Automatic annotation of data extracted from large web sites
(EXTENDED ABSTRACT)

Luigi Arlotta[1], Valter Crescenzi[1], Giansalvatore Mecca[2], and Paolo Merialdo[1]

[1]Università Roma Tre [2]Università della Basilicata

1 Introduction

The extraordinary amount of data that is currently available on the Web cannot be fully exploited because web pages are intended to be browsed by humans, and not computed over by applications. This drawback has motivated several researchers to study methods and techniques aimed at facilitating the creation of *wrappers*, i.e. web data extraction programs (see [9] for a survey on wrappers).

Some recent proposals, based on machine learning approaches, are able to semi-automatically create wrappers. Since the major effort of the programmer is the production of the training set, several researchers have focused on the development of systems that offer sophisticated user interfaces to assist the programmer in building the training set for the target pages. These systems allow us to quickly build wrappers for pages that contain data we are interested in.

However, since the creation of wrappers as well as their maintenance require a human intervention, the costs of a reliable application dramatically augment with the number of wrapped sources.

In order to reduce the wrapper production and maintenance costs, recent studies have concentrated on the automatic generation of wrappers. In the context of the RoadRunner project, we have developed a system that, based on a grammar inference approach [3, 2], automatically infers a wrapper from a collection of similar pages. Based on different techniques, systems for the same goal have been recently developed also by Arasu and Garcia-Molina [1] and by Wang and Lochovsky [11].

Automatic systems leverage on the observation that data published in the pages of very large sites usually come from a back-end database and are embedded within a common HTML template. Therefore many pages share a common structure, and differences correspond to the data coming from the database. The wrapper generation process aims at inferring a description of the common template, which is then used to extract the embedded data values.

These proposals reduce but do not eliminate the need for a human intervention. Since wrappers are built automatically, the values that they extract are anonymous and a human intervention is still required to associate a meaningful name to each data item. The automatic annotation of data extracted by automatically generated wrappers is a novel problem, and it represents a step towards the automatic extraction and manipulation of web data.

This paper reports our recent researches for automatically annotating data extracted from data-intensive web sites. Our proposal is based on the observation that web pages are designed to be consumed by human users, and therefore they usually contains text strings, i.e. labels, whose goal is to explicate to the final user the intentional meaning of the published data.

We have developed and experimented a working prototype, called LABELLER, that works as a companion system to our wrappers generator. Although LABELLER has been developed in the framework of the ROADRUNNER project, its underlying approach can be applied together with other wrapper generator systems.

The paper is organized as follows: Section 2 presents the problem we address; Section 3 overviews our approach and describes the algorithm we have implemented to annotate anonymous data. Experiments with the prototype system are reported in Section 4. Section 5 discusses limitations and opportunities of our approach. Finally Section 6 presents some related work.

2 Problem Description

We have analyzed about 50 automatically generated wrappers that work on pages from several web sites: a large majority of the data extracted by the wrappers are accompanied with a string representing a meaningful name of the value. This is not surprising: since web pages are intended to be browsed by humans, it is a common practice that the published data are accompanied by textual descriptions to help the user to correctly interpret the underlying information. Therefore, our approach for labelling data extracted from automatically generated wrappers relies on the observation that important information about the semantics of data is often available on the web pages themselves.

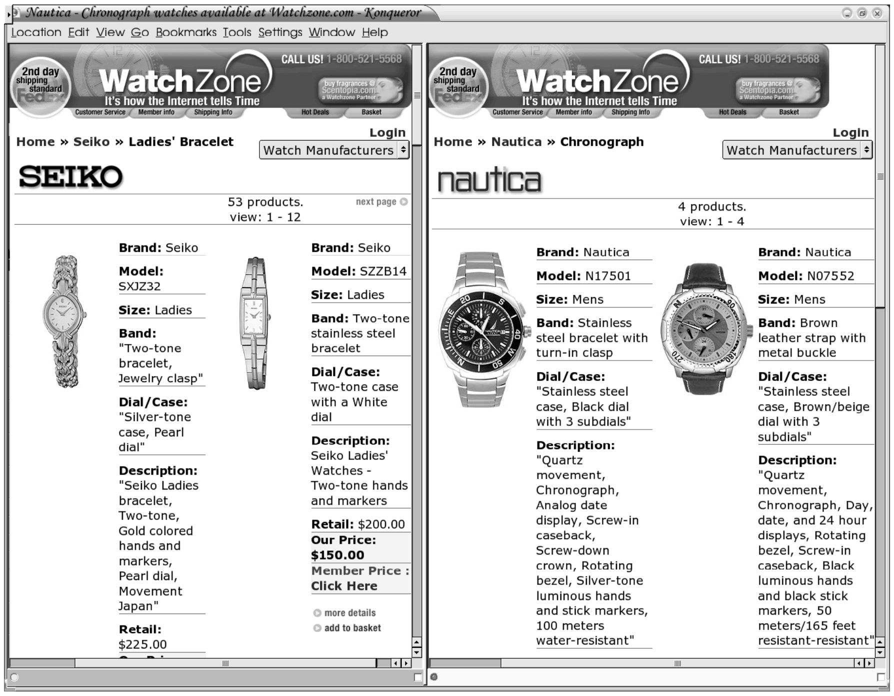
Let us introduce an example to present the overall context and the issues we address. Consider the HTML pages shown in Figure 1. Feeding our wrappers generator system with a set of sample pages like these, we obtain a grammar description of the common template, as follows:[1]

```
<html>...(...<b>Brand:</b>$J<hr><b>Model:</b>$K<hr>...
  <div><b>Our Price:<br>$Q</b></div>...)+...</html>
```

