World Wide Web (2007) 10:157–179
DOI 10.1007/s11280-007-0021-1

Information Extraction from Web Pages Using Presentation Regularities and Domain Knowledge

Srinivas Vadrevu · Fatih Gelgi · Hasan Davulcu

Received: 2 May 2006 / Revised: 19 September 2006 /
Accepted: 8 January 2007 / Published online: 2 March 2007

Abstract World Wide Web is transforming itself into the largest information resource making the process of information extraction (IE) from Web an important and challenging problem. In this paper, we present an automated IE system that is domain independent and that can automatically transform a given Web page into a semi-structured hierarchical document using presentation regularities. The resulting documents are weakly annotated in the sense that they might contain many incorrect annotations and missing labels. We also describe how to improve the quality of *weakly annotated data* by using domain knowledge in terms of a statistical domain model. We demonstrate that such system can recover from ambiguities in the presentation and boost the overall accuracy of a base information extractor by up to 20%. Our experimental evaluations with TAP data, computer science department Web sites, and RoadRunner document sets indicate that our algorithms can scale up to very large data sets.

Keywords information extraction · web · page segmentation · grammar induction · pattern mining · semantic partitioner · metadata · domain knowledge · statistical domain model

S. Vadrevu (✉) · F. Gelgi · H. Davulcu
Department of Computer Science and Engineering, Arizona State University,
Tempe, AZ 85287, USA
e-mail: svadrevu@asu.edu

F. Gelgi
e-mail: fagelgi@asu.edu

H. Davulcu
e-mail: hdavulcu@asu.edu

1 Introduction

The vast amount of data on the World Wide Web poses many challenges in devising effective methodologies to search, access and integrate information. Even though the current Web represents a large collection of heterogeneous sources, their data presentation and domain specific metadata adheres to certain regularities.

In this paper we present automated algorithms for gathering metadata and their instances from collections of domain specific and attribute rich Web pages by using the presentation regularities that are present within the Web pages and by utilizing domain knowledge. Our system works without the requirements that (a) the Web pages need to share a similar presentation template or (b) that they need to share the same set of metadata among each other. Hence, we cannot readily use previously developed wrapper induction techniques [4, 16, 18] which require that the item pages should be template driven or the ontology driven extraction techniques [8, 10, 11] which require that an ontology of concepts, relationships and their value types is provided apriori in order to find matching information.

Our system proceeds in three phases. In the first phase, we use the presentation regularities that are present in the Web pages to organize the content in a hierarchical XML-like structure and annotate the labels in the Web page with their semantic roles. In the second phase, we build a statistical domain model from the hierarchical content structures, exploiting the domain knowledge. In the third phase, we utilize the statistical domain model to improve the semantic role annotations of the labels within each of the Web pages.

Unlike plain text documents, Web pages organize and present their content using hierarchies of HTML structures [23]. Different logical blocks of content, such as taxonomies, navigation aids, headers and footers as well as various other information blocks such as indexes, are usually presented using different HTML structures. Furthermore, whenever an information block contains a list of items, these items themselves are presented consecutively and regularly using repeating similar HTML substructures. In this paper, we present an information extraction (IE) system, called *Semantic Partitioner* that uses these *presentation regularities* to transform a given Web page into a semi-structured document, with the labels in the Web page annotated with their *semantic roles*. However the labels may be *weakly annotated* as the automated IE system is prone to make certain mistakes due to the ambiguity in presentation.

In addition to the presentation regularities present in the Web, there are also regularities in the metadata that can be exposed to refine the structured information extracted from the automated systems. Sometimes the presentation regularities alone are not sufficient to automatically structure the content in a Web page and we need to utilize the *domain knowledge* in order to organize the content. To obtain such domain knowledge we combine the weakly annotated data extracted from individual Web pages by automated IE systems into a statistical domain model. The performance of IE systems can be enhanced by the use of such domain knowledge. The extracted knowledge is probabilistic because the extracted data may be unreliable and it may change depending upon the context. In this paper, we present how to extract such domain knowledge in terms of a statistical domain model from the weakly annotated data and how to utilize the domain knowledge to improve the performance of automated IE systems.

The key contributions and innovations of our system can be summarized as follows:

- A domain independent IE system that organizes the content in a Web page into a *weakly annotated* semi-structured document using presentation regularities.
- A statistical domain model that is built from the *weakly annotated* data obtained from an automated IE system.
- A domain model based IE system that uses domain knowledge and presentation regularities to further improve the annotations made by the statistical domain model.

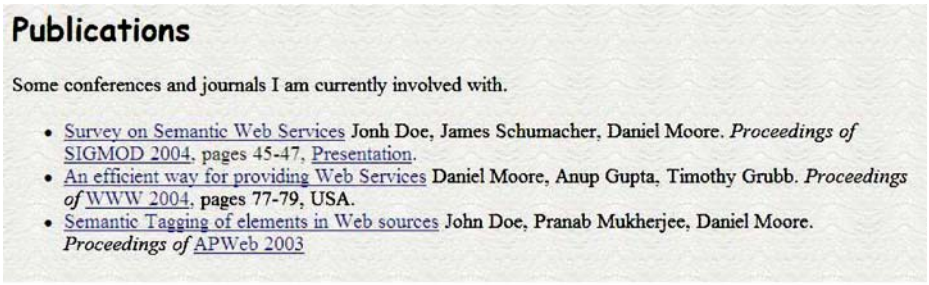
We currently do not align the extracted metadata or instance information with any available ontology or knowledge structure. Various existing approaches developed for mapping and merging ontologies [20] can be utilized for this purpose. We also currently do not process plain text inside the Web pages, i.e., any text fragments that contain modal verbs. However, we hypothesize that the metadata extracted from the attribute rich segments of the Web pages can be used to extract information from text with the help of natural language processing techniques [15].

The rest of the paper is organized as follows. Section 2 presents the background material about information extraction and tagging Web pages. Section 3 presents the Semantic Partitioner system that performs IE on Web pages using the presentation regularities. In the following Section 4, we explain how to utilize the hierarchical structures obtained by Semantic Partitioner and build a statistical domain model. Next, we describe how to improve the semantic role annotations by using the domain model in Section 5. Section 6 presents complexity analysis of our algorithms and Section 7 briefly discusses the related work. We provide the experimental results with various data sets in Section 8 and conclude the paper in Section 9 with some future directions of our work.

2 Background on information extraction

Information extraction is the process of extracting pertinent or relevant information from text, databases, semi-structured and multimedia documents in a structured format. An information extraction system performs this task either automatically or with human intervention by using techniques from various areas like machine learning, data mining, pattern mining, and grammar induction. Since, the World Wide Web is transforming itself into the largest information source ever available, the process of information extraction from the Web is both challenging and interesting problem, which is the primary focus of this paper.

An example of an information extraction task is shown in Figure 1. The example shows the fragment of a faculty home page (Figure 1a), and the structured information extracted from this Web page (Figure 1b). The Web page contains three publications, with each publication having at most five attributes: title, authors, conference, address, and presentation. These attributes can also be referred to as *metadata* as they describe the data. The Web page does not contain the metadata labels themselves, but it contains values from these metadata attributes. In addition,



Publications

Some conferences and journals I am currently involved with.

