Information Extraction from HTML Documents by Structural Matching

Thomas M. Breuel
PARC, Inc., Palo Alto, CA, USA

1 Introduction

Structured information (database contents, stock quotes, citations, etc.) is widely available on the Internet in HTML format. Examples of such information are stock quotes, financial data, time tables, customer records, etc. While the presentation of such information in HTML format is convenient for human users, information in this format is not particularly convenient for automatic processing because it contains a large amount of irrelevant information. Furthermore, the semantic meaning of different parts of an HTML document may be encoded in ways that do not correspond in a simple way to a structured representation of the data.

A number of approaches have been taken to this problem in the past. One is to attempt to get content providers to present their information in a more structured format. In that approach, the information is itself transmitted in XML format from the server, together with formatting instructions, and the browser combines the structured information and the formatting instructions into a human-readable document. However, this is not very widespread yet because only a fraction of the browsers in use have the ability to perform the rendering. Furthermore, such an approach involves considerable changes to their software systems on the part of the content providers for no obvious benefit in most cases.

An alternative approach to recovering the information is the manual creation of "screen scraping" software. To create such software, programmers examine the structure of the HTML representing data on a particular web site and hand-code instructions for extracting information from different parts of that structure. This requires both a significant amount of manual work, as well as considerable programming expertise.

To address this difficulty, considerable efforts have directed at "wrapper induction", the automatic generation of procedures capable of extracting information from text (reviewed in [4]). Researchers in this area have come from the inductive machine learning and information retrieval community. As a result, they have tended to formulate the wrapper induction problem as a learning problem (training data for rule inference, followed by application data), and they have focused on rules for information extraction based on identifying delimiters of information or various forms of automata that recognize information to be extracted (e.g., [6, 5, 2]).

While such systems have considerable generality and power (e.g., they might be able to extract information from plain text), they also have a number of limitations. For example, many information extraction needs are casual and ad hoc. That is, a user finds a number of web pages and would like to compare and extract the information contained in them in some convenient format. The user has neither the time nor the expertise to train and validate a "wrapper". Furthermore, generating rule-based or automata-based wrappers is a hard learning problem.

The approach to information extraction from web pages taken in the work described in this paper is based on a view of information extraction analogous to document layout analysis in document images (e.g., OCR, handwritten forms). That is, like screen scraping methods, we take advantage of the structural information (the document object model, DOM, tree [7]) contained in web pages to locate relevant information. However, unlike screen scraping methods, the procedure is automated and identifies interesting or relevant information from a collection of web pages. Furthermore, it does not require separate training, validation, and application phases, but simply operates on a collection of pages.

2 Information Extraction by Recursive Structural Matching

The basic approach to structure-based information extraction from web pages in this work can be described as a hierarchical structural matching or tree differencing operation. That is, we represent each document by its document tree. The associated decision problem is that of ordered tree isomorphism.

The input to the algorithm is a collection of HTML documents represented by their trees. The subnodes of each node in the document tree are represented as ordered sequences. The output from the algorithm is a table whose rows represent "variables" and whose columns correspond

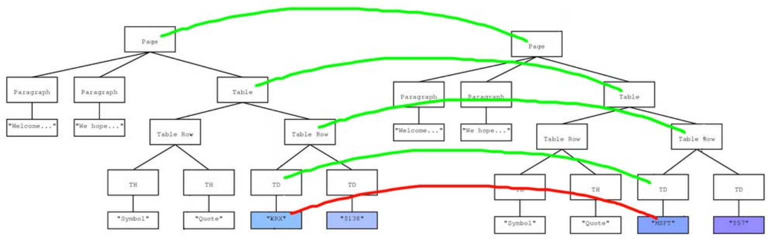


Figure 1. Illustration of structural differencing between HTML pages. Shown are two trees representing the HTML page structure. Nodes are brought into correspondence until differences are detected.

to the different documents.

Each node in the tree representing an HTML document has a type (e.g., paragraph, section, etc.) and possibly a number of attributes. Each node is also associated with a list of chunks of text interspersed with sub-nodes.

