USING PARSER PERFORMS SEGMENTATION OF WEB PAGES AND EXTRACTION OF TEMPLATES

[1]M.Subba Rao, [2]M.Roberts Masillamani, [3]Teena Joseph

[1]Professor, [2]Associate Professor, [3]Associate Professor,
[1,2,3]Department of Computer Science and Engineering, AITS, Rajampet, India.

Abstract : Many Web sites contain much explicit and implicit structure, both in layout and content that we can exploit for the purpose of information extraction. This paper describes an approach to *automatic* extraction and segmentation of records from Web tables. Automatic methods do not require any user input, but rely solely on the layout and content of the Web source. Our approach relies on the common structure of many Web sites, which present information as a list or a table, with a link in each entry leading to a detail page containing additional information about that item. We describe two algorithms that use redundancies in the content of table and detail pages to aid in information extraction. The first algorithm encodes additional information provided by detail pages as constraints and finds the segmentation by solving a constraint satisfaction problem. The second algorithm uses probabilistic inference to find the record segmentation. We show how each approach can exploit the web site structure in a general, domain-independent manner, and we demonstrate the effectiveness of each algorithm on a set of twelve Web sites.

***IndexTerms* – Parser, Web Pages, Extract.**

1. INTRODUCTION

The World Wide Web is a vast repository of information. The amount of data stored in electronic databases accessible to users through search forms and dynamically generated Web pages, the so-called *hidden Web* [26], dwarfs the amount of information available on static Web pages. Unfortunately, most of this information is presented in a form accessible only to a human user, *e.g.*, list or tables that visually lay out relational data. Although newer technologies, such as XML and the Semantic Web, address this problem directly, only a small fraction of the information on the Web is semantically labeled. The overwhelming majority of the available data has to be accessed in other ways.

Web wrappers are popular tools for efficiently extracting information from Web pages. Much of the research in this area over the last decade has been concerned with quick and robust construction of Web wrappers, usually with the help of machine learning techniques. Because even the most advanced of such systems learn correct wrappers from examples provided by the user, the focus recently has been on minimizing the number of examples the user has to label, *e.g.*, through active learning. Still, even when user effort is significantly reduced, the amount and the rate of growth of information on the Web will quickly overwhelm user resources. Maintaining wrappers so that they continue to extract information correctly as Web sites change requires significant effort, although some progress has been made on automating this task [1].

Extraction of records or tuples of data from lists or tables in HTML documents is of particular interest, as the majority of Web sites that belong to the hidden Web are presented in this manner. Record extraction is required for a multitude of applications, including web data mining and question answering.

The main challenge to automatic extraction of data from tables is the great variability in HTML table styles and layout. A naive approach based on using HTML <table> tags will not work. Only a fraction of HTML tables are actually created with <table> tags, and these tags are also used to format multi-column text, images, and other non-table applications. The vast majority of HTML documents use non-standard tags to format tables, including text separators, such as ~, to separate fields and
 to separate different items as well as fields. More sophisticated automatic approaches to table recognition and information extraction have been suggested which rely on the Document Object Model [13] or regularities in HTML tags.

2. RELATEDWORK

Several researchers have addressed the problem of detecting tables in Web and plain text documents and segmenting them into records.

2.1 Table Extraction from HTML Documents

Existing approaches to extracting table data from Web documents can be classified as heuristic or machine learning. Heuristic approaches to detecting tables and record boundaries in Web documents include using the Document Object Model (DOM) and other features to identify tables.

Domain-specific heuristic rules that rely on features such as percent signs and date/time formats have also been tried successfully.

Machine-learning approaches learn a model of data from a set of labeled training examples using hand-selected features. Brokers et al. use multiple heuristic features, including domain-specific controlled vocabularies, to learn a Hidden Markov-based probabilistic model from a set of training examples. Hurst trained a Naive Bayes classifier, while Wang et al. describe a domain-independent classifier that uses non-text layout features (average number of columns/rows, average cell length and consistency) and content features (image, form, hyperlink, alphabetic, digit, others).

