can do auto check\[17\].

This paper comprises of two main functionalities that are Region separation and Information extraction. The formatting tags in web pages will lead to incorrect segmentation\[15\] so it needs to be removed from the web documents. Initially building a tag tree by ignoring the formatting tag is done\[24\]. From the tag-tree, the tree matching process is performed for finding the data regions. Based on a similarity threshold it can find the data records from the data regions called information extraction. Here it also eliminates redundancy. The architecture flow is provided in Figure 1. This new approach provides high efficiency and effectiveness by automatically extracts all the relevant data from the web using tree matching, tree alignment and mining methods\[16\]. Thus, there is no need of the web document to be annotated already. This will improve the accuracy by eliminating the redundancy and noise content which has been a problem with existing.

2.1 Processing and Checking the Input

Initially, it checks with the extension to get the compatible files for the extraction.

- Enquire (input: list, files; int: k, l)
- Str= {}  
- Int k=0
While list ≠ <> then do
• For list = max down to min do
  • If list[i] = endswith “.htm” then
    • Add list[i].getName to FileList
    • Add list[i].getAbsolutePath to FilePath
    • str=str+list[i].getName()
    • increment k
  • Else No Selection
  • Taglist[]=k3

2.2 Parsing the Document

The task of analyzing the strings and processing it to get the parsed data to present the data in a structured way.

1. Top-down parsing: It parses the input document and searches the parse trees to catch the left-most sources using a top-down expansion.
2. Bottom-up parsing: It parses the document to trace the supreme basic elements. This parsing is also called as Shift Reduce.

• Tree based parser
  The requirements of memory and resource will be more for large documents. Here incorrect HTML formatting tags are ignored. For example: DOM. But every node in document must be parsed from the top of the document to the bottom.

• Event based parser
  This type of parsers deals with the set of events. For example: SAX. It provides interface on high level and more intuitive.

• Tag based
  Tag parsing processes the document by analyzing the every ‘<’ character found and considered as a start tag and if that character followed by ‘\’ then it must be an end tag.

2.2.1 Jericho HTML

This is one of the java libraries for manipulating the parts of the HTML documents. It also includes the server side tags. It can also reproduce any unrecognized or invalid HTML. Hence badly structured HTML will not affect the parsing rest of the document. It has both event and tree based parsers characteristics. Provides efficient tag recognition and tag position cache. Initially the text of whole document was allowed to be stored in the memory and then searching the related sections for relevant data. It has built in features like extracting all the text from HTML mark up, HTML rendering with simple text formatting. Formatting the HTML source code by removing the unnecessary white space and compact the HTML source code. This is well suited for feeding the search engines such as apache lucene.

2.3 Filtering the Documents

Extracting the data from those documents needs some removal of HTML tags which could be common for all types of scripts.

• scanning the StopTags.txt and add in st
• return st
• creates the st_wrd arraylist
• read stop_wd and stores in st_wrd
• store the taglist.length in newlist1
• for taglist.length max down to min do
• ArrayList at1=TagList[i]
• Removes the stop word present in at1
• ArrayList at2=remove_stopwd(st_wrd,at1)

![Figure 1. Architecture of proposed approach.](image-url)
• Returns result
• Newlist1[i] → at2 (after removing stop tags)

2.3.1 Neko HTML
It is designed to be as light weight and ease of usage. It is one of the HTML scanner and Tag balancer for scanning and correcting the missing tags by closing and handles it automatically. Tim Weninger states that these processing will help to find the text to tag ratio.

2.4 Finding Tag Patterns
- create newlist2 and saves the length of taglist.
- create alllist
- take newlist1 (tags without stop tag) and saves as at1
- create at2
- get at's content and convert to string and save it to g1 to g8
- create String ms1 by concatenate all the sub strings g1 to g8
- if at2 !contains ms1 add ms1 to at2
- if alllist !contains ms1 add ms1 to alllist
- newList2[i] → at2 (tag patterns)

2.5 Expand the Patterns
- Expand(taglist : at1, at2; int k)
- For newList2.length max down to min do
  - ArrayList at1=newList2[i]
  - For k<newList2.length max down to 1 do
    - ArrayList at2=newList2[k];
    - int cn1=0;
    - if at1>at2
      - for(int j=0;j<at1.size();j++)
        - if(at2.contains(at1.get(j))) then cn1++ else
          - for(int j=0;j<at2.size();j++)
            - if(at1.contains(at2.get(j))) increment cn1+
    - combination of tags in documents i+k+at1.size+at2.
    - size+cn1
    - display allList & size (final pattern)