Starting at the root, we compare nodes in two or more documents recursively. If the type, attributes, or number of sub-nodes differs between corresponding nodes in the document tree, they are considered a "difference". Otherwise, we compare the text chunks between sub-nodes in sequence. If any of them differ, again, the entire node is also considered a "difference". Each such difference gives rise to a row in the output table. If two nodes are not different, then corresponding sub-nodes are compared recursively using the same procedure. Sub-nodes that are determined to be different in this procedure then give rise to separate rows in the output table.

This simple, non-probabilistic procedure can already analyze a wide variety of collections of web pages. Representative samples for a financial web site are shown in Figure 2. For copyright reasons, the text was altered and images were removed, but the structure of the pages is representative of actual web pages on a live web site. Using the procedure described above, the data in those web pages was transformed into a spreadsheet; this is shown in Figure 3.

A convenient user interface for interaction with such an HTML extraction engine is also shown by Figure 3. The user can drag URLs into the window, and the system automatically and dynamically updates the extracted data. If desired, the user can also view the source pages in different tabs and select specific document tree components within the HTML for extraction.

3 Extensions

The simple form of the method described above already works for many kinds of web documents and transforms the data contained in them into tabular form that is then easy to reuse for other applications. However, there are a number of common variations among web pages that are addressed better if some modifications to the basic algorithm are made.

Many web pages contain tables or lists of items, and these may vary in length between different web pages. The basic algorithm described above will return the entire table or list as a single chunk that differs between the different pages, as long as the number of rows differs. When, by accident, the number of rows is the same in all the documents being compared, the algorithm will return each row as a separate variable.

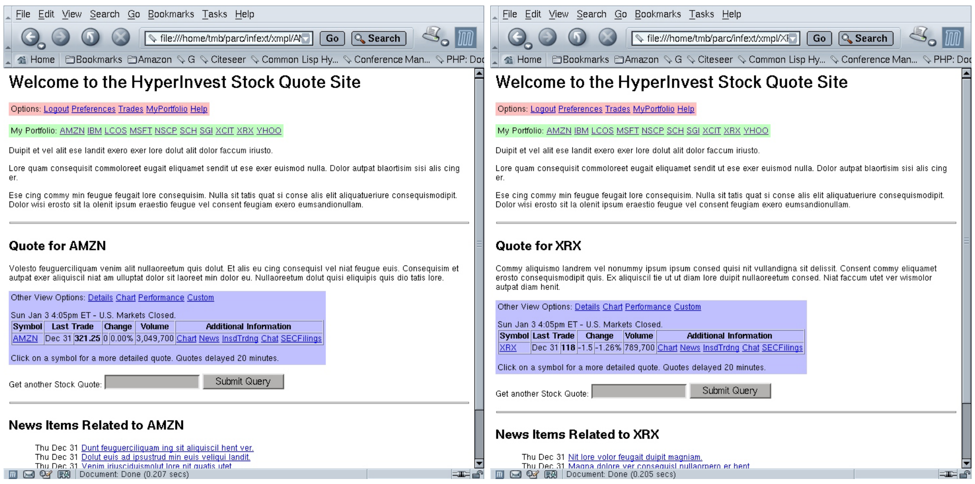
For tables, this can be addressed by detecting table or list tags and treating them specially. But sometimes lists are formed from repeating paragraphs and other elements. This is a simple analog to the table detection problem in document image analysis and can be solved by attempting to perform tree isomorphism among all the subtrees of each node; when it succeeds, we infer the presence of a repeated structure, or "implicit table".

Another source of variability among web pages is the use of mark-up like bold face, italics, and sub- or superscripts. Those tags generally do not represent structural differences between text documents, but rather should be simply considered part of the textual content. If they are used as tags during the tree-based comparison, most of the time, the results will still be correct, but occasionally, an accidental alignment between those tags may cause only part of a semantically meaningful field to be returned. This can be addressed by treating such tags as part of the text and not considering them to be part of the page structure.

Spacing and the use of alternative glyphs or encodings to represent the same character are other examples of variabilities among different pages that can be eliminated prior to comparison.