- Survey on Semantic Web Services Jonh Doe, James Schumacher, Daniel Moore. *Proceedings of* SIGMOD 2004, pages 45-47, Presentation.
- An efficient way for providing Web Services Daniel Moore, Anup Gupta, Timothy Grubb. *Proceedings of* WWW 2004, pages 77-79, USA.
- Semantic Tagging of elements in Web sources John Doe, Pranab Mukherjee, Daniel Moore. *Proceedings of* APWeb 2003

(a) A fragment of a faculty Web page showing his publications.

Title	Authors	Conference	Pages	Address	Presentation
Survey on Semantic Web Services	John Doe, James Schumacher, Daniel Moore	Proceedings of SIGMOD 2004	pages 45—47		Presentation
An efficient way for providing Web Services	Daniel Moore, Anup Gupta, Timothy Grubb	Proceedings of WWW 2004	pages 77—79	USA	
Semantic Tagging of elements in Web sources	John Doe, Pranab Mukherjee, Daniel Moore	Proceedings of APWeb 2003			

(b) The structured information extracted from the Web page.

Figure 1 Example of an information extraction task.

each of the publication record may not contain values for all the five attributes and some of the values may be missing. The goal of the information extraction system in this task is to be able to extract the data associated with all their metadata labels as shown in Figure 1b.

3 Information extraction using presentation regularities

In this section we discuss the semantic partitioning algorithm that works by exploiting the presentation regularities in Web pages. Our semantic partitioning algorithm can detect repeating HTML substructures and organize the content into a *hierarchical content structure* consisting of *groups* of *instances*, while skipping HTML blocks that do not constitute a group. The semantic partitioning algorithm requires no training and works automatically on each Web page.

The semantic partitioning algorithm works in four phases: page segmentation, grouping, promotion and semantic role annotation.

3.1 Page segmentation

A Web page usually contains several pieces of information [5] and it is necessary to partition a Web page into several segments (or information blocks) before organizing the content into hierarchical groups. In this section we describe our page segmentation algorithm that partitions a given Web page into flat segments.

The page segmentation algorithm relies on the DOM tree representation of the HTML Web page and traverses it in a top–down fashion in order to segment the content of the page, which lies at the leaf nodes. In the HTML DOM tree of a Web page, the formatting information about a label can be revealed from its root-to-leaf path in the DOM tree. For example, a root-to-leaf path html.body.table.td.b.a of a label l reveals that l is bold and has a link. We utilize this information to identify the presentation of a label and distinguish the presentation formatting of one label with that of another label in the Web page.

We define a segment as a contiguous set of leaf nodes within a Web page. The algorithm aims to find *homogeneous segments*, where the presentation of the content within each segment is uniform. The algorithm employs a split-traverse based approach that treats all uniform set of nodes as homogeneous segments and further splits non-uniform nodes into smaller segments until each segment is homogeneous. The algorithm relies on the concept of *entropy* in order to determine whether a segment is homogeneous.

We define the notion of entropy of a node in a DOM tree in terms of the uncertainty in the root-to-leaf paths under the node. Our algorithm is based on the observation that a well organized or homogeneous system will have low entropy. We use this principle while traversing the DOM tree from root to leaf nodes in a breadth-first fashion, and split the nodes until each and every segment in the DOM tree is homogeneous.

Entropy is generally used to measure the uncertainty in the system. Hence if any random variable has low entropy, then there is less uncertainty in predicting the possible values that the random variable can take. In our case, we view each node as a random variable in terms of the root-to-leaf paths P_is under the node. We define a set of nodes in the DOM tree to be homogeneous if the paths in the random variable are uniformly distributed, and it is easy to predict the next path within the set.

Definition 1 The path entropy $H_P(N)$ of a node N in the DOM tree can be defined as

$$H_P(N) = -\sum_{i}^{k} p(i) \log p(i),$$

where $p(i)$ is the probability of path P_i appearing under the node N.

We use the concept of path entropy to partition the DOM tree into segments. The algorithm to partition a given Web page into segments is given in Algorithm 1. This algorithm is initialized with a vector of nodes containing just the root node of the DOM tree of the Web page. The *MedianEntropy* is calculated as the median of the path entropies of all the nodes in the DOM tree. Essentially we assume the nodes whose path entropy is less than the median entropy of all the nodes in the DOM tree to be homogeneous and output it as a pure segment. We use median entropy because it acts as a representative sample for an average value in the set. If a node is not homogeneous, then we traverse the children of the nodes in order to find the homogeneous segments. Our page segmentation algorithm is able to identify four segments in the Faculty home page as shown in Figure 2a. Please note that we currently ignore any text fragments in the Web page that contains modal verbs as they add to the noise in identifying the patterns.

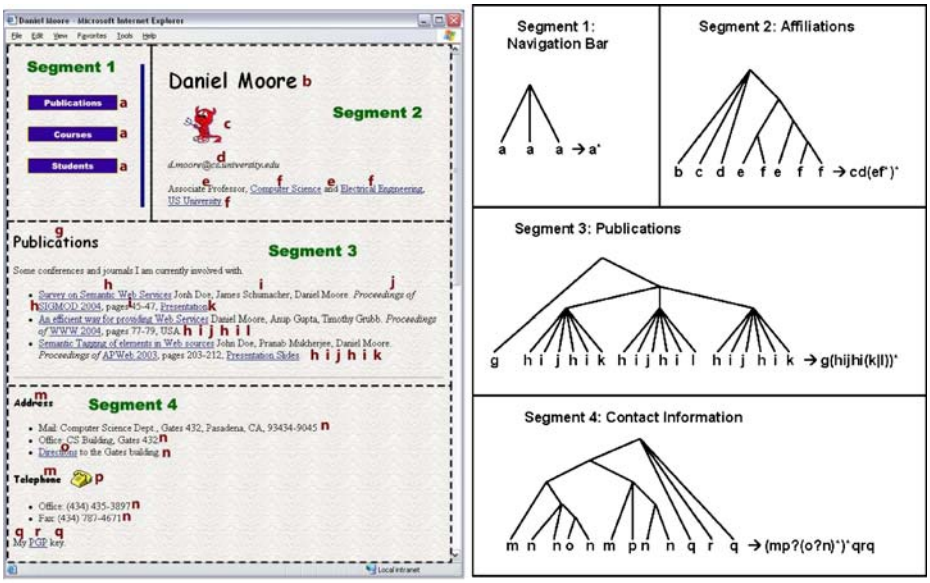
Algorithm 1 Page Segmentation Algorithm

PageSegmenter
Input: Nodes[], a node in the DOM tree
Output: A set of segments

```
1: for Each Subset S of Nodes[] do
2:    H_P(S) := Average Path Entropy of all nodes in S
3:    if H_P(S) ≤ MedianEntropy then
4:       Output all the leaf nodes under N as a new segment PS
5:    else
6:       PageSegmenter(Children(S))
7:    end if
8: end for
```

3.2 Inferring group hierarchy

Even though the page segmentation algorithm organizes the content in a Web page in terms of segments that present the content in a uniform fashion, each segment just contains a flat representation of the labels. In many scenarios, the content is usually represented as a hierarchy of labels and we need to infer the hierarchy among the labels in order to properly depict the content structure. We define a *group* to be a



(a) Web page with Segments Marked (b) Group Tree for each Page Segment

Figure 2 **a** Shows an example of a faculty home page. The labels in the page are marked with their corresponding path identifier symbols. The page segments in the web page are also marked as segments 1 to 4. **b** Shows the sequence of path identifiers, the regular expression inferred from it, and the corresponding group tree for each segment.

contiguous collection of instances that are presented together in a Web page, where an *instance* is a repeating element of a group. Such a group hierarchy is helpful in organizing the content and determining the most general concept in the page and its attributes.

