PEWeb: Product Extraction from the Web Based on Entropy Estimation

Hieu Xuan Phan
School of
Information Science
Japan Advanced Institute of
Information and Technology
hieuxuan@jaist.ac.jp

Susumu Horiguchi
School of
Information Science
Japan Advanced Institute of
Information and Technology
hori@jaist.ac.jp

Bao Tu Ho
School of
Knowledge Science
Japan Advanced Institute of
Information and Technology
bao@jaist.ac.jp

Abstract

Mining product descriptions from e-commercial web sites is an important task in information extraction from the Web. In this paper, we propose an efficient technique to do the task. This technique first discovers the set of product descriptions based on the measure of entropy at each node in the HTML tag tree. Afterwards, a set of association rules based on heuristic features is employed to filter the output and therefore enhance the precision. The experimental results of PEWeb system show that the proposed method outperforms existing automatic techniques remarkably.

1. Introduction

With the explosion of the Web, a large amount of information on many different subjects has become available online. A majority part of this huge data repository is formatted in regularly structured objects like lists, tables, records or product descriptions. A *product description* usually contains common fields such as picture, name, manufacturer, price, etc. Some of them may be missing. Automatic extracting product descriptions from the Web is useful for information integration and therefore users can easily locate desired products from a huge number of e-commercial web sites.

Extracting product descriptions from the Web faces several challenging problems, that are how to effectively recognize product regions within input web pages, how to exactly split a product region into separated product descriptions, and how to identify real product descriptions and therefore get rid of noisy objects.

Several existing approaches information extraction such as wrapper induction, NLP-based, ontology-based, and HTML structured-based methods can be employed to extract product descriptions. Wrapper induction tools such as WIEN [10], SoftMealy [9], and STALKER [13] develop wrappers based on domain-specific sample pages to get extraction rules which, in turn, are used extract data objects in future *similar* pages. However, these tools are still time-consuming and labor-intensive due to difficulties in writing and maintaining wrappers as well as annotating sample pages. NLP-based tools such as RAPIER [4], SRV [8], and WHISK [15] usually use traditional NLP techniques, e.g. lexical and part-of-speech tagging, to learn rules for extracting relevant data existing in highly grammatical documents. These tools, of course, are not appropriate for extracting less-grammatical web pages. Ontology-based technique [7] encounters difficulty in writing and maintaining domain ontologies. HTML-aware tools such as W4F [14], XWRAP [12], and RoadRunner [6], rely on inherent structural features of HTML documents for accomplishing data extraction. Although these tools achieve a higher level of automation comparing to those of wrapper induction, they still need user intervention either at building extraction rules or labeling sample pages.

MDR [2], OMINI [3], and IEPAD [5] are automatic and HTML-aware tools. OMINI [3] uses a set of extraction algorithms to locate the smallest subtree that contains all objects of interest. Then, it employs a suite of object extraction algorithms to find the correct object separator tags that can separate objects. IEPAD [5] proposes a method to find patterns from the HTML tag strings, and then use the patterns to extract objects. The method uses a PAT tree to find patterns. However, both OMINI and IEPAD returns many noisy objects because the diversity in separator tags and drawback of PAT tree in locating match of patterns, respectively. MDR [2] employs the *edit distance* string matching algorithm to recognize data regions containing a set of generalized nodes. Afterwards, MDR determines data records in each generalized node by using coarse-grained heuristic observations. The first drawback of MDR is the computational complexity of the edit distance algorithm. The second drawback of MDR is that its course-grained heuristic observations such as dollar sign are not enough for identifying true data records and, therefore, results in many noisy objects.