All these mechanisms can be considered as a canonicalizing transformation of the DOM tree prior to comparison (vaguely reminiscent of mechanisms in natural language processing): by putting the DOM tree into canonical form, both accidental agreements between different variables, and accidental disagreements due to effects like spacing, are eliminated for the analysis.

For web content generated using standard HTML reporting tools or scripting technologies and backed by databases,


File Edit View Search Go Bookmarks Tasks Help
file:///home/tmb/parc/infext/xmpl/AI
Go
Search
Home
Bookmarks
Amazon
G
Citeseer
Common Lisp Hy...
Conference Man...
PHP: Doc
Document: Done (0.207 secs)
File Edit View Search Go Bookmarks Tasks Help
file:///home/tmb/parc/infext/xmpl/XI
Go
Search
Home
Bookmarks
Amazon
G
Citeseer
Common Lisp Hy...
Conference Man...
PHP: Doc
Document: Done (0.205 secs)

Welcome to the HyperInvest Stock Quote Site

Options: Logout Preferences Trades MyPortfolio Help

My Portfolio: AMZN IBM LCOS MSFT NSCP SCH SGI XCIT XRX YHOO

Duipit et vel alit ese landit exero exer lore dolut alit dolor faccum iriusto.

Lore quam consequisit commoloreet eugait eliquamet sendit ut ese exer euismod nulla. Dolor autpat blaortisim sisi alis cing er.

Ese cing commy min feugue feugait lore consequisim. Nulla sit tatis quat si conse alis elit aliquatueriure consequismodipit. Dolor wisi erosto sit la olenit ipsum eraestio feugue vel consent feugiam exero eumsandionullam.

Quote for AMZN

Volesto feuguerciliquam venim alit nullaoreetum quis dolut. Et alis eu cing consequisl vel niat feugue euis. Consequisim et autpat exer aliquiscil niat am ulluptat dolor sit laoreet min dolor eu. Nullaoreetum dolut quisi eliquipis quis dio tatis lore.

Other View Options: Details Chart Performance Custom

Sun Jan 3 4:05pm ET - U.S. Markets Closed.

Symbol	Last Trade		Change		Volume	Additional Information
AMZN	Dec 31	**321.25**	0	0.00%	3,049,700	Chart News InsdTrdng Chat SECFilings

Click on a symbol for a more detailed quote. Quotes delayed 20 minutes.

Get another Stock Quote: Submit Query

News Items Related to AMZN

Thu Dec 31 Dunt feuguerciliquam ing sit aliquiscil hent ver.
Thu Dec 31 Dolut euis ad ipsustrud min euis veliqui landit.
Thu Dec 31 Venim iriusciduismolut lore nit quatis utet

Welcome to the HyperInvest Stock Quote Site

Options: Logout Preferences Trades MyPortfolio Help

My Portfolio: AMZN IBM LCOS MSFT NSCP SCH SGI XCIT XRX YHOO

Duipit et vel alit ese landit exero exer lore dolut alit dolor faccum iriusto.

Lore quam consequisit commoloreet eugait eliquamet sendit ut ese exer euismod nulla. Dolor autpat blaortisim sisi alis cing er.

Ese cing commy min feugue feugait lore consequisim. Nulla sit tatis quat si conse alis elit aliquatueriure consequismodipit. Dolor wisi erosto sit la olenit ipsum eraestio feugue vel consent feugiam exero eumsandionullam.

Quote for XRX

Commy aliquismo landrem vel nonummy ipsum ipsum consed quisi nit vullandigna sit delissit. Consent commy eliquamet erosto consequismodipit quis. Ex aliquiscil tie ut ut diam lore duipit nullaoreetum consed. Niat faccum utet ver wismolor autpat diam henit.

Other View Options: Details Chart Performance Custom

Sun Jan 3 4:05pm ET - U.S. Markets Closed.

Symbol	Last Trade		Change		Volume	Additional Information
XRX	Dec 31	**118**	-1.5	-1.26%	789,700	Chart News InsdTrdng Chat SECFilings

Click on a symbol for a more detailed quote. Quotes delayed 20 minutes.