One possible way to achieve such hierarchical organization of the content is to work directly on the DOM tree in a top–down fashion [6]. But such approaches suffer from handling the noise in the leaf nodes and in successfully detecting the boundaries as look-ahead searching is expensive. An efficient alternative approach is to utilize the presentation information (embedded in their root-to-leaf tag path in the DOM tree) of the labels in order to infer a hierarchy among them. We transform each segment into a sequence of path identifiers of root-to-leaf paths of the leaf nodes and infer regular expressions from this sequence in order to extract patterns from them. For example, the path sequence corresponding to the Segment 3 as marked in Figure 2b is *ghijhikhijhilhijhik*.

After the segment is transformed into a sequence of path identifiers, we extract patterns from it by inferring a *path regular expression* from the sequence of path identifiers. We use the standard regular language operators:

- *Kleene star (*):* The kleene star operator conveys the repeating presentation patterns within the Web page. For example, if the Web page presents the names of the courses that a faculty member teaches, they are usually presented with similar presentation template and the kleene star operator would identify this pattern.
- *Optional (?) and Union (|):* The optional and the union operators allow the patterns to accommodate noise.
- *Concatenation (.):* The concatenation operator allows a contiguous sequence to be formed from its sub-patterns.

We build the path regular expression for a given sequence by incrementally building it using bottom-up parsing. We go through the sequence several times, each time folding it using one of the four operators, until there are no more patterns left. The algorithm for building the path regular expression from a sequence is described in Algorithm 2. The *InferRegEx* method is initialized for each segment in the page with its corresponding path sequence.

Once the path regular expression is inferred from the sequence of path identifiers, every kleene star in the path regular expression is treated as a group and its members are treated as instances of the group. For example, in Segment 3 of the Web page in Figure 2b, the path regular expression $(hijhi(k|l))*$ is identified as a group which corresponds to the publications of the corresponding faculty member. The nested kleene star symbols are transformed into nested group hierarchies from the path sequence as shown in Figure 2b.

3.3 Promotion

After grouping, the content of the Web page is organized into hierarchical group structures, but each of these group structures do not have any label. The labels for the groups, which we call as *group headers*, are important as they play a role in connecting the repeating set of instances with the corresponding concept or attribute label. For example, in a news Web page such as CNN, a group structure containing all

Algorithm 2 Inferring Path Regular Expressions

InferRegEx(S)
Input: S, a sequence of symbols
Output: S, a new regex sequence of symbols

```
1: patternFound = true;
2: repeat
3:    patternFound = false;
4:    for len = 1:length(S)/2 do
5:       for i = 0:length(S) do
6:          currPattern := subseq(j, j+i);
7:          if ExtendPattern(currPattern, S, j+i+1) = true then
8:             ReplacePattern(currPattern, S);
9:             patternFound = true;
10:         end if
11:      end for
12:   end for
13: until patternFound = true
14: return S;
```

End of InferRegEx

ExtendPattern(P, S, startIndex)
Input: P, current pattern; S, a sequence of symbols; startIndex, the start index to look patterns for
Output: boolean, indicating whether the pattern is extended

```
1: for i = startIndex:length(S) do
2:    consensusString := IsMatch(currPattern, subseq(i,i+length(currPattern));
3:    if consensusString ≠ null then
4:       currPattern := consensusString;
5:       ExtendPattern(currPattern, S, i+length(currPattern))
6:    end if
7: end for
```

End of ExtendPattern

IsMatch(P_1, P_2)
Input: P_1, first pattern; P_2, second pattern
Output: consensusString obtained by alining P_1 and P_2 or *null*

```
1: if EditDistance(P1, P2) ≤ MaxLength(P1,P2)/3 then
2:    return Consensus(P1, P2)
3: else
4:    return null
5: end if
```

End of IsMatch

ReplacePattern(P, S)
Input: P, current pattern; S, a sequence of symbols
Output: none

```
1: replace all occurrences of the pattern P in the sequence S by a new symbol P'.
```

End of ReplacePattern

the scientific articles cannot be of much use unless it is labeled as *Science* or *Sci/Tech* for indexing and searching purposes.

Bootstrapping the Promotion: In the grouping phase, all the leaf nodes that appear before the group are identified as candidate group headers and the goal of the promotion algorithm is to select the appropriate group header from these candidates for the group. These group headers are bootstrapped by promoting the label as the header whenever there is only one candidate for a particular group.

Frequent Label based Promotion: Whenever similar Web pages from the same domain are available, we identify all the frequent labels in the domain from these similar pages and promote the closest frequent label that is present in the candidate headers of a group as the label for the group.

Naive Bayes based Promotion: When many similar Web pages obtained from similar domains and from the same context are available, the candidate group headers can be used as a training data when deciding on a label to promote as a group header for a group. In such scenarios, we use the words in the instances as features to train a Naive Bayes classifier and compute the likelihood of every candidate group header with a set of instances. Later we promote the closest one as the header for the group.

Path Consistent Annotation: Typically similar presentation templates are used to represent similar labels within the same Web page. Hence if one of those similarly presented labels is promoted with the above rules, then we promote all the other labels within the same Web page with the same presentation template on top of the next groups, whenever applicable.

Figure 3 shows the final hierarchical content structure after promotion for the Web page shown in Figure 2. It can be noticed that the labels 'Daniel Moore,'

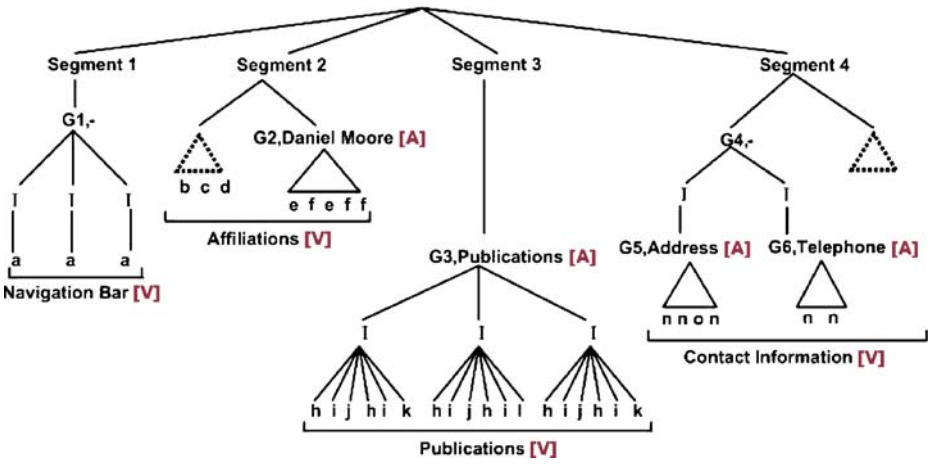


Figure 3 The complete hierarchical structure of the web page shown in Figure 1 after promotion. The group structures G1 through G6 with their respective promoted labels are shown in each page segment. The parts of the Web page that do not correspond to any group structure are shown with *dotted triangles*. The figure also illustrates meta semantic role annotation by annotating the labels with their corresponding semantic roles (C – concepts, A – attributes, V – values).