This grammar works as wrapper by considering the non-terminals symbols (`$J`, `$K`, ...) as place-holders: during the parsing of the input pages, they match with the strings to extract. Data extracted from the two pages by this wrapper are shown in Figure 2.

It is worth noting that the extracted data are anonymous; nevertheless, looking at the source pages in Figure 1, we observe that useful labels are available on the pages themselves as part of the common template. For instance, the boldface "`Brand`" compares nearby the extracted value "`Seiko`", "`Model`" is close to "`SXJZ32`", "`SZZB14`", and so on. More interestingly, many of the anonymous

[1] For the sake of presentation, we report a partial and simplified version of the real grammar.


Nautica - Chronograph watches available at Watchzone.com - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
WatchZone
It's how the Internet tells Time
CALL US! 1-800-521-5568
Login
Watch Manufacturers

Home » Seiko » Ladies' Bracelet

SEIKO

53 products.
view: 1 - 12

next page

Brand: Seiko
Model: SXJZ32
Size: Ladies
Band: "Two-tone bracelet, Jewelry clasp"
Dial/Case: "Silver-tone case, Pearl dial"
Description: "Seiko Ladies bracelet, Two-tone, Gold colored hands and markers, Pearl dial, Movement Japan"
Retail: $225.00

Brand: Seiko
Model: SZZB14
Size: Ladies
Band: Two-tone stainless steel bracelet
Dial/Case: Two-tone case with a White dial
Description: Seiko Ladies' Watches - Two-tone hands and markers
Retail: $200.00
Our Price: **$150.00**
Member Price : **Click Here**
more details
add to basket

Home » Nautica » Chronograph

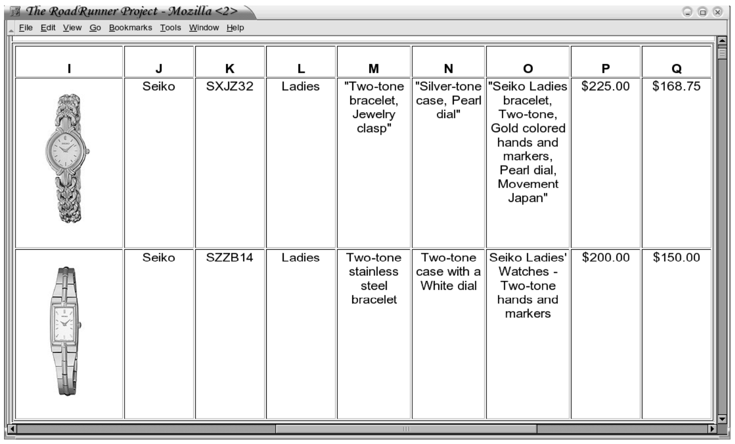
nautica

4 products.
view: 1 - 4

Brand: Nautica
Model: N17501
Size: Mens
Band: Stainless steel bracelet with turn-in clasp
Dial/Case: "Stainless steel case, Black dial with 3 subdials"
Description: "Quartz movement, Chronograph, Analog date display, Screw-in caseback, Screw-down crown, Rotating bezel, Silver-tone luminous hands and stick markers, 100 meters water-resistant"