2.6 Multi String Alignment
newlist2= tag pattern
alllist = all tag text
- multistring(newlist2, alllist, cn=0)
- termWg[i][j]=new int[newList2.length][allList.size()]
- For newList2.length max down to min do
  - ArrayList lt1=newList2[i]
  - For all List. size max down to min do
    - String g1=all List. get(j). to String()
    - For lt1.size max down to min
      - String g2=lt1.get(k). to String()
      - if(g1.equals(g2)) then cn++
      - term Wg [i][j] --> cn;
      - Array List sim1=new Array List();
      - For newList2.length max down to min
        - double cs=0
        - String g1=String. value Of (i)
        - For (int j=i+1; j<newList2.length; j++)
          - int muld, td1, td2 = 0;
          - For (int k=0; k<all List. size(); k++)
            - k1->term Wg [i][k] & k2->term Wg [j][k]
            - muld=muld+(k1*k2)
            - td1=td1+(k1*k1) & td2=td2+(k2*k2)
            - d1=Math.sqrt(td1) & d2=Math.sqrt(td2)
            - divd=d1*d2;
            - cos=muld/divd;
            - sin=Math.sqrt(1-(cos*cos));
            - cs=cs+cos;
            - sim += “+i+” : “+j+” : “+cos+” : “+sin
            - if(cos>0.9)
              - g1=g1+”#”+j;
              - sim1->g1 & sim2->g1
      - String g1=sim1.get(i).toString();
      - String g2=sim1.get(j).toString();
      - if(g1 && sim2 contains g2)
        - remove(g2) from sim2
      - return sim2

2.7 Region Separation
The Information extractor has difficulty in separating the information from the entire document. Region extractor heals this situation and enhances the extraction process. Initially, it has the res buffer as empty. Constructing the tag trees by avoiding the formatting tags and matches with similarity threshold. It can find the data regions which have relevant data. Using max node in generalized, separate the data regions and stores in data record array counter.

- region separation (data region, data record)
  - res={}
  - Formatter (result Output)
    - starts ignore Formatting Tags
    - construct Tag Tree (input, ignore Formatting Tags)
    - Simple Tree Matching (similarity threshold, max Node In Generalized Nodes)
2.8 Classifying the Extracted Text

Stanford NER classifier is a java implementation of a named entity recognizer label. It works in the sequence of words in text such as persons, company, money etc.

- classify (inputs:s2; array list: person, loc, time, org, money, date)
- String cls = “english.muc.7class.distsim.crf.ser.gz”
- Calling CRF Classifier. get Classifier No Exceptions (cls)
- For sim2.size() max down to min
- String gs->sim2.get(i).toString()
- If gs contains("#")
- g2[]->gs.split("#")
- for full g2.length
- h1->Integer.parseInt(g2[j])
- nn=FilePath.get(h1).toString()
- save source as htm
- convert non html elements
- res->classifier. classify To String (text)
- sp1[]->res. split
- s1->sp1[i1].trim();
- if(s1.endsWith("PERSON"))
- if(!person.contains(s2))
- s2->add person
- checks the s1 classes (LOCATION, TIME, DATE, MONEY, ORGANIZATION) in s2 if not
- add classes to s2
- return s2

3. Evaluation Measure

Our approach could extract more relevant terms than the irrelevant data items and it extracts from all the inputs. The F measure is based on the precision and recall of retrieved information. The higher the F measure will have more accuracy. Precision is the ratio of extracted data that are important (Precision=Is/S). Recall is the ratio of important data that are extracted (Recall=Is/I). Where S is the count of extracted sections. I is the count of important sections. Is is the count of extracted important sections. Here each webpage is treated as set of documents and each region are as desired result. The calculated measures will compare with one of the best approaches in an existing system to prove our approach has high efficiency.

![Figure 3. Efficiency result of hybrid approach.](image-url)
Overall f-measure (F) calculated by average of all values,

\[ F = \frac{1}{t} \sum_{o} \frac{n_{t} \times \max[f(d_{c}, s_{c})]}{n} \]

Where \( t \) is the total length of the document, \( n \) is the total volume of documents.