In this paper, we propose a new and simple but efficient technique to automatically extract product descriptions from web collections with various representation styles. Our method is based on the observation that product descriptions usually have similar display format and they are contained in similar HTML subtrees. To extract these product descriptions, the source HTML page must first be parsed to form

DOM-like HTML tag tree. Then, entropy estimation is performed, by a proposed entropy measurement, at each internal node of the tree to measure the similarity among subtrees. Nodes with high entropy value, i.e. high similarity among its subtrees, should contain potential product descriptions in their descendant nodes. Finally, entropy information is combined with a set of association rules based on heuristic features to identify exact product descriptions. The experimental results of PEWeb system show that our system outperforms MDR dramatically in terms of precision. In addition, the simple representative value mapping allows fast extraction comparing to MDR system that uses edit distance string matching. Computational complexity analysis in Section 3.1 shows this advantage.

The remained part of the paper is organized as follows. Section 2 presents three steps of the proposed technique. Section 3 demonstrates experimental results and some discussions. Finally, Section 4 concludes the paper and states the future work.

2. The Proposed Approach

Our approach includes three main steps: building HTML tag tree, entropy measurement, and product description extraction. They will be in turn presented in the following subsections.

2.1. HTML Tag Tree

HTML documents naturally have hierarchical and nesting structure of tags. However, not all web pages tightly conform to the W3C HTML specification. Thus, input web pages are first preprocessed to fix potential mistakes (e.g. missing or mismatched closing tags, overlapping paired tags or not properly nested tags, etc.). Our system (PEWeb) uses HTML Tidy [17], a tool from W3C, to clean up web pages and convert them into XHTML 1.0 [18]. The tag tree is somewhat similar to the DOM tree [16]. Each node is corresponding to a tag. Each tag node together with its descendants constitutes a subtree. From this point, *tag node*, *tree node*, *node*, and *subtree* will be used interchangeably. Additionally, each node in our HTML tag tree contains following types of information.

tt	*t*ag *t*ype, i.e. single or paired tag
tw	*t*ag *w*eight
tl	*t*ag *l*evel, i.e. the distance from it to the root node
dp	*dep*th of this subtree
rv	*r*epresentative *v*alue for this subtree
er	*e*ntropy *r*atio of the node
sr	*sc*o*r*e of the tree node
st	a list of pointers to its *sub*trees
sp	no. of adjacent subtrees a product *sp*ans

Table 1. Information associated with tag node

2.2. Entropy Measurement

This section mainly describes entropy estimation to identify *product regions* containing potential product descriptions. This idea originates from the observation that similar product descriptions are usually contained in similar subtrees of tags. The term *similar*, in this sense, means that these subtrees have analogical structures in both subtree skeleton and tag position. Figure 1 shows that four product descriptions reside in very similar subtrees D, E, F, and G surrounded by four ovals. The essential problem is to measure the similarity among these subtrees. One of the existing solutions is string matching using *edit distance* algorithm to compare the similarity between two tag strings. However, this algorithm has the computational complexity $O(|s_1||s_2|)$, where $|s_1|$ and $|s_2|$ are lengths of two strings. So the computational time of this approach is high especially when the size of HTML file is large and the number of online e-commercial web sites is really huge.

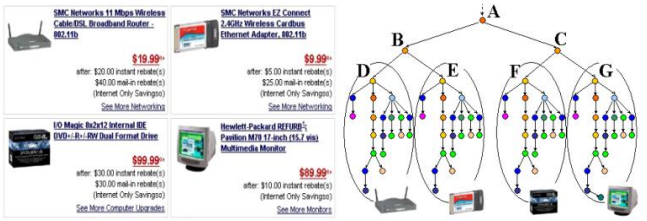

Figure 1. Product descriptions & tag tree

To measure the similarity of subtrees at each tree node, we perform the two following steps. First, structure of each subtree is mapped into a *representative value* that should satisfy three properties: (1) two subtrees have close representative values if they are similar in structure, (2) two trees have divergent representative values if they are dissimilar in structure, and (3) the computational complexity for this mapping is as small as possible. Second, entropy on representative values corresponding to subtrees will be estimated. The higher the entropy is, the larger similarity these subtrees have.