Brand: Nautica
Model: N07552
Size: Mens
Band: Brown leather strap with metal buckle
Dial/Case: "Stainless steel case, Brown/beige dial with 3 subdials"
Description: "Quartz movement, Chronograph, Day, date, and 24 hour displays, Rotating bezel, Screw-in caseback, Black luminous hands and black stick markers, 50 meters/165 feet resistant-resistant"

Fig. 1. Input HTML Pages from http://www.watch.com


The RoadRunner Project - Mozilla <2>
File Edit View Go Bookmarks Tools Window Help

I	J	K	L	M	N	O	P	Q
	Seiko	SXJZ32	Ladies	"Two-tone bracelet, Jewelry clasp"	"Silver-tone case, Pearl dial"	"Seiko Ladies bracelet, Two-tone, Gold colored hands and markers, Pearl dial, Movement Japan"	$225.00	$168.75
	Seiko	SZZB14	Ladies	Two-tone stainless steel bracelet	Two-tone case with a White dial	Seiko Ladies' Watches - Two-tone hands and markers	$200.00	$150.00

Fig. 2. Data Extraction Output

fields extracted by the wrapper could be named after some terminal symbol of the grammar itself. For instance the terminal symbol `Brand` could be the label for all the values of `$J`.

The goal of LABELLER is that of analyzing the wrapper and a set of sample pages in order to locate inside the common template meaningful labels for the otherwise anonymous data.

We fix the terminology in this settings by calling *variants* the non-terminal symbols and *invariants* the non-tag terminal symbols of the grammar.We call *data-values* the strings that match with variants (e.g. "`Seiko`", "`SXJZ32`", "`SZZB14`") and *labels* those matching with invariants (e.g. "`Brand`", "`Model`"). More precisely, LABELLER's goal is that of associating to each variant a meaningful label chosen among the invariants.

3 The RoadRunner Labeller

Our technique to annotate extracted data is inspired by the following observation: since web pages are designed to be presented on a browser to a human user, usually values and labels are *visually close* to each other.

Therefore, first LABELLER computes the coordinates of the bounding boxes of every data-value and every label in a given sample page (as rendered by a graphical web browser in standard 800×600 resolution). Then it tries to find the optimal association label/data-value by analyzing their spatial relationships.

We have developed several heuristics to establish the correct associations: (*i*) values and labels are close to each other, (*ii*) usually a label is vertically, horizontally, or centrally aligned to its associated values, (*iii*) labels are usually placed either to the left or above values, (*iv*) it is not allowed that either a label or a value is between another value and its label.

Also, we impose the constraint that a variant cannot have more than one label.

Actually, the same variant/invariant could occur many times if the sample pages contain list of items, such as tables. In this cases, we simply consider only the first occurrences of each variant/invariant.

3.1 The Labelling Algorithm

The algorithm implemented by LABELLER takes as input a wrapper and a set of sample pages; as output, it produces a set of label/variant associations for the given wrapper.

Initially it chooses an arbitrary order for the set of labels $L = \{l_i, i = 1 \ldots N_l\}$ and for the set of variants $V = \{v_j, j = 1 \ldots N_v\}$ of the wrapper. Then it calls the function LABELS to compute a set of label/variant associations (l_i/v_j) on each of the given samples.

LABELS tries to exploit the *spatial relationships* between data-values and labels in the graphical rendering of one web page as displayed by an ordinary browser. It computes an $N_l \times N_v$ matrix M of *scores*, that is positive real numbers such that $M[i, j]$ is a measure of the "goodness" of l_i as label for v_j; as smaller are the scores as better are the associations.

Some associations are immediately discarded without further evaluation by setting the corresponding score in M to $+\infty$. Namely, we discard (l_i/v_j) if any of the following conditions holds: (i) l_i is below or to the right of v_j; (ii) the distance between l_i and v_j is greater than D_{max} pixels; (iii) l_i is placed on the diagonal of v_j; (iv) there exists another variant/invariant which is located between l_i and v_j. The first three conditions reflect the fact that a label is usually placed above or to the left of the corresponding data-value, never too far from it. The last condition considers that anything between a data-value and its label would keep the reader from intuitively associating them.

Given a pair (l/v) and the corresponding bounding-boxes on a page s, our score function is defined as follows:

$$\textsc{score}(l, v, s) = \textsc{distance}(l, v, s) \cdot \sin(2 \cdot \textsc{alignment}(l, v, s))$$

This definition captures two aspects: *distance* and *alignment*, because labels are usually close and aligned to the corresponding data-values.