'Publications,' 'Address,' and 'Phone Number' are promoted over the adjacent groups $G2$, $G3$, $G5$, and $G6$ as group headers. The groups $G1$ and $G4$ do not have any group headers as they did not have any candidate labels to promote over them.

3.4 Annotation with semantic roles

After the group hierarchy has been found and the appropriate labels have been promoted on the groups, we have a powerful content structure that organizes the labels in the Web page in a uniform fashion. We utilize this content structure to annotate the labels in the Web page with metadata tags.

Our annotation framework involves the following four semantic roles:

- *Concept (C)*: A concept defines an abstract or symbolic category or a class of similar items. For example, 'Faculty' and 'Computer Science Department' are some of the concepts in the academic domain.
- *Attribute (A)*: An attribute is a key property of an object or a concept or the name of the relationship. For example, 'publications,' 'address' and 'telephone' are some of the attributes of the 'Faculty' concept in Figure 2a.
- *Value (V)*: A value is a string that provides the value information for an attribute of a certain object or a concept. For example, '(424) 435-3897' and '(434) 787-4671' are the values of the attributes 'Office Telephone' and 'Fax Telephone' attributes in Figure 2a.
- *Noise (N)*: A label that does not belong to any of the above semantic role is assigned to be noise. For example, some of the labels in headers, footers or navigational aids could be annotated as noise.

Figure 3 shows an example of the assignment of tags for the Web page shown in Figure 2a. Within the context of a certain concept, we interpret all the group headers as attribute labels and the all the instance values as value labels of the corresponding attribute. Please note that this assignment of roles may not always be robust; for example, the label 'Daniel Moore' has been tagged as an attribute, but it is actually the name of the object that belongs the concept 'Faculty.' Therefore we call this data as weakly annotated because some of the annotations may be incorrect and some labels may not be annotated at all.

3.5 Discussion on weakly annotated data

Typically presentation regularities are sufficient to organize the content of an HTML Web page and extract information from its data rich segments. However, when the presentation in a Web page does not correlate with the semantic organization of its content, then the performance of extraction systems [3, 9, 19, 22] detoriates.

Figure 4 shows a typical shopping Web page where the product categories and featured list of products are presented. Even though the presentation of the labels is mostly regular, there are some noisy labels, the ones that are circled, that makes it difficult to skip and extract the repeating item structure. Some of these presentation irregularities are categorized as follows:

- *Presentation Inconsistency:* The HTML formatting does not correlate with the logical organization of the page content. For example, in the shopping Web page

Figure 4 A sample products Web page that highlights presentation irregularities. Presentation outliers, labels that do not adhere to the adjacent formatting, and Insertions/Deletions, labels that only present in some of instances in a similarly presented group, are *circled*.

shown in Figure 4, the labels such as '$199.99' and '$299.99' are presented with a bold emphasized formatting that does not match their value semantic roles.

– *Metadata/Data Ambiguity:* Labels with distinct semantic roles are presented with identical formatting. For example in the left hand part of the Web page in Figure 4, the category names such as 'Action,' 'Comedy,' 'Drama' under the section 'DVDs' and the labels such as '-more' are presented using identical formatting and there is no way to distinguish them with presentation information alone.
– *Presentation Outliers:* This occurs when a group of similarly presented items contains some irregularly presented items. For example, in Figure 4, the circled labels under the 'BROWSE BY CATEGORY' taxonomy represent outlier concept labels with irregular presentation.
– *Insertions/Deletions:* Within a group of similar items, some label types are optional. For example in the 'FEATURED PRODUCTS' section of the Web page in Figure 4, the labels 'Last Day to Save!!' or 'Qualifies for FREE SHIPPING' are optional labels.

These presentation irregularities may lead to misinterpretations of the content by IE systems, thus producing weakly annotated data. Such weakly annotated data can be corrected by using the domain knowledge. In the next section, we explain how to

incorporate the domain knowledge into an IE system by using a statistical domain model.

4 Extracting domain knowledge

In this section, we model the domain knowledge from the hierarchical structures given above. The crucial idea is to identify the degree of relationship between the labels and construct a *relational graph* which is easy to generate probability distributions of the roles for each label in the domain. Such a graph would capture the global statistics of the labels and their associations within a domain. We first describe the statistical domain model and then discuss how to obtain the probability distributions.

4.1 Statistical domain model

Before we proceed to the details about our algorithms, we define the notation that we use in the rest of the paper as follows:

- The *ontological roles* $\mathcal{R}$ is the set of *Concept*, *Attribute*, *Values* or *Noise*. Formally,

$$\mathcal{R} = \{C, A, V, N\}.$$

- A *term* is a pair $\langle l, r \rangle$ composed of a label l and a role $r \in \mathcal{R}$. In other words, terms are tagged labels in the Web pages. Each label in a Web page is assumed to be tagged with only one of the given ontological roles above.
- In this setting, we consider all the labels in each Web page are tagged with roles, hence we define a *Web page* to be a vector of its terms. Formally, assuming m labels in the Web page $\mathcal{W}$;

$$\mathcal{W} = \{\langle l_1, r_1 \rangle, \langle l_2, r_2 \rangle, \ldots, \langle l_m, r_m \rangle\}.$$

- The *statistical domain model* $\mathcal{G}$ is a weighted undirected graph where the nodes are terms in the domain, and the weights on the edges represent the association strength between terms.
- The *semantic role distribution* $\mathcal{D}_l$ of a label l is a probability distribution of the four roles $\{P_c, P_a, P_v, P_n\}$, where P_c, P_a, P_v, P_n are the probabilities of l being a concept, attribute, value and noise, respectively within a certain context – which represents the Web document of the label. That is, the role distribution of a label might vary in different Web pages.
- In our framework, the *context* of a label $l \in \mathcal{L}$ in a Web page $\mathcal{W}$ is the Web page $\mathcal{W}$ itself.