2.2.1 Mapping Representative Value

The *representative value* of a tag tree T (T is also the root node), denoted as $T.rv$, is calculated by the following formula,

$$T.rv = T.tw + \sum_{N_i \in N} (N_i.tl \times N_i.co \times N_i.tw) \qquad (1)$$

where N is the set of all descendant nodes of T. $N_i.tl$ is the tag level, i.e. the distance from tag node N_i to the root node T. $N_i.tw$ is the tag weight of the tree node N_i. The main usage of $N_i.tw$ value is to help distinguish among different HTML tags. $N_i.co$ is the child order of the node N_i among its siblings. Figure 2 shows an example of representative value

calculation. In this figure, each oval contains tag name and tag weight of each tree node. The number associated with tree edge is the child order. The tag level of the root node, i.e. T, starts from 1.

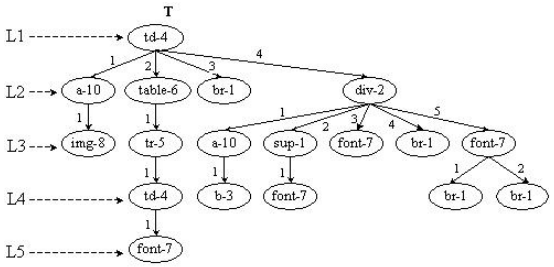

Figure 2. Representative value calculation

The formula (1), in a sense, does not tightly conform the first two properties. As a result, it cannot achieve an exact measurement like *edit distance* algorithm. However, the diversity in HTML representation styles allows us to choose this simple and approximate mapping with small time complexity rather than a sophisticated one because the probability that two dissimilar HTML subtrees have close representative values is small. The computational complexity of the above formula is $O(n)$ where n is number of tag nodes in tree T. Our experimental results show that the *representative value* estimated using the formula (1) is completely acceptable.

2.2.2 Entropy Calculation

Supposing that a tree node T has a set of subtrees $S = \{T_1, T2, \ldots, T_n\}$ with the corresponding set of *representative values* $R = \{T_1.rv, T_2.rv, \ldots, T_n.rv\}$. Let $P = \{T_i | T_i \in S \text{ and } T_i.dp \geq DTh\}$. The Shannon's entropy of T with respect to their children's representative values, denoted as $E(T)$, is calculated as,

$$E(T) = -\sum_{T_i \in P} \frac{T_i.rv}{\sum_{T_i \in P} T_i.rv} \ln \frac{T_i.rv}{\sum_{T_i \in P} T_i.rv} \qquad (2)$$

Our observations show that product descriptions usually reside in subtrees with the depth greater than or equal to minimum depth threshold DTh. Thus, we use this threshold to remove noisy subtrees that containing no real product description.

The $E(T)$ value in the formula (2) is not normalized. This means that $E(T)$ tends to be high when the cardinality of P is growing. Thus, we need to normalize this formula to get convenience for later use. In the above formula, Shannon's entropy get its maximum value $\ln |P|$ when all representative values are equal. Therefore, we get the normalized value of $E(T)$ between 0 and 1, denoted as $T.er$ (*e*ntropy *r*atio) as,

$$T.er = \frac{E(T)}{\ln |P|} \qquad (3)$$

The ERC algorithm for entropy ratio computation in formula (3) for all nodes in the HTML tag tree is described in Algorithm 1. This algorithm uses T and minimum depth threshold DTh as inputs. After calculating, entropy ratio at each tree node will be updated.