2.2 Table Extraction from Plain Text

Automatic table extraction from plain text documents is a line of research parallel to the work on HTML table extraction. There are differences between plain text and HTML tables that make the two fundamentally different problems. Plain text documents use white space and new line for the purpose of formatting tables: new lines are used to separate records and white spaces are used to separate columns, among other purposes. Record segmentation from plain text documents is, therefore, a much easier task. Closely

3. OVERVIEW OF THE PROBLEM

In this section we give an overview of the problem of record extraction and segmentation using the structure of a Web site to aid in extraction. As we discussed above, many Web sites that present information contained in databases follow a *de facto* convention in displaying information to the users and allowing them to navigate it. This convention affects how the Web site is organized, and gives us additional information we can leverage for information extraction. Such Web sites generate list and detail pages dynamically from templates and fill them with results of database queries. Figure 1 shows example list and detail pages from the Verizon Superpages site. The Superpages site allows customers to search over 16 million yellow page listings and a national white pages database by name, phone number or business type. As shown in the figure, the results returned for a search include the fields, name, address, city, state, zip and phone. Here the text "More Info" serves as a link to the detail page. Note that list and detail pages present two views of the record. Using automatic techniques, we can potentially combine the two views to get a more complete view of the record. For example, maps of the addresses are shown on the detail pages in Figure 1, but absent from the list pages.

3.1 Page Templates

Consider a typical list page from a Web site. As the server constructs the list page in response to a query, it generates a header containing the company logo, followed in most cases by an advertisement, then possibly a summary of the results, such as ''Displaying 1-10 of 214 records.'', table header and footer, followed by some concluding remarks, such as a copyright information or navigation aids.

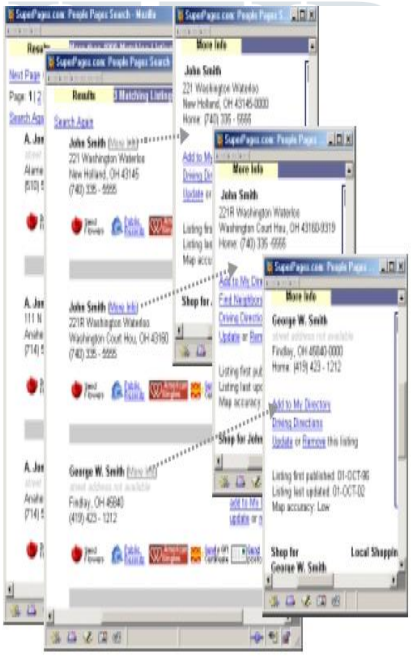

Figure 1: Example list and detail pages from the Superpages site (identifying information has been removed to preserve confidentiality).

In the above example, the header includes Results, 3 Matching Listings, Search Again and the associated HTML. The footer includes advertisement and navigational links. We call this part of the page the *page template*. The page template of a list page contains data that is shared by all list pages and is invariant from page to page. A different page template is used to generate detail pages. As the server writes out records to the table, it uses a *table template*.

This template contains layout information for the table, that is the HTML tags or ASCII characters used to separate columns and rows, or format attribute values.

Given two, or preferably more, example list pages from a site, we can derive the template used to generate these pages and use it to identify the table and extract data from it. The process starts with a set of list pages from a site and a set of detail pages obtained by following links from one of the list pages. The pages are tokenized — the text is split into individual words, or more accurately tokens, and HTML escape sequences are converted to ASCII text. Each token is assigned one or more syntactic types [1], based on the characters appearing in it. The three basic syntactic types we consider are: HTML, punctuation, and alphanumeric. In addition, the alphanumeric type can be either numeric or alphabetic, and the alphabetic can be capitalized, lowercased or allcaps. This gives us a total of eight (non-mutually exclusive) possible token types. A template finding algorithm (*e.g.*, one described in [1, 18]) is used to extract the main page template. *Slots* are sections of the page that are not part of the page template. If any of the tables on the pages contain more than two rows, the tags specifying the structure of the table will not be part of the page template, because they will appear more than once on that page. Likewise, table data will also not be part of the page template, since it varies from page to page. Therefore, the entire table, data plus separators, will be contained in a single slot. Considering that tables usually contain a significant amount of data, we use a heuristic that the table will be found in the slot that contains the largest number of text tokens.