Statistical domain model is a relational graph $\mathcal{G}$ generated from automatically extracted data. The nodes in $\mathcal{G}$ denote the labels with their semantic roles and the edges denote the association strengths between the annotated labels. Node weights are initialized as the counts of the corresponding terms and the edge weights are the counts of the corresponding edges in the hierarchies. Formally, assuming w_{ij} as the weight between the terms i and j, and w_i as the weight of the node i, $w_{ij} = w_{ji} = |i \leftrightarrow j|$ and $w_i = |i|$ where $|i \leftrightarrow j|$ and $|i|$ denote the number of times the edge (i, j)

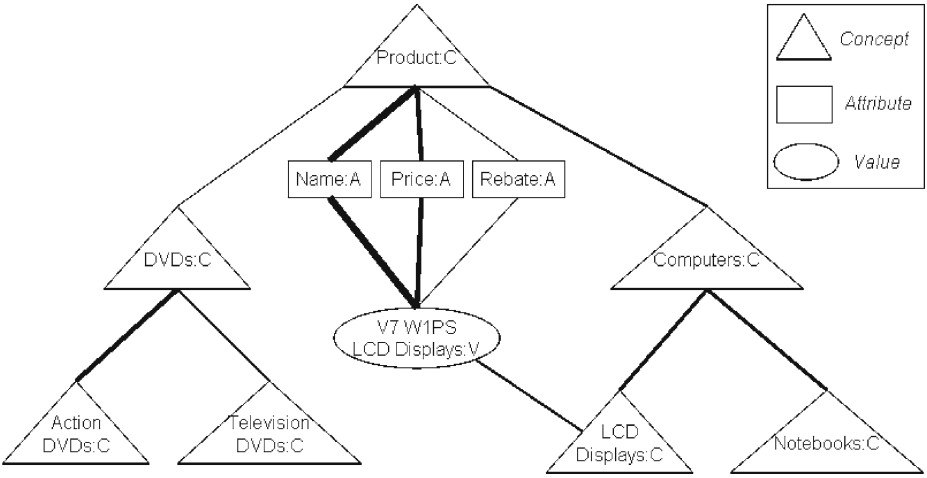


Figure 5 A fragment of the relational graph obtained by aggregating several products Web pages, that is relevant to the page shown in Figure 4. The thickness of the *line* represents the strength of the association between two labels. Each label is also annotated with its semantic role.

appeared in the hierarchies and label i appeared in the entire domain respectively. Note that the edges are undirected since association strength between labels is a bi-directional measure.

A fragment of the statistical domain model obtained for the shopping domain from a collection of Web pages is presented in Figure 5. The nodes are the tagged labels such as 'Product:C' which specifies the concept node of the 'Product' label. The thicker the edge, the stronger the relationship is. For instance, the value 'V7 W1PS LCD Display' is more related with the attribute 'Price' than the attribute 'Rebates' in the graph.

4.2 Role distribution calculation

The probability calculations are briefly introduced below.

Definition 2 For a given Web page $\mathcal{W}$, the probability of a label l tagged with a role $r \in \mathcal{R}$ is $P_r(l|\mathcal{W})$.

In order to reduce the complexity and to utilize the context, the probabilities are calculated using a simple Bayesian model with the following assumptions:

Assumption 1 *All the terms in $\mathcal{G}$ are independent from each other but the given term $\langle l, r \rangle$.*

Assumption 2 *The prior probabilities of all the roles of a label l are uniform.*

Here, the first assumption is the well-known "naive" assumption of the simple Bayesian models. Note that we only utilize the first order relationships of a term in its context, i.e., neighbors of the term in $\mathcal{G}$. One can easily extend the model

for higher order relationships. Assumption 2 states that the role distribution of a label shouldn't depend on its frequency but only its context. Otherwise when a label which is frequent in the domain, the role probability distribution will be strongly biased towards its frequent role in the domain, and will dominate the contextual information. Now, given the two assumptions above we can state the following theorem:

Theorem 1 *Let* $\mathcal{W} = \{t_1, t_2, \ldots, t_m\}$. *Then the normalized probability of a label* l *tagged with the role* r *is,*

$$P_r(l|\mathcal{W}) = \frac{\prod_{i=1}^{m} P_r(l|t_i)}{\sum_{k \in R} \prod_{i=1}^{m} P_k(l|t_i)}. \tag{1}$$

Proof By Bayes's rule,

$$P_r(l|\mathcal{W}) = P_r(l|t_1, t_2, \ldots, t_m) = \frac{P_r(t_1, t_2, \ldots, t_m|l)\, P_r(l)}{P(t_1, t_2, \ldots, t_m)}.$$

Using the independence assumption,

$$= \frac{\prod_{i=1}^{m} P_r(t_i|l)\, P_r(l)}{\prod_{i=1}^{m} P(t_i)}$$

Again using Bayes's rule,

$$= \frac{\prod_{i=1}^{m} P_r(l|t_i)\, P(t_i)}{P_r(l)^m} \cdot \frac{P_r(l)}{\prod_{i=1}^{m} P(t_i)} = \frac{\prod_{i=1}^{m} P_r(l|t_i)}{P_r(l)^{m-1}}$$

This is the unnormalized probability. Since $P_r(l)^{m-1}$ is constant by Assumption 2, we can remove it and add the normalization factor $\sum_{k \in R} \prod_{i=1}^{m} P_k(l|t_i)$ in the denominator. That is,

$$P_r(l|\mathcal{W}) = \frac{\prod_{i=1}^{m} P_r(l|t_i)}{\sum_{k \in R} \prod_{i=1}^{m} P_k(l|t_i)}.$$

□

A conditional probability such as $P_r(l|t_i)$ depends on the association strength between the terms $\langle l, r\rangle$ and t_i in the relational graph $\mathcal{G}$. That is, $P_r(l|t_i) = \frac{P_r(l,t_i)}{P(t_i)} = \frac{w_{\langle l,r\rangle t_i}}{w_{t_i}}$ by Bayes's rule where $w_{\langle l,r\rangle t_i}$ is the weight of the edge $(\langle l, r\rangle, t_i)$ and w_{t_i} is the weight of the node t_i. Our probability model is based on the methodology of association rules [1]. Hence, the initialization for the above conditional probabilities is defined analogous to $P_r(l|t_i) \equiv Confidence(t_i \rightarrow \langle l, r\rangle)$ [2]. This formulation is consistent with Assumption 2 since it is independent from the prior, $P_r(l)$. For more details, interested reader can refer to the technical report [13].

The domain knowledge for a given Web page is represented in terms of semantic role distributions $\mathcal{D}_l$ for each label l in the Web page. P_r in the role distributions of a label l in a Web page is the shorthand notation for $P_r(l|\mathcal{W})$. As the output, these distributions are presented to the IE system.

5 Integrating the domain knowledge into IE system

The key phases in the IE system presented in Section 3 are *grouping* and *promotion*. In this section we describe how to accommodate the domain knowledge into these phases.

5.1 Grouping

To accommodate the domain knowledge in the IE system, the grouping algorithm described in Section 3.2 is extended to include the role distributions of the labels. The only change in the new grouping algorithm occurs in the similarity measure between two candidate sequences of labels. In Algorithm 2, the similarity measure between two sequences of labels is computed using the minimum edit distance between their path regular expressions. In the new grouping algorithm, we add a new similarity measure that takes the similarity of the role assignments of the label sequences into account when computing the similarity measure.

Let l_1 and l_2 be two label sequences. Let S_p be the path similarity measure between l_1 and l_2 computed using the edit distance between their path regular expressions. The role similarity measure S_r between l_1 and l_2 is computed by calculating Pearson Correlation Coefficient [21] between the role distributions $\mathcal{D}_{l1}$ and $\mathcal{D}_{l2}$ of the two labels. The role similarity measure ensures agreement in the role assignment of labels in the sequences. The total similarity measure between l_1 and l_2 is computed by a weighted sum of S_p and S_r. Therefore two label sequences are said to be similar if and only if there is an agreement in both their path regular expressions and their semantic role assignments of individual labels.