Algorithm 1 - ERC: Entropy Ratio Calculation

Require: Node T, DTh.
Ensure: Entropy ratios of all nodes in T are calculated.

```
1:  Entropy = 0; T.er = 0; Count = 0; TotalRV = 0;
2:  for T_i ∈ T.st do
3:    if T_i.dp ≥ DTh then
4:      Count++; TotalRV += T_i.rv;
5:    end if
6:  end for
7:  for T_i ∈ T.st do
8:    if T_i.dp ≥ DTh then
9:      Portion = T_i.rv/TotalRV;
10:     Entropy += -Portion × ln(Portion);
11:     ERC(T_i, DTh);
12:   end if
13: end for
14: if Count ≥ 2 then
15:   T.er = Entropy / ln(Count);
16: end if
```

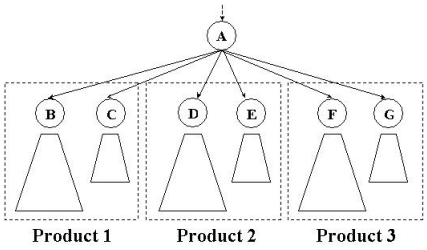

Figure 3. Product of two adjacent subtrees

Sometimes, one product description may span two or three adjacent subtrees as depicted in Figure 3. We easily recognize this case because the entropy ratio of the node A with respect to $R_2 = \{B.rv + C.rv, D.rv + E.rv, F.rv + G.rv\}$ tends to be higher than that of the node A with respect to $R_1 = \{B.rv, C.rv, D.rv, E.rv, F.rv, G.rv\}$. The algorithm ERC should be slightly modified to deal with this situation.

2.3. Product Description Extraction

Nodes with large entropy ratios will have high probability of containing regularly formatted objects. However, not all these objects are actual product descriptions. A large portion of these objects are advertisements, navigation links, etc. These objects, in a sense, are noisy information and we need

to remove them. Obviously, information about entropy ratio is not enough to do this. Thus, a scoring technique based on a set of association rules is combined with entropy ratio to extract and filter the output. Extracted objects having the score greater than or equal to the minimum score threshold (STh) will be considered product descriptions. Other objects are noise and will be ignored.

In our system, a set of association rules [1] is used to give score for each tree node. These association rules are extracted from a database with 19 attributes (i.e. items) and 1000 records. All 19 features or items are listed in Table 2. The first 18 items chosen based on heuristics are useful for classification. For example, $nPrices$ is the number of occurrences of strings like "List Price", "Our Price", "Sale Price", or "Outlet Price" a node contains. The last item (*isProduct*) is the target attribute. The set of 1000 training examples corresponding to 1000 tree nodes were gathered from different e-commercial web sites. This set contains both positive (product descriptions) and negative (noisy objects) training examples.

Feature	Description
er	Entropy ratio of tree node
dp	The depth of the subtree
rv	Representative value of tree node
nLinks	No. of <*a*> tag
nImages	No. of <*img*> tag
nFonts	No. of <*font*> tag
nBolds	No. of <*b*> tag
nItalics	No. of <*i*> tag
nBLs	No. of <*br*> tag
nLists	No. of <*li*> tag
nDollars	No. of currency signs (e.g. $, £)
nPercents	No. of % signs
nDigits	No. of digital characters
nPrices	No. of "List Price", "Our Price"
nSaves	No. of "Saving", "You Save"
nSeeMores	No. of "See More", "Read More"
nRatings	No. of "Rate", "Rating"
nLengths	The length of non-tagged text
isProduct	*True* is product, *False* otherwise

Table 2. Heuristic features for filtering outputs

The antecedent of each rule is a combination or subset of the first 18 items in Table 2. The consequent of each rule is the target attribute, i.e. *isProduct*. In the experiments, minimum support and minimum confidence (strength) are set to 2% and 90%, respectively. The number of discovered association rules is 64. In this output, we pick out the 30 most significant rules that not only have high confident and support factors but also less contradict each other. Some examples of association rules accompanied by their support and confidence are listed in Table 3.

We use this set of association rules to give score for each future tree node. If the consequent of a rule is "isProduct = False" (negative rule) we decrease the score of the node by x points. If the consequent of a rule is "isProduct = True" (positive rule) we increase the score of the node by y. Both x and y depend on the confidence and support of each rule. In the PEWeb system, we set $x = y = support \times confident/100$. The higher the score of a tag node is the larger possibility that tag node is a product description. We use a minimum score threshold, STh, to determine whether a tag tree node is a product description.