4. A CSP APPROACH TO RECORD SEGMENTATION

CSPs are stated as logical expressions, or constraints, over a set of variables, each of which can take a value from a finite domain (Boolean, integer, *etc.*). The case where the variables and logical formulas are boolean is known as Boolean satisfiability, the most widely studied area of CSP.

In a pseudo-boolean representation, variables are 0-1, and the constraints can be inequalities. The CSP problem consists of finding the value assignment for each variable such that all constraints are satisfied at the same time. When constraints are inequalities, the resulting problem is an optimization problem.

5. A PROBABILISTIC APPROACH TO RECORD SEGMENTATION

An alternate approach is to frame the record segmentation and extraction task as a probabilistic inference problem. Common probabilistic models for information extraction include hidden Markov models (HMMs) [25], and conditional random fields (CRFs) [15, 23]. In these approaches, a labeled training set is provided and the model is learned using standard probabilistic inference techniques; because the state is *hidden*, the common approach to learning the models is to use the expectation maximization (EM) algorithm.

While these approaches have their computational limitations, they have been applied successfully to a wide range of problems beyond information extraction including speech recognition and robot navigation. Unfortunately, here we are faced with a more challenging problem. We do **not** have a labeled training set to start from. The key to our success will be to:

Factor: We will factor the state space and observation set of the HMM to allow for more efficient learning (because fewer parameters will be required).

Bootstrap: We will use the information from the detail pages to help bootstrap the learning algorithm. The constraints from the detail extracts will provide useful information that can keep our learning algorithm on track.

Structure: We will use a hierarchical model to capture global parameters such as the length of the record, or the period, to make our inference more tractable. Note that while there is a global record length, the record lengths of the individual records may vary; for some records not all columns will be presented.

Table 3: Positions of extracts on detail pages. Entry of 1 **means extract** *Ei* **was observed at position** *k* **on page** *j* **(***poskj***).**

	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8
pos_1^{730}	1				1			
pos_1^{772}		1						
pos_1^{812}			1					
pos_1^{846}				1				1
pos_2^{536}	1				1			
pos_2^{578}				1				1
pos_2^{608}						1		
pos_2^{642}							1	

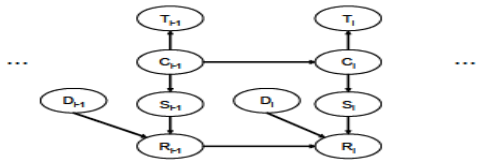

Figure 2: A probabilistic model for record extraction from list and detail pages.

6. DISCUSSION/RESULTS

Each approach has its benefits and drawbacks, which make them suitable in different situations. The CSP approach is very reliable on clean data, but it is sensitive to errors and inconsistencies in the data source. One such source of data inconsistency was observed on the Michigan corrections site, where an attribute had one value on the list pages and another value on the detail pages. This by itself is not a problem; however, the list page string appeared on one detail page in an unrelated context. The CSP algorithm could not find an assignment of the variables that satisfied all the constraints. The probabilistic approach, on the other hand, tolerates such inconsistencies and is more expressive than the CSP representation. Its expressiveness gives us the power to potentially assign extracts to individual attributes, and, when combined with a system that automatically extracts column labels [2] from tables, reconstruct the relational database behind the Web site.