The rest of the grouping algorithm proceeds in a similar fashion as described in Section 3.2 in that a regular expression is inferred from the sequence of labels and each Kleene star is interpreted as a group and the members of Kleene star are interpreted as instances.

5.2 Promotion with semantic roles

The promotion algorithm is also slightly altered from the one described in Section 3.3 to bias towards metadata labels in choosing the headers for the group. The *group headers* are obtained by identifying the closest metadata label, with role assignment of a concept or an attribute, as identified from the role distributions. The *group header* is chosen by the following simple formula:

$$GroupHeader(G) = \arg\max_{l_i} \{d_i * (P_a + P_c)\}. \tag{2}$$

where l_i is the label and d_i is the distance of l_i from the group structure.

We also promote metadata labels inside the instances of the groups over the next sequence of values with similar path identifiers. This promotion within the instances assists in extracting the attribute-value relationships that are present within the instances of group structures.

5.3 Significance of domain knowledge

Figure 6a shows a fragment of a course Web page. The labels in the page are annotated with the HTML root-to-leaf path identifiers that highlight the presentation

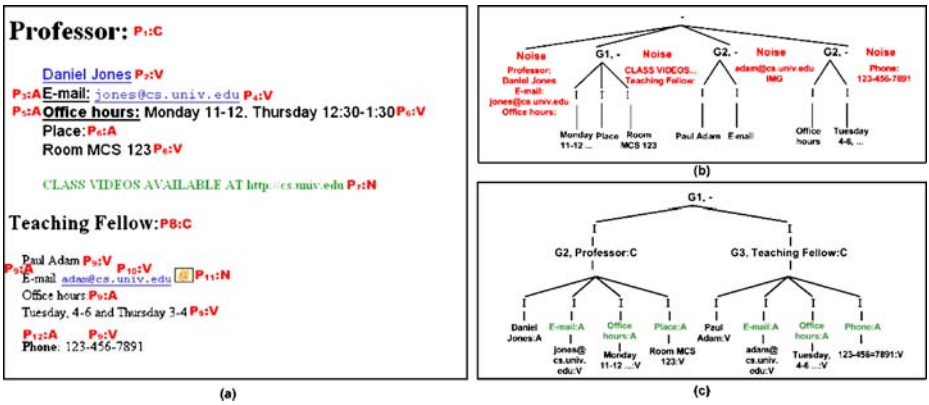


Figure 6 **a** Shows a fragment of a course Web page. For each label in the fragment, its path identifier symbol and its semantic role are marked. **b** Shows the group structures extracted by semantic partitioner algorithm that relies solely on presentation regularities. **c** Shows the group structures extracted by using a domain model.

formatting and their semantic roles. An automated IE system such as semantic partitioning algorithm described in Section 3 that relies solely on the presentation would not be able to distinguish the 'Place' attribute of the 'Professor' concept and 'Office hours' attribute of the 'Teaching Fellow' concept from their respective values 'Room MCS 123' and 'Tuesday, 4–6 ...' since their presentation formatting is identical. Also such a system cannot associate the attributes 'E-mail' and 'Office hours' with their corresponding concepts 'Professor' and 'Teaching Fellow'. Similarly, it would fail to associate these attributes with their respective values, due to various presentation irregularities that would hide the repetitive sub-structures and their boundaries. Figure 6b shows the output of the semantic partitioning algorithm that relies solely on the presentation.

If the system is aware of the domain knowledge, it can more accurately separate metadata from data, and discover the relationships between them. For example, in Figure 6, if the system is aware of the fact that the labels 'E-mail,' 'Office hours,' 'Place' and 'Phone' are attributes, and the labels 'jones@cs.univ.edu,' 'Monday 11–12 ...,' 'Room MCS 123,' 'adam@cs.univ.edu,' 'Tuesday, 4–6 ...,' and '123-456-7891' are values, it can discover correct groupings for $G2$ and $G3$ for the concepts 'Professor' and 'Teaching Fellow.' The accurate group structures obtained by using such a domain knowledge as well as presentation regularities is shown in Figure 6c.

6 Complexity analysis

The page segmentation algorithm works directly on the DOM tree of the Web page in a top-down fashion and its complexity is $O(n \lg n)$, where n is the total number of nodes in the DOM tree. The group hierarchy inference phase iteratively goes through the path sequence in a bottom-up fashion until no more regular expressions are found. Assuming there are k nodes in the segment, the worst complexity of this phase is $O(k^3)$. Since k is considerably smaller than the number of leaf-nodes in

the Web page, this is reasonable for the Web. The promotion and labeling phases are linear in the number of nodes in the corresponding group. The complexity of generating the role distributions from the statistical model is $O(m + p)$ where m and p are the total number of labels and Web pages in the collection respectively [13].

7 Related work

In this section, we discuss the related work from several areas and show how our system is different from them.

Template based algorithms: RoadRunner [9] works with a pair of documents from a collection of template generated Web pages to infer a grammar for the collection using union-free regular expressions. ExAlg [3] is another system that can extract data from template generated Web pages. ExAlg uses equivalence classes (sets of items that occur with the same frequency in every page) to build the template for the pages by recursively constructing the page template starting from the root equivalence class. TAP [14] is a system that extracts RDF triplets from template driven Web sites in order to generate a huge knowledge base that has a Web searchable interface. These algorithms are based on the assumption that the input Web pages are template driven in their presentation and are typically driven by standard metadata. Our approach differs from all these approaches in that it does not require that the input Web pages are template driven and it can effectively handle noise.

Grammar induction based algorithms: Grammar induction based systems employ a strong bias on the type and expected presentation of items within Web pages to extract instances. XTRACT [12] is such a system that can automatically extract Document Type Descriptors (DTDs) from a set of XML documents. It transforms each XML document to a sequence of identifiers and infers a common regular expression that serves as a DTD, using the Minimum Description Length (MDL) principle. Our pattern mining algorithm is different from these approaches and parses the given sequence in a bottom-up fashion and infers the grammar on-the-fly as it goes through the sequence multiple number of times.

Page Segmentation algorithms: VIPS algorithm [5] is a vision-based page segmentation algorithm that is based on HTML heuristics relying on specific tags such as font size and `<HR>` to partition the page into information blocks. Our page segmentation algorithm is similar to the VIPS algorithm in traversing the DOM tree in top-down fashion, but our algorithm uses well-defined information theoretic methods in order to measure the homogeneity of the segment, whereas the VIPS algorithm is based on HTML heuristics.

KnowItAll [11] and C-PANKOW [7] systems extract facts from a collection of Web pages starting with a seed set of factual patterns that are either manually specified or (semi)automatically engineered. The Semtag and Seeker system [10] uses the domain ontologies extracted from automatic wrappers [16] to annotate Web pages. Such systems extract relationships and facts that match these patterns from the natural language text segments of the Web pages. Hence, they are complementary to our systems capabilities that processes the data rich segments of Web pages.