Association rule	Sup.	Conf.
$nLinks \leq 1 \wedge nImages \leq 0$ $\wedge\ 0 < nFonts \leq 1 \wedge nBolds$ $\leq 0 \rightarrow isProduct =$ False	2.9%	100%
$2 \leq nLinks \leq 5 \wedge$ $0 < nImages \leq 2 \wedge nBold \leq 3$ $\rightarrow isProduct =$ True	9.3%	98%
$0 < nDollars \leq 3 \wedge$ $0 < nPrices \leq 3 \wedge$ $nIamges \leq 2$ $\rightarrow isProduct =$ True	15.5%	99%

Table 3. Association rule for scoring

Algorithm 2 - PEWeb: Product Extraction from the Web

Require: T, $ERTh$, STh, DTh.
Ensure: Product descriptions in the tree T.

```
1: if T.dp < DTh then
2:    Return;
3: end if
4: if T.er ≥ ERTh then
5:    for T_i ∈ T.st do
6:       if T_i.sr ≥ STh then
7:          Extract T_i;
8:       else
9:          PEWeb(T_i, ERTh, STh, DTh);
10:      end if
11:   end for
12: else
13:   for T_i ∈ T.st do
14:      PEWeb(T_i, ERTh, STh, DTh);
15:   end for
16: end if
```

The PEWeb algorithm uses inputs as T (tag tree node), $ERTh$ (minimum entropy ratio threshold), STh (minimum score threshold), and DTh (minimum depth threshold). The algorithm visits all nodes under T recursively. At each node, PEWeb compares the node's entropy ratio ($T_i.er$) with $ERTh$. If the node's entropy ratio is greater than or equal to the threshold, PEWeb will consider all the node's children. At each child node T_i, if $T_i.sr$ is greater than or equal to the minimum score threshold (STh), T_i is a product description, otherwise PEWeb will be called recursively with T_i to search for potential product descriptions below T_i. If $T.er$ is smaller

than $ERTh$ threshold, PEWeb will also be called recursively to search for potential product descriptions under node T. Sometimes, product description spans two or three consecutive subtrees. In this case, the PEWeb algorithm needs to be modified slightly to extract this kind of product descriptions.

For example, the node A in Figure 1 has entropy ratio that is greater than $ERTh$. PEWeb will consider all children of node A, i.e. B and C. However, B and C have small scores because their values of the first 18 features in Table 2 *tell* that these nodes are unlikely real product descriptions. In other words, their values of features in Table 2 are not consistent with positive association rules. Thus, the PEWeb, in turn, will be called recursively with B and C. At this level, D, E, F, and G will be extracted because their scores are high thanks to large supports from the positive association rules.

3. Evaluation

3.1. Computational Complexity

The time complexity of MDR algorithm [2] is $O(nK)$ without considering the complexity of edit distance algorithm, where n is the average number of child nodes at each internal node and K is the maximal number of nodes that a generalized node contains. Let N be the number of internal nodes in the overall HTML tree, the overall computational complexity (with considering the complexity of edit distance algorithm) of MDR is $O(NnK|s|^2)$, where $|s|$ is the average length of tag strings associated with HTML subtrees.

The overall computational complexity of PEWeb system is the sum of those of representative value mapping algorithm $O(N)$, of entroy ratio calculation (ERC) algorithm $O(Nn)$, and of product extraction from the Web (PEWeb) algorithm $O(N)$. In the above estimations, N and n are the number of internal nodes of HTML tree and the average number of children at each internal nodes, respectively. From the above estimations, we conclude that the overall complexity of PEWeb system is $O(Nn)$. This time complexity is much smaller than that of MDR. Actually, when K is large, e.g. experiment with web page 13 in Table 4, MDR takes long time to complete.