The distance is simply defined as the minimum distance between the two bounding boxes. The ALIGNMENT factor is a measure of the alignment and is computed as the smallest angle amongst several possibilities. Namely, we consider the line crossing at the two midpoints, and all the lines crossing at the two farthest point on the same side (left, right, top, or bottom) of the two bounding-boxes; then we consider all possible (absolute values of) angles encompassed between one coordinate axis and these lines. Note that $\sin(2 \cdot \textsc{alignment}(l, v, s))$ has a minimum when the boxes are perfectly aligned, and a maximum when one is perfectly on the diagonal of the other.

Function LABELS terminates by producing the associations (l_i/v_j) according to a trivial *best first* strategy.

The results of every LABELS invocation are then collected and only the associations which have been established without contradictions in all input samples are considered part of the final output. That is, if the same label has been associated to different variants in different samples, it will not be part of the output.

While it is clearly possible that associations inferred according to the scores are due to random factors, i.e. a data-value and one label are incidentally close each other in one sample, it is unlikely that the two will be rendered so close in many samples if the web-designer did not "plan" this way. The experiments have confirmed this observation: the comparison of the results on several samples reduced the number of false positives.

4 Experiments

Based on the above algorithm, we have developed LABELLER, a working prototype of the labelling system and used it to run a number of experiments on real web sites. We have generated wrappers for 19 collections of pages, each collection containing about 10 pages. We have then used these wrappers to extract data from the input pages. Finally, we have run Labeller to annotate the extracted data with a label extracted from the input pages.

To evaluate the effectiveness of the system we have manually defined and stored in a database the correct associations between data-values and labels. The results produced by Labeller have been compared against this collection.

samples		**wrappers**			**results**			
site	description	nesting	#invariant	#variants	#ok	#error	#fp	#fn
www.fifaworldcup.com	players	1	93	30	22	0	1	0
www.fifaworldcup.com	teams	1	57	31	7	0	1	0
half.ebay.com	used books	2	26	29	5	2	0	1
it.finance.yahoo.com	stock quotes	1	52	26	16	1	1	0
match.com	mate finder	0	50	25	19	0	0	0
postershop.com	posters	1	29	16	6	0	0	0
www.allrecipes.com	recipes	1	42	21	9	0	2	1
www.allrecipes.com	recipe index	1	22	11	2	0	1	1
superstore.ubid.com	products	1	35	15	6	0	2	0
www.watchzone.com	watches	1	12	16	8	0	0	0
www.polo.com	dresses	1	11	11	1	0	0	0
autos.yahoo.com	cars	2	70	58	46	0	2	1
www.hotjobs.com	job listing	1	29	10	4	0	2	0
www.overstock.com	jewelries	1	35	13	3	0	0	0
ncdc.noaa.gov	storms	1	51	11	9	0	1	0
www.fbi.com	most wanted	0	17	18	12	0	0	0
www.vitacost.com	drugs	1	30	23	15	0	0	0
houseandhome.msn.com	house finder	1	169	104	45	0	2	3
cinema.com	movies	1	13	12	10	1	0	1
	totals		843	480	245	4	15	8
	totals (%)				90,07%	1,47%	5,51%	2,94%

Fig. 3. Experimental Results

The table in Figure 3 reports the results of our experiments and contains the following elements:(*i*) *samples:* the url of the site and a short description of the collection of pages considered for the experiment. (*ii*) *Wrapper:* some elements about the inferred wrapper; in particular we report the level of nesting of the inferred schema (*nesting*), the number of variants (*#variants*) and the number of invariants (*#invariants*) of the inferred wrapper. (*iii*) *Results:* the results obtained from the Labeller; namely: number of correctly labelled data attributes (*#ok*); number of errors, i.e. number of variants the system has associated a wrong label with (*#error*); number of false positives (*#fp*), i.e. number of variants that although do not have a label in the source page, they have been associated with a label; number of false negative (*#fn*), i.e. number of variants for which the system was not able to associate a label, although a label was present in the source page.

As it can be seen from the table, for a large majority of variants (around 90%) the system is able to correctly identify the right label, even in the presence of nested structures. The system produces a small number of false positives (around 5%) and a very small number of errors (around 1.5%).

Finally, note that the system correctly associates a label with 51% of the wrapper variants. Here it is important to observe the association variant-label is at the schema level. Most of the labelled variants represent data-values which are repeated in the pages, e.g. data-values associated with variants $J, $K, ... in Figure 2. As a consequence the labels cover a large majority of the extracted data-values (for our set of pages, around 82% of data-values).