8 Experimental results

In this section, we describe the data we used in our experiments and provide the results and discussions for our experiments.

8.1 Experimental setup

We used three different data sets in order to evaluate the efficacy of our algorithms. In the first two data sets, we show how our algorithms work with template-driven and non-template-driven Web pages. In the third data set, we compare our approach with another IE system, RoadRunner [9].

The first data set consists of TAP KB,[1] containing the categories AirportCodes, CIA, FasMilitary, GreatBuildings, IMDB, MissileThreat, RCDB, TowerRecords and WHO. These categories alone comprise 9,068 individual attribute-rich Web pages. We provide experimental results for this data set with our algorithms and compare them against the relations obtained by TAP.

As our second data set, we prepared CSEDepts data set which is composed of individual Web pages from Faculty and Courses domains, consisting of 125 Web sites and more than 15,000 individual Web pages. To demonstrate the performance of our semantic role annotation algorithms, we created a smaller data set containing randomly chosen 120 Web pages from each of the *faculty* and *course* categories. We provide experimental results for this data set with our algorithms.

As the third data set, we selected the RoadRunner[2] [9] data. We compare our extraction of data values with the RoadRunner system in this experiment.

The experimental results are obtained by comparing the data annotations of the algorithms to manually annotated data by eight human volunteers who are non-project member computer science graduate students. The inter-human agreement on manual annotation was 87%, which indicates that the data annotations can be ambiguous and can be interpreted differently in various contexts.

8.2 Experiments with the TAP data set

The Table 1 shows the experimental results for the TAP data set using semantic partitioning algorithm that relies solely on presentation regulariteis. The algorithm achieves 100% F-Measure with annotating the labels with the concept label. Since the TAP data set contains only one concept per page, the algorithm is able to easily identify the label. However, the algorithm suffers from low recall with annotating as attribute labels because some of the attribute labels are single valued and there is no group associated with them, and they are labeled as values. Nevertheless, the algorithm is able to tag with the attribute labels correctly whenever it does, as the precision is above 91%. As expected, the recall and precision numbers for the value label annotation are exactly opposite for the same reasons that many attribute labels are labeled as values.

[1] TAP Home page is located at http://tap.stanford.edu.

[2] RoadRunner experimental results can be found at http://www.dia.uniroma3.it/db/roadRunner/experiments.html.

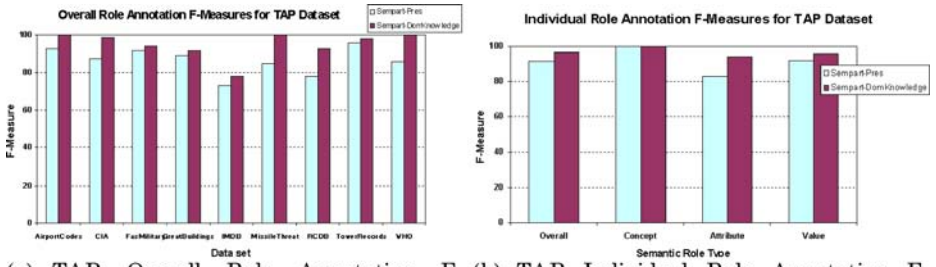
Table 1 Experimental results with TAP data set using semantic partitioner algorithm.

DomainName	P(C)	R(C)	F(C)	P(A)	R(A)	F(A)	P(V)	R(V)	F(V)	Avg F-Measure
AirportCodes	100	100	100	94	83	88	86	98	92	93
CIA	100	100	100	96	54	69	88	99	92	87
FAS Military	100	100	100	96	76	85	84	99	91	92
Great buildings	100	100	100	95	61	74	88	99	93	89
Missile threat	100	100	100	96	63	76	66	99	79	85
IMDB	100	100	100	72	56	63	63	51	56	73
Roller coster Database (RCDB)	100	100	100	78	52	46	84	91	88	78
Tower records	100	100	100	75	69	72	62	55	58	77
WHO	100	100	100	100	94	97	85	100	92	96
Overall	100	100	100	91	67	74	79	92	85	86

P(C), R(C), F(C) denote the precision, recall and F-measure of the concept annotation. Similar notation is used for attributes (A), and values (V).

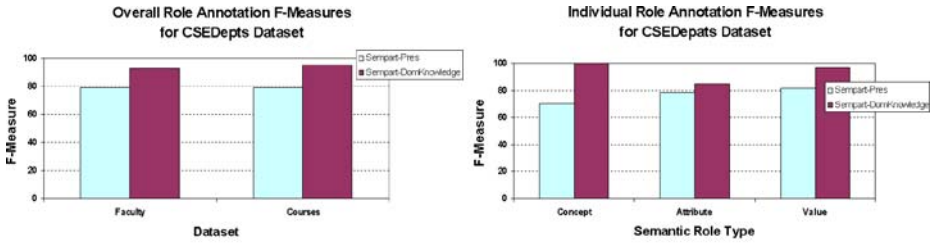
For the TAP data set, the statistical domain model and corresponding semantic role distributions for each Web page are generated from automatic wrapper based IE systems. The extracted domain knowledge is fed to our algorithm that extracts the relational facts and also improves the semantic role annotations of labels. We compare the performances of the semantic partitioning algorithm with and without utilizing the statistical domain model.

The experimental results for semantic role annotations are shown in Figure 7. The graphs show the F-Measure values of concept labeling, attribute labeling, value labeling and overall labeling. The semantic partitioning algorithm that relies solely on presentation regularities and does not utilize the statistical domain model was not able to identify the attribute names correctly in some cases because the presentation does not distinguish the attribute name and its values. But the algorithm that makes use of the statistical domain model in terms of semantic role distributions overcomes such irregularities in presentation. As it can be seen from the results, the accuracies for some categories in the TAP data set are lower than others. The low F-Measure value occurs when the statistical domain model cannot recover from unreliable information extracted from automated systems.



(a) TAP Overall Role Annotation F-Measures (b) TAP Individual Role Annotation F-Measures

Figure 7 The role annotation F-measure values from the TAP data set for semantic partitioning algorithm, with and without using the domain knowledge.



(a) CSE Overall Role Annotation F-Measures (b) CSE Individual Role Annotation F-Measures

Figure 8 The role annotation F-measure values from CSEDepts data set for semantic partitioning algorithm, with and without using the domain knowledge.

8.3 Experiments with the CSEDepts data set

Figure 8 shows and compares the F-measure values for semantic partitioning algorithm with and without utilizing the domain knowledge. It shows that the semantic partitioning algorithm based on presentation regularities alone achieves about 90% F-measure, and the semantic partitioning algorithm that uses the statistical domain model achieves more than 95% F-measure. The F-measure for the concepts in the initial semantic partitioning algorithm is low because the number of concept labels in the domain are very few and they are ambiguously located along with the other attribute labels. However, the statistical domain model is helpful in disambiguating these concept values and boost the F-measure. The algorithm is able to perform fairly accurately in annotating the attribute and value labels in both cases, which is very vital because, these labels are the most frequent on the Web. The F-measure for value annotation is the highest among all role annotations, as the semantic partitioner is able to correctly identify the groups and thus identify the instances of the concept and attribute labels accurately.