3.2. Experimental Comparative Evaluation

In this section, we evaluate the experimental results of our system (PEWeb) that implements the proposed method. PEWeb was developed with MS Visual C++ and the first version is now available at www.jaist.ac.jp/∼hieuxuan/softwares/peweb/. We also compare the PEWeb with MDR, the state-of-the-art system for mining data records from the Web. MDR can be downloaded from www.cs.uic.edu/∼liub/MDR/MDR-download.html. Both PEWeb and MDR have several options and the experimental results were performed using their default values.

Testing Data: Experimental web pages are e-commercial pages and listed in Table 4. Several pages in the list are from the experiments of MDR [2]. These pages belong to different domains such as book, hardware and software, food, art, toys, medicine, automobile, magazine, cosmetic, etc. These pages are also diverse in representation style. Performing experiment on many pages from a single site is unnecessary because pages belonging to the same site usually have similar structures. Thus, each site has only one or two pages as its representatives.

Point of View of Product Description: MDR [2], OMINI [3], and IEPAD [5] consider data records as regularly formatted objects. This means that the outputs of these system contain both product descriptions and noisy objects (e.g. advertisements, navigation links, form components, etc.). That is why the recall and precision of these system are so high in the corresponding papers. We consider data records as product descriptions and our system tries to eliminate noisy objects as many as possible thanks to its scoring system based on association rules. Our system will follow the later point of view, i.e. only product descriptions are considered data records, other kinds of objects will be considered as noise.

Parameter Setting: Both PEWeb and MDR have several options. MDR has *similarity* threshold and *dollar sign* options. The default value of the similarity threshold is 60% and this is also the recommended value. The second option of MDR is the *dollar sign*. If users check the dollar sign option, the outputs only have data records containing dollar sign. The default value of this option is *false* (i.e. not checked). PEWeb's options include *minimum depth threshold* (DTh), *minimum entropy ration threshold* ($ERTh$), and *minimum score threshold* (STh). The default values of these options are 3, 0.90, and 15 respectively. In the experiments, we use the default option values for both PEWeb and MDR except that the *dollar sign* option of MDR will be tested for both of its values.

Experimental Results: The experimental results are shown in Table 4. In this table, the first column is the page number, the second column is the URL of page (with some details omitted), the third column is the number of product descriptions available in the page, the next two columns are the number of found product descriptions and the number of correct product descriptions of PEWeb, the next two columns are the number of found product descriptions and the number of correct product descriptions of MDR (the value in parentheses is experimental value with *dollar sign* option checked). In the last two lines of the table, we sum the number of product descriptions, the number of found product descriptions, and the number of correct product descriptions. Then, the standard measures *recall* and *precision* are computed for the overall 35 web pages. Figure 4 depicts the recall comparison among PEWeb, MDR, and MDR with *dollar sign* option checked. The graph is drawn based on recall value of each web page in Table 4. Similarly, precision comparison is shown in Figure 5.

Discussion: In Table 4 we see that both recall and precision of PEWeb (97%, 96%) are higher than those of MDR (84%, 60%). That the precision of PEWeb is lager than that of MDR is reasonable because the outputs of MDR include noisy records. The recall of MDR should be greater than that of PEWeb because measuring similarity using *edit distance*

algorithm should be more exact than measuring similarity using entropy of representative values. However, the experimental results in Table 4 is opposite. This situation can be explained that MDR system can not fix all the potential errors of HTML code in input pages, so there are some product descriptions residing in web pages are can not be reached. When MDR is run with option *dollar sign* checked, the recall (74%) of MDR decreases and the precision (88%) of MDR increase. The decline of recall is because there are several product descriptions that do not contain the dollar sign ($). They contain no currency sign or other kinds of currency (e.g. £) rather than dollar sign. Besides, the increase of precision can be explained that many noisy objects containing no dollar sign are eliminated. However, this precision can not attain the precision of PEWeb because there are some objects containing dollar sign but they are not the true product descriptions. The comparison of recall and precision in Figure 4 and Figure 5 gives more detailed information for each single page testing.