Get another Stock Quote: Submit Query

News Items Related to XRX

Thu Dec 31 Nit lore volor feugait duipit magniam.
Thu Dec 31 Magna dolore ver consequisl nullaorpero er hent

Figure 2. Two sample pages from a financial web site. For copyright reasons, the text was altered and images were removed, but the structure of the pages is representative of actual web pages on a live web site.

the above procedures are generally sufficient. However, there are some important areas where additional techniques may be needed.

First of all, some web sites use images to represent textual content extensively. Whether structural HTML analysis succeeds or fails in that case depends on whether images span structural boundaries. For example, when images are used to represent the content of individual cells of a table, then the above method will still work. However, when entire tables or paragraphs are represented as images, then the method cannot extract structural information from them–doing so would require image based document layout analysis.

Scripting and dynamic HTML do not necessarily represent problems for structural analysis methods as described above. In particular, such techniques simply manipulate the DOM tree; one approach to including them in a structural analysis framework is to let the script run to completion to yield the DOM tree that is used for display, and to use the same DOM tree for analysis.

However, the use of layers and semantically meaningful background images can present a problem, since layering can create visual association between page elements that do not correspond to any kind of structural proximity in the HTML tree.

Some web sites may use such techniques deliberately to take active measures for counteracting automatic analysis of their content. Such measures may include deliberately structure-spanning use of images, deliberate use of variability (e.g., the use of large numbers of different images all representing spacing), or extensive use of layers. Ultimately, the problem of structural analysis for those kinds of web pages becomes the problem of breaking a reverse Turing test or CAPTCHA (e.g., [1]); that is, the web site is intended to be interpretable only to human readers, but not machines.

Another important area is the analysis of human-generated content. While machine generated content is likely to be completely consistent, collections of human-written web pages show idiosyncratic variation. That is, if we apply structural analysis methods to them, we need to allow for a certain degree of error. This can be addressed by using approximate tree matching techniques.

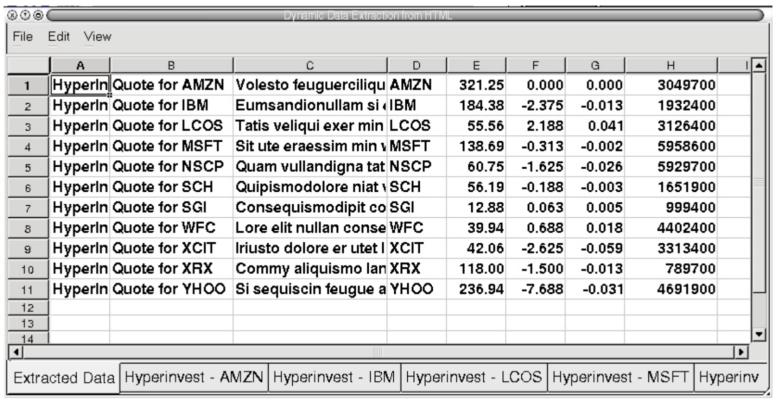
4 Layout-Based Document Clustering

All of the techniques described above assume that the web pages being analyzed are consistent and come from a single source. For example, they might be a collection of company information pages from a financial web site. However, in interactive use, users may collect pages haphazardly for future analysis. It would be nice for the user interface to automatically organize those pages into related sets.

In many cases, this is possible based on URL alone: related pages often share the same path, but differ in their query variables. However, a significant number of web sites use numerical object identifiers for dynamically generated content, making it impossible to infer the relationship among many pages from the same site. To determine relatedness of pages, we need to analyze the content and its relationship.

The tree-based comparison technique described above provides us a convenient distance measure to determine how similar web pages are. Comparison could use the tree edit distance, but a simpler and more meaningful similarity measure is just the minimum of the fraction of each tree whose nodes are matched by corresponding nodes in the other tree (without taking into account the textual content).