5 Limitations and Opportunities

The encouraging results of our experiments show that our technique can be adopted as a starting point for more sophisticated approaches. The capability to label data can sensibly improve the degree of automation of related techniques [11, 10, 4, 5]. Our approach essentially requires only the distinction between variants and invariants and can be easily adapted to different automatic wrapper generation systems [1, 11].

The most restrictive hypothesis of our data annotation approach is rather obvious: we need that textual labels describing the meaning of the extracted fields are present in the page. However, many data are published on web pages leaving their meaning implicit.In fact, often pages do not include explicit labels for those data whose semantics can be clearly understood from the context (e.g. the title of a book in a page describing details about that book). On the other hand, we observe that it is unlikely, especially in a data-intensive web site, that a large number of relevant attributes are left unlabelled. Just to give an example, consider a bookstore site, such as `amazon.com`: usually the authors and the title of books are not explicitly labelled; on the contrary, detailed data, such as ISBN, publisher, prices and many other attributes, are presented with an associated label. More important, labels are usually associated with those data that can assume an ambiguous meaning; following the bookstore example, labels like `"our price"`, `"saving"`, `"list price"` are associated with data dealing with prices: these labels are indispensable to correctly interpret the published information.

We observe that the meaning of data implicitly described by the context can be interpreted by domain ontologies; on the contrary, ontology based approaches cannot work with several ambiguous data, such as the prices of our example. Therefore we believe that it might be interesting to combine our approach with ontology-based annotation systems.

6 Related Work

The problem we address is somehow new, since the development of automatic wrapper generator systems is a recent result.

A related problem is that of identifying instances of specific value domains, such as people names, phone numbers, prices, and so on. Software modules specialized to this issue are the so called *recognizers* [8]. However their usage in the context of web data extraction is limited and the number of domains they can work on is relatively small. Also they cannot associate different labels with data-values of the same domain; for example they can identify several variants representing a price in a page, but they cannot distinguish the different meanings (list price, price and saving) of the variants.

Knoblock *et al.* have addressed problem which is reminiscent to ours in the context of the wrapper maintenance issue [10]. They have developed a wrapper generator system that needs to be trained by a set of labelled samples. Whenever a wrapper fails they have to regenerate the wrapper; to this end their system tries to automatically build a set of labelled examples by using the data extracted before the wrapper failure [7].

Some wrapper generation approaches are based on domain ontologies [6]: they derive the data extraction rules from knowledge about the domain. Therefore, in principle they do not need to face the labelling problem, since the extracted data represent instances of known concepts. The strong dependence on domain is the major limitation of these approaches.

References

1. A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *ACM SIGMOD 2003*, 2003.
2. V. Crescenzi and G. Mecca. On automatic information extraction from large web sites. Technical Report rt-dia-76-2003, Università di Roma "Roma Tre", 2003. `http://web.dia.uniroma3.it/research/2003-76.pdf`.
3. V. Crescenzi, G. Mecca, and P. Merialdo. roadRunner: Towards automatic data extraction from large Web sites. In *International Conf. on Very Large Data Bases (VLDB 2001), Roma, Italy, September 11-14*, 2001.
4. A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD Conference*, 2001.
5. D. W. Embley, D. Jackman, and L. Xu. Multifaceted exploitation of metadata for attribute match discovery in information integration. In *Workshop on Information Integration on the Web*, pages 110–117, 2001.
6. D. W. Embley and J. Y. S. N. Y. Record-boundary discovery in web documents. In *ACM SIGMOD International Conf. on Management of Data*, pages 467–478, 1999.
7. C. Knoblock, K. Lerman, S. Minton, and M. I. Accurately and reliably extracting data from the web: A machine learning approach. *IEEE Data Engineering Bulletin*, 23(4):33–41, 2000.
8. N. Kushmerick. Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence*, 118:15–68, 2000.
9. A. Laender, B. Ribeiro-Neto, A. Da Silva, and T. J. A brief survey of web data extraction tools. *ACM SIGMOD Record*, 31(2), June 2002.
10. I. Muslea, S. Minton, and C. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Journal of Autonomous Agents and Multi-Agent Systems*, 4:93–114, 2001.
11. J. Wang and F. Lochovsky. Data-rich section extraction from html pages. In *Proceedings of the 3rd International Conference on Web Information Systems Engineering (WISE 2002), 12-14 December 2002, Singapore, Proceedings*, pages 313–322. IEEE Computer Society, 2002.