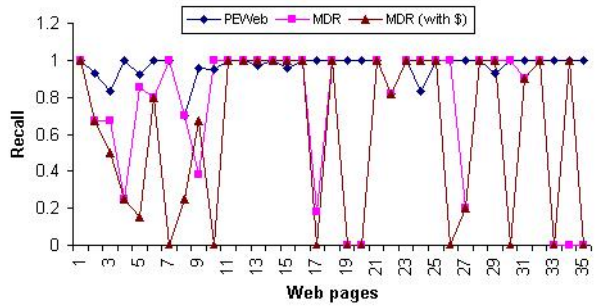

Figure 4. Recall comparison

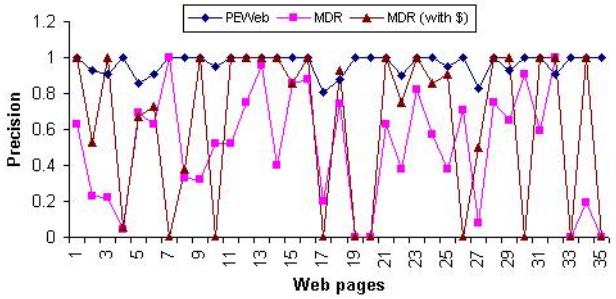

Figure 5. Precision comparison

4. Conclusion and Future Work

In this paper, we proposed a novel and efficient method to automatically extract product descriptions from the Web. Our technique uses entropy estimation and association rule-based classification. Measuring similarity among subtree structures is very efficient because it is easy to implement and has a small computational complexity. Classifying outputs based on association rules of important heuristic features helps PEWeb system effectively eliminates noisy objects. The experimental results show that PEWeb outperforms the state-of-the-art mining system significantly.

The product description generated by PEWeb is a chunk of HTML code and text. In the future, we will labeling for data fields in product descriptions such as name, price, manufacturer, category, etc. Some models suitable for this task are Hidden Markov Models, Maximum Entropy Markov Models, and Conditional Random Fields. Additionally, the next version of PEWeb can automatically detect e-commercial web sites for extraction process.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. *Proceedings of the ACM SIGMOD*, pages 207–216, 1993.

[2] L. Bing, G. Robert, and Z. Yanhong. Mining data records in web pages. *Proceedings of ACM SIGKDD*, 2003.

[3] D. Buttler, L. Liu, and C. Pu. A fully automated extraction system for the world wide web. *Proceedings of IEEE ICDCS-21*, 2001.

[4] M. Califf and R. J. Mooney. Relational learning for pattern-match rules for information extraction. *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 328–334, 1999.

[5] C.-H. Chang and S.-L. Lui. Iepad: Information extraction based on pattern discovery. *Proceedings of WWW-10*, 2001.

[6] V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. *Proceedings of the 26th VLDB*, pages 109–118, 2001.

[7] D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, Y.-K. Ng, D. Quass, and R. D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data and Knowledge Engineering*, 31(3):227–251, November 1999.

[8] D. Freitag. Machine learning for information extraction in informal domains. *Machine Learning*, 39(2-3):169–202, 2000.

[9] C.-N. Hsu and M.-T. Dung. Generating finite-state transducers for semi-structured data extraction from the web. *Information Systems*, 23(8):521–538, 1998.

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

[11] A. H. F. Laender, B. A. Ribeiro-Neto, and A. S. Da Silva. Debye - data extraction by example. *Data and Knowledge Engineering*, 40:121–154, 2002.

[12] L. Liu, C. Pu, and W. Han. Xwrap: An xml-enable wrapper construction system for web information sources. *Proceedings of the 16th IEEE International Conference on Data Engineering*, pages 611–621, 2000.

[13] I. Muslea, S. Minton, and C. A. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent*, 4(1-2):93–114, 2001.

[14] A. Sahuguet and F. Azavant. Building intelligent web applications using lightweight wrappers. *Data and Knowledge Engineering*, 36(3):283–272, 2001.

[15] S. Soderlan. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.

[16] W3C. The document object model. *www.w3.org/DOM/*.