### 4. Conclusion and Future Work

The proposed hybrid approach for extracting the web information is the most efficient method. Our implementation with the net beans IDE 8.0 performed experiments on collection of web documents. Figure 2 these were randomly selected from the internet and perform the proposed hybrid approach and gets the classified reliable results. It may have some additional data due to inappropriate use of terms and symbols. In future this method could be improved with the more collection of diverse web documents with even more higher efficiency. Because for each and every kind of documents need separate algorithms.

### 5. References

1. Sleiman HA, Corchuelo R. TEX an efficient and effective unsupervised web information extractor. Knowledge-Based Systems. 2013.
2. Shah K, Fouzia J, Mashwani SR, Alam I. Linked open data towards the realization of semantic web a review. Indian Journal of Science and Technology. 2014 Jun; 7(6):745–64.
3. Johnson F, Gupta SK. Web content mining techniques a survey. International Journal of Computer Applications. 2012 Jun; 47(11):44–50.
4. Ananthi J. A survey web content mining methods and applications for information extraction from online shopping sites. International Journal of Computer Science and Information Technologies (IJCST). 2014; 5(3):4091.
5. Hing JL, Siew EG, Egerton S. Information extraction for search engines using fast heuristic techniques. Data and Knowledge Engineering. 2010; 69(2):169–96.
6. Ferraraa E, De Meob P. Web data extraction applications and techniques: a survey. Knowledge Based Systems. 2014 Jun; 70:301–23.
7. Zheng X, Gu Y, Li Y. Data extraction from web pages based on structural semantic entropy. International World Wide Web Conference Committee (IW3C2); 2012 Apr. p. 93–102.
8. Miao G, Tatemura J, Hsiung WP, Sawires A, Moser LE. Extracting data records from the web using tag path clustering. International World Wide Web Conference Committee (IW3C2); 2009 Apr. p. 981–90.
9. Sleiman HA, Corchuelo R. An unsupervised technique to extract information from semi structured web pages. Web Information Systems Engineering WISE. 2012; 7651:631–7.
10. Crescenzi V, Mecca G. Automatic information extraction from large websites. JACM. 2004; 51(5):731–79.
11. Kushmerick N, Weld DS, Doorenbos RB. Wrapper induction for information extraction. IJCAI. 1997.
12. Mustafa AS, Kumaraswamy YS. Performance evaluation of web services classification. Indian Journal of Science and Technology. 2014 Oct; 7(10):1674–81.
13. Ghobadi A, Rahgozar M. An ontology based semantic extraction approach for B2C eCommerce. The International Arab Journal of Information Technology. 2011 Apr; 8(2):163–70.
14. Upadhyay GM, Dhingra K. Web content mining its techniques and uses. International Journal of Advanced Research in Computer Science and Software Engineering. 2013 Nov.
15. Weninger T. Text Extraction from the Web via Text-to-Tag Ratio; 2009.
16. Zhai Y. Web data extraction based on partial tree alignment. International World Wide Web Conference Committee (IW3C2); 2005 May. p. 76–85.
17. Kayed M, Hui CC. FiVaTech page level web data extraction from template pages. IEEE Transaction on Knowledge and Data Engineering. 2010.
18. Kaur S, Tyagi A. Noise reduction and content extraction from web pages using DOM based page segmentation. JCTA. 2014.
19. Sleiman HA, Corchuelo R. Towards a method for unsupervised web information extraction. ICWE. 2012; 7387:427–30.
20. Gulhane P, Rastogi R, Sengamedu SH, Tengli A. Exploiting content redundancy for web information extraction. IEEE Transaction on Knowledge and Data Engineering. 2010.
21. Sleiman HA, Corchuelo R. A survey on region extractors from web document. IEEE Transaction on Knowledge and Data Engineering. 2012; 25(9):1960–81.
22. Liu W, Meng X, Meng W. ViDE a vision based approach for deep web data extraction. IEEE Transactions on Knowledge and Data Engineering. 2010 Mar; 22(3): 447–60.
23. Hsu CN, Dung MT. Generating finite state transducers for semi structured data extraction from the Web. Information Systems. 1998; 23(8):521–38.
24. Sunil Kumar T, Suvarchala K. A study web data mining challenges and application for information extraction. IOSR Journal of Computer Engineering (IOSRJCE). 2012; 7(3):24–9.
25. Sleiman HA, Corchuelo R. A reference architecture to devise web information extractors CAiSE workshops. Advanced Information Systems Engineering Workshops. 2012; 112:235–48.