	Probabilistic				CSP				
Wrapper	Cor	InC	FN	FP	Cor	InC	FN	FP	notes
Amazon	4	6	0	1	0	0	10	0	a, b
Books	2	5	3	4	0	0	10	0	
BN	5	5	0	0	2	0	8	0	a, b, c, d
Books	5	5	0	0	0	0	10	0	
Allegheny	20	0	0	0	20	0	0	0	
County	16	4	0	0	20	0	0	0	
Butler	15	0	0	0	15	0	0	0	
County	12	0	0	0	12	0	0	0	
Lee	16	0	0	0	16	0	0	0	
County	5	0	0	0	5	0	0	0	
Michigan	7	0	0	0	4	3	0	0	
Corrections	12	4	0	0	2	8	6	0	c, d
Minnesota	11	0	0	0	4	7	0	0	a, b, c, d
Corrections	17	2	0	0	8	9	0	2	
Ohio	8	2	0	0	10	0	0	0	
Corrections	10	0	0	0	10	0	0	0	
Canada	18	7	0	0	25	0	0	0	
411	1	4	0	0	1	4	0	0	c, d
Sprint	17	3	0	0	20	0	0	0	
Canada	8	12	0	0	20	0	0	0	
Yahoo	0	10	0	0	5	5	0	0	a, b, c, d
People	10	0	0	0	10	0	0	0	b
Super	3	0	0	0	3	0	0	0	a, b
Pages	9	6	0	0	15	0	0	0	
Precision	0.74				0.85				
Recall	0.99				0.84				
F	0.85				0.84				

Notes
a. Page template problem; b. Entire page used; c. No solution found;
d. Relax constraints

Both techniques (and a combination of the two) are likely to be required for robust and reliable large-scale information extraction. We stress that the approaches are novel in that they are domain independent, unsupervised, and rely on the content of Web pages rather than their layout.

7. CONCLUSION

There are multiple ways to represent and leverage the additional information contained in the structure of Web sites. In this work we investigated two of them: 1) a logic-based approach in which we encode the information provided by detail pages as constraints and solve them to obtain the record segmentation, and 2) a probabilistic inference approach in which we represent the observations and structure of the table as a probabilistic model and use an inference algorithm to find appropriate segmentation. Both approaches have widely used, efficient algorithms for solving problems. Each has its benefits and drawbacks, that make them preferable in different situations. The constraint-satisfaction approach is very reliable on clean data, but it is sensitive to errors and inconsistencies in the data source. The probabilistic approach on the other hand, tolerates inconsistencies and is more expressive than the constraint-based approach, and, beyond record segmentation, it can perform record extraction. Both techniques (or a combination of the two) are likely to be required for large-scale robust and reliable information extraction.

8. REFERENCES

[1] A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of data*, 2003.
[2] L. Arlotta, V. Crescenzi, G. Mecca, and P. Marialdo. Automatic annotation of data extracted from large web sites. In *Proceedings of the Sixth International Workshop on Web and Databases (WebDB03)*, 2003
[3] V. Brokers, K. Deshmukh, and S. Sarawagi. Automatic segmentation of text into structured records full text. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, 2001.
[4] C. H. Chang, and S. C. Lui. IEPAD: Information Extraction based on Pattern Discovery. In *10th International World Wide Web Conference (WWW10), Hong Kong*, 2001.
[5] H. Chen, S. Tsai, and J. Tsai. Mining tables from large scale html texts. In *18th International Conference on Computational Linguistics (COLING)*, 2000.
[6] W. W. Cohen, M. Hurst, and L. S. Jensen. A Flexible Learning System for Wrapping Tables and Lists in HTML Documents. In *11th International World Wide Web Conference (WWW10), Honolulu, Hawaii*, 2002.
[7] V. Crescenzi, G. Mecca, and P. Merialdo. Automatic web information extraction in the roadrunner system. In *Proceedings of the International Workshop on Data Semantics in Web Information Systems (DASWIS-2001)*, 2001.
[8] V. Crescenzi, G. Mecca, and P. Merialdo. RoadRunner: Towards automatic data extraction from large web sites. In *Proceedings of the 27th Conference on Very Large Databases (VLDB)*, Rome, Italy, 2001.
[9] C. Gazen. Thesis proposal, Carnegie Mellon University.
[10] Z. Ghahramani and M. I. Jordan. Factorial hidden Markov models.