[17] W3C. Html tidy. *www.w3.org/People/Raggett/Tidy/*.

[18] W3C. Xhtml 1.0. *www.w3.org/TR/xhtml1/*.

No.	Web site (some details of URLs are omitted)	Product (Record)	PEWeb		MDR	
			F.	C.	F.(with $)	C.(with $)
1	books.shopping.yahoo.com/ (books)	15	15	15	24 (15)	15 (15)
2	shopping.yahoo.com/ (MP3 players)	15	15	14	43 (19)	10 (10)
3	auctions.yahoo.com/ (general)	12	11	10	36 (6)	8 (6)
4	www.overstock.com/ (general)	8	8	8	50 (31)	2 (2)
5	shop.lycos.com/ (general)	13	14	12	16 (3)	11 (2)
6	art.listings.ebay.com/ (digital arts)	51	56	51	65 (56)	41 (41)
7	software.thepricesearch.com/ (softwares)	10	10	10	10 (0)	10 (0)
8	www.amazon.com/(general)	20	14	14	43 (13)	14 (5)
9	www.amazon.com/ (DVD players)	24	23	23	28 (16)	9 (16)
10	www.bookscanada.ca/ (books)	22	22	21	42 (0)	22 (0)
11	www.kidsfootlocker.com/ (kids foot)	12	12	12	23 (12)	12 (12)
12	chemstore.cambridgesoft.com/ (softwares)	6	6	6	8 (6)	6 (6)
13	chemstore.cambridgesoft.com/ (softwares)	221	219	218	230 (220)	220 (220)
14	www.compusa.com/ (general)	8	8	8	20 (8)	8 (8)
15	www.compusa.com/products/ (CD drives)	24	23	23	28 (28)	24 (24)
16	qualityinks.com/ (discount softwares)	21	21	21	24 (21)	21 (21)
17	www.ubid.com/ (general)	17	21	17	15 (0)	3 (0)
18	www.ubid.com/ (monitors & projectors)	70	80	70	95 (75)	70 (70)
19	www.nothingbutsoftware.com/ (general)	13	13	13	4 (2)	0 (0)
20	www.radioshack.com/ (general)	3	3	3	6 (0)	0 (0)
21	www.radioshack.com/ (digital cameras)	10	10	10	16 (10)	10 (10)
22	www.softwareoutlet.com/ (softwares)	11	10	9	24 (12)	9 (9)
23	www.sephora.com/ (cosmetics)	9	9	9	11 (9)	9 (9)
24	www.etoys.com/etoys/ (electronic toys)	12	10	10	21 (14)	12 (12)
25	www.etoys.com/ ($10 - $20)	20	21	20	52 (22)	20 (20)
26	eat.epicurious.com/ (general)	5	5	5	7 (0)	5 (0)
27	www.drugstore.com/ (new drug products)	10	12	10	24 (4)	2 (2)
28	www.apple.com/buy/ (Mac softwares)	6	6	6	8 (6)	6 (6)
29	www.nextag.com/ (flowers & plants)	15	15	14	23 (15)	15 (15)
30	search.kelkoo.co.uk/(automobiles)	20	20	20	22 (0)	20 (0)
31	www.onsale.com/ (general)	21	21	21	32 (19)	19 (19)
32	www.1-kitchen-store.com/electric-cookware/	10	11	10	10 (10)	10 (10)
33	www.wiredseek.com/shop/ (CDs, DVDs)	31	31	31	4 (4)	0 (0)
34	www.target.com/ (general)	6	6	6	32 (6)	6 (6)
35	www.magazinesofamerica.com/ (magazines)	6	6	6	2 (2)	0 (0)
Total		**777**	**787**	**756**	**1098 (655)**	**649 (576)**
Recall (Rc) & Precision (Pr)			**Rc: 97%** **Pr: 96%**		**Rc: 84% (74%)** **Pr: 60% (88%)**	

Table 4. Recall and Precision of PEWeb and MDR