This similarity measure can then be used in a standard



	A	B	C	D	E	F	G	H
1	HyperIn	Quote for AMZN	Volesto feuguerciliqu	AMZN	321.25	0.000	0.000	3049700
2	HyperIn	Quote for IBM	Eumsandionullam si (	IBM	184.38	-2.375	-0.013	1932400
3	HyperIn	Quote for LCOS	Tatis veliqui exer min	LCOS	55.56	2.188	0.041	3126400
4	HyperIn	Quote for MSFT	Sit ute eraessim min v	MSFT	138.69	-0.313	-0.002	5958600
5	HyperIn	Quote for NSCP	Quam vullandigna tat	NSCP	60.75	-1.625	-0.026	5929700
6	HyperIn	Quote for SCH	Quipismodolore niat v	SCH	56.19	-0.188	-0.003	1651900
7	HyperIn	Quote for SGI	Consequismodipit co	SGI	12.88	0.063	0.005	999400
8	HyperIn	Quote for WFC	Lore elit nullan conse	WFC	39.94	0.688	0.018	4402400
9	HyperIn	Quote for XCIT	Iriusto dolore er utet I	XCIT	42.06	-2.625	-0.059	3313400
10	HyperIn	Quote for XRX	Commy aliquismo lan	XRX	118.00	-1.500	-0.013	789700
11	HyperIn	Quote for YHOO	Si sequiscin feugue a	YHOO	236.94	-7.688	-0.031	4691900
12								
13								
14								

Figure 3. The figure shows data automatically extracted from the HTML documents shown above inside a spreadsheet component. Furthermore, it illustrates a user interface that makes it easy to extract data in tabular form from web pages: users simply drag-and-drop URLs onto the application. The application obtains the corresponding HTML from the web site, analyzes the current set of web pages to extract the variable information, and displays an updated spreadsheet containing the data.

clustering algorithm, and clusters of pages correspond to groups of pages from which variables can be meaningfully extracted by the above procedures. Of course, structural document clustering of web pages may have other applications as well.

5 Discussion

This paper has described on-going work in applying techniques and ideas from image-based document layout analysis to the analysis of the structure of HTML documents. By making the structure of the document the primary focus of the analysis, rather than its textual content, we obtain information extraction methods that are both simple to implement and appear to work on a large variety of documents.

There has been some related work on comparing HTML and XML documents. For example, [3] describes *HtmlDiff*, a tool for detecting and highlighting changes in web pages. More recently, [8] describe *X-Diff*, a change detection algorithm that allows for unordered subtrees. However, the application of differencing in this work differs in that it is not used to detect changes over time, but to extract information that varies among different web pages at the same time.

The approach described in this paper also differs substantially from wrapper induction approaches described in the literature [6, 5, 2]. Those approaches attempt to recover, from training data, a representation of a regular expression or grammar that is then capable of extracting desired information from novel pages. Computing and representing wrappers is an algorithmically challenging task. In contrast, the methods presented in this paper avoid computing any kind of wrapper altogether, which results in a much simpler algorithm.

Furthermore, wrapper induction methods have been designed with a separation of model building and application phases, and with the goal of extracting specific information from web pages. In contrast, the approach escribed in this paper is interactive: users collect web pages they are interested in, obtain variables contained in those web pages in tabular form, and select the data they are interested in.

References

[1] A. Coates, H. Baird, and R. Fateman. Pessimal print: A reverse turing test,. In *6th International Conference on Document Analysis and Recognition, Seattle, WA, USA*, pages 1154–1158, 2001.

[2] V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *27th International Conference on Very Large Databases (VLDB 2001)*, 2001.

[3] F. Douglis and T. Ball. Tracking and viewing changes on the web. In *USENIX Annual Technical Conference*, pages 165–176, 1996.

[4] L. Eikvil. Information extraction from world wide web - a survey. Technical Report 945, Norweigan Computing Center, 1999.

[5] N. Kushmerick. Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence*, 118(1-2):15–68, 2000.

[6] I. Muslea, S. Minton, and C. Knoblock. A hierarchical approach to wrapper induction. In O. Etzioni, J. P. Müller, and J. M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 190–197, Seattle, WA, USA, 1999. ACM Press.

[7] W3C DOM Technical Committee. Document object model technical reports. `http://www.w3.org/DOM/DOMTR`, 2003.

[8] Y. Wang, D. DeWitt, and J. Y. Cai. X-diff: An effective change detection algorithm for xml documents. To appear in ICDE 2003; available at `http://www.cs.wisc.edu/ yuanwang/xdiff.html`.