8.4 Experiments with RoadRunner data set

Table 2 shows the number of Web pages in each of the ten categories from which the IE systems, RoadRunner and semantic partitioner systems, with and without utilizing the statistical domain model were able to extract the relevant information. By exposing the regularities in the extraction patterns, the semantic partitioner system with the statistical domain model was able to extract relevant information from 8 out of 10 categories, whereas the original system was only able to extract data from 6 categories. The resulting semantic partitioner system with the statistical domain model was also able to perform better than the RoadRunner system in one category (package directory) as the RoadRunner system requires two Web pages to infer the presentation template. The 'uefa' data is organized in terms of complex tables, RoadRunner was able to infer the template by using two sample pages whereas the semantic partitioner (both initial and modified) was unable to extract from such tables using a single page. The results show that the performance of the semantic partitioner with the statistical domain model is comparable to that of RoadRunner system. The statistical domain model for this data set is learnt from over

Table 2 Comparison of the performance the RoadRunner algorithm with semantic partitioner system, before and after utilizing the statistical domain model.

Site	Classes			Comparative results		
	Description	No. of pages	Metadata	RoadRunner	Sempart (before)	Sempart (after)
amazon.com	Cars by brand	21	Yes	21	–	21
amazon.com	Music bestsellers by style	20	No	–	–	–
buy.com	Product information	10	Yes	10	10	10
buy.com	Product subcategories	20	Yes	20	20	20
rpmfind.net	Packages by name	30	No	10	10	10
rpmfind.net	Packages by distribution	20	No	20	20	20
rpmfind.net	Single package	18	No	18	18	18
rpmfind.net	Package directory	20	No	–	20	20
uefa.com	Clubs by country	20	No	20	–	20
uefa.com	Players in the national team	20	No	20	–	–

100 shopping and sports Web sites. The precision and recall of metadata extraction from the RoadRunner data set are 89 and 94%. Please note that our system is able to achieve this performance without the requirement of template-drivenness unlike RoadRunner system.

9 Conclusions and future work

In this paper, we have presented an IE system, Semantic Partitioner, that can automatically structure the content of a Web page into a semi-structured hierarchical document, with the labels in the Web page *weakly annotated* with their semantic roles. We also presented details about how to build a statistical domain model that can be utilized as domain knowledge to enhance the performance of an automated IE system. We demonstrate that automated IE coupled with an automatically constructed domain model can recover from ambiguities in the presentation and improve the accuracy of the base information extractor. Our experimental evaluations with TAP, computer science department Web sites, and RoadRunner data sets indicate that our algorithms can scale up to large data sets in terms of running time complexity. Hence, we conclude that current automated IE systems can benefit from using the domain regularities in order to extract information from all kinds of data rich Web documents. Such structured information also enables browsing of information in order to see interesting connections among a set of objects.

In our future work, we intend to identify the missing attribute labels of values in a given Web page by using aggregated statistical domain model which may obtain the labels from different Web pages. This research also involves merging of labels that refer to the same entities which is closely related with the area of schema matching and merging. Another dimension is to extend our work to perform information

extraction from text blobs of the Web pages with the help of natural language processing techniques.

Acknowledgements This work was partially supported by the Office of Naval Research (ONR) under its Multidisciplinary Research Program of the University Research Initiative (MURI) under Grant No. N00014-04-1-0723.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: ACM SIGMOD Conference on Management of Data, pp. 207–216. Washington, D.C. (1993)
2. Alpaydin, E.: Introduction to Machine Learning, chapter 3, pp. 39–59. MIT Press, Cambridge, MA (2004)
3. Arasu, A., Garcia-Molina, H.: Extracting structured data from web pages. In: ACM SIGMOD Conference on Management of Data, San Diego, USA (2003)
4. Ashish, N., Knoblock, C.A.: Semi-automatic wrapper generation for internet information sources. In: Conference on Cooperative Information Systems, pp. 160–169 (1997)
5. Cai, D., Yu, S., Wen, J.-R., Ma, W.-Y.: Vips: a vision-based page segmentation algorithm. Technical Report MSR-TR-2003-79, Microsoft Technical Report (2003)
6. Chkrabarti, S.: Integrating the document object model with hyperlinks for enhanced topic distillation and information extraction. In: International World Wide Web (WWW) Conference (2001)
7. Cimiano, P., Ladwig, G., Staab, S.: Gimme' the context: context-driven automatic semantic annotation with c-pankow. In: The 14th International World Wide Web (WWW) Conference (2005)
8. Ciravegna, F., Chapman, S., Dingli, A., Wilks, Y.: Learning to harvest information for the semantic web. In: Proceedings of the 1st European Semantic Web Symposium, Heraklion, Greece (2004)
9. Crescenzi, V., Mecca, G.: Automatic information extraction from large web sites. J. Artists' Choice Mus. **51**(5), 731–779 (2004)
10. Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., McCurley, K.S., Rajagopalan, S., Tomkins, A., Tomlin, J.A., Zien, J.Y.: A case for automated large-scale semantic annotation. Journal of Web Semantics **1**(1), 115–132 (2003)
11. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in knowitall. In: International World Wide Web (WWW) Conference (2004)
12. Garofalakis, M., Gionis, A., Rastogi, R., Seshadri, S., Shim, K.: XTRACT: a system for extracting document type descriptors from xml documents. In: ACM SIGMOD Conference on Management of Data (2000)
13. Gelgi, F., Vadrevu, S., Davulcu, H.: Automatic extraction of relational models from the web data. Technical Report ASU-CSE-TR-06-009, Arizona State University, April (2006)
14. Guha, R., McCool, R.: TAP: a semantic web toolkit. Semantic Web Journal (2003)
15. Hearst, M.A.: Untangling text data mining. In: Association for Computational Linguistics (1999)
16. Kushmerick, N.: Wrapper induction: efficiency and expressiveness. Artif. Intell. **118**(1–2), 15–68 (2000)
17. Kushmerick, N., Weld, D.S., Doorenbos, R.B.: Wrapper induction for information extraction. In: Intl. Joint Conference on Artificial Intelligence (IJCAI), pp. 729–737 (1997)
18. Liu, L., Pu, C., Han, W.: Xwrap: an xml-enabled wrapper construction system for web information sources. In: International Conference on Data Engineering (2000)
19. Muslea, I., Minton, S., Knoblock, C.: Stalker: learning extraction rules for semistructured. In: Workshop on AI and Information Integration (1998)
20. Noy, N., Musen, M.: Prompt: algorithm and tool for automated ontology merging and alignment. In: Proceedings of the 17th Conference of the American Association for Artificial Intelligence (AAAI). AAAI Press, Menlo Park, CA (2000)
21. Pearson, K.: On the coefficient of racial likeliness. Biometrica **18**, 105–117 (1926)

22. Vadrevu, S., Gelgi, F., Davulcu, H.: Semantic partitioning web pages. In: The 6th International Conference on Web Information Systems Engineering (WISE) (2005)
23. Yang, G., Tan, W., Mukherjee, S., Ramakrishnan, I.V., Davulcu, H.: On the power of semantic partitioning of web documents. In: Workshop on Information Integration on the Web, Acapulco, Mexico (2003)