Preprint version for publication in TWEB

Robust Detection of Semi-structured Web Records Using DOM Structure Knowledge Driven Model

LIDONG BING, The Chinese University of Hong Kong
WAI LAM, The Chinese University of Hong Kong
TAK-LAM WONG, Caritas Institute of Higher Education

Web data record extraction aims at extracting a set of similar object records from a single Web page. These records have similar attributes or fields and they are presented with a regular format in a coherent region of the page. To tackle this problem, most existing works analyze the DOM tree of an input page. One major limitation of these methods is that the lack of a global view in detecting data records from an input page results in a myopic decision. Their brute-force searching manner in detecting various types of records degrades the flexibility and robustness. We propose a Structure Knowledge Oriented Global Analysis (Skoga) framework which can perform robust detection of different kinds of data records and record regions. The major component of Skoga framework is a DOM structure knowledge driven detection model which can conduct a global analysis on the DOM structure to achieve effective detection. The DOM structure knowledge consists of background knowledge as well as statistical knowledge capturing different characteristics of data records and record regions as exhibited in the DOM structure. The background knowledge encodes the semantics of labels indicating general constituents of data records and regions. The statistical knowledge is represented by some carefully designed features that capture different characteristics of a single node or a node group in the DOM. The feature weights are determined using a development data set via a parameter estimation algorithm based on structured output Support Vector Machine. An optimization method based on divide-and-conquer principle is developed making use of the DOM structure knowledge to quantitatively infer and recognize appropriate records and regions for a page. Extensive experiments have been conducted on four data sets. The experimental results demonstrate that our framework achieves higher accuracy compared with state-of-the-art methods.

Categories and Subject Descriptors: I.5.1 [**Pattern Recognition**]: Models—*Statistical, Structural*; H.3.m [**Information Storage and Retrieval**]: Miscellaneous

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Web Data Records, DOM Structure Knowledge, Web IE

The work described in this paper is substantially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: CUHK413510) and the Direct Grant of the Faculty of Engineering, CUHK (Project Codes: 2050476 and 2050522). This work is also affiliated with the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies.
The authors would like to thank Mohammed Kayed and Chia-Hui Chang for providing us their demo system of FiVaTech, as well as Gengxin Miao for providing us some details of their experiments.
Author's addresses: L. Bing, Department of Systems Engineering & Engineering Management, The Chinese University of Hong Kong. W. Lam, Department of Systems Engineering & Engineering Management, The Chinese University of Hong Kong. T. L. Wong, Department of Computer Science, Caritas Institute of Higher Education.
Contact author's e-mail address: L. Bing, binglidong@gmail.com.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

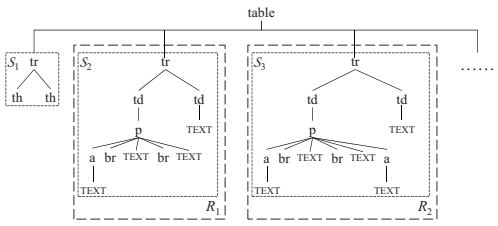
Traditional information extraction (IE) task aims at extracting data from basically unstructured free text. In contrast, Web IE deals with Web documents (or Web pages) which are semi-structured and coded with HTML. Typically, a Web page may describe either a single object or a group of similar objects. For example, the description page of a digital camera describes different aspects of the camera. On the other hand, the faculty list page of a department presents the information of a group of professors. Corresponding to the above two types, Web IE methods can be broadly categorized into two classes, namely, description details oriented extraction [Wong et al. 2006; Yang et al. 2010; Zhai and Liu 2007] and object records oriented extraction [Liu et al. 2003; Miao et al. 2009]. The former aims at extracting the description details of a single object from its description page, while the latter aims at extracting a set of similar object records. In this paper, we focus on the latter task. Many Web sites make use of regular format to present information units, known as *data records*, which have similar attributes or fields in a coherent region of a Web page, known as *data record region*. Some sites prefer to display the record information in a semi-structured way in static Web pages to facilitate easy browsing, such as a list of faculty members, a list of breaking events, etc. This brings in a large amount of relational Web tables [Cafarella et al. 2008], Web lists [Elmeleegy et al. 2009], and generic type of data record sets [Miao et al. 2009]. Some sites run a server-side program to fill products' information, retrieved from back-end databases, in a predefined template to generate Web pages, which are referred to as deep or dynamic Web pages [Cafarella et al. 2011; He et al. 2007; Madhavan et al. 2008]. Therefore, semi-structured information on the Web is tremendously popular. If such information can be exploited, it is very useful for developing various applications such as online market intelligence [Baumgartner et al. 2009], knowledge base population [Bing et al. 2013], etc.

Two samples of record regions are depicted in Figs. 1(a) and 1(b) with their DOM tree structures given in Figs. 1(c) and 1(d) respectively. In the first record region, each row of the table, excluding the header row $\mathcal{S}_1$, is a data record with the record R_1 corresponding to $\mathcal{S}_2$ and the record R_2 corresponding to $\mathcal{S}_3$. While in the second record region, each row of the table contains three data records. Fig. 2 depicts two more complex record regions. In the record region in Fig. 2(a) with its DOM given in Fig. 2(c), each data record is composed of several table rows. For example, the record R_1 is composed of three rows, i.e., $\mathcal{S}_2$, $\mathcal{S}_3$, and $\mathcal{S}_4$. In the record region in Fig. 2(b) with its DOM given in Fig. 2(d), different fields of R_1 and R_2 are intertwined in the first three subtrees, i.e., $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$. To tackle the problem of record detection, most existing works, such as MDR [Liu et al. 2003], DEPTA [Zhai and Liu 2006], NET [Liu and Zhai 2005], ViPER [Simon and Lausen 2005], FiVaTech [Kayed and Chang 2010], and our pervious method RST [Bing et al. 2011], analyze the DOM tree of an input page so as to detect data record regions as well as record boundaries. One major limitation of these methods is that the lack of a global view in detecting data records from an input page results in a myopic decision. For example, consider the record region given in Fig. 1(d). Myopic searching methods cannot conduct a comprehensive analysis that takes the three layers, namely, <table>, <tr>, and <td>, into global consideration. Precisely, each <tr> is processed separately and the similarity of records in different <tr>'s is not exploited. Consequently, the local decision on each <tr> cannot lead to a global optimal detection result. It is also possible that the <tr>'s in Fig. 1(d) are wrongly recognized as data records. In addition, this traversal searching manner is time consuming and cannot support real-time extraction needs.

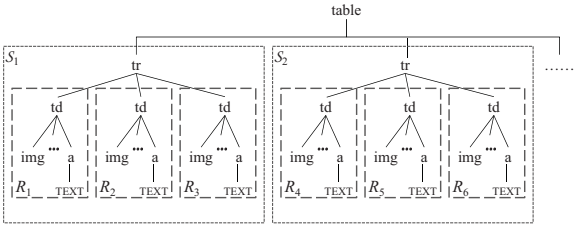
Another limitation of most existing works is due to their heuristic criteria related to characteristics embedded in the HTML source code or visual perception. When pro-

(a) A page fragment of a flat record region (a record region with its data records arranged one by one and each record is composed of one sub DOM tree).

(b) A page fragment of a nested record region (a record region with several subregions and each subregion has its records arranged one by one).



(c) DOM tree of (a).



(d) DOM tree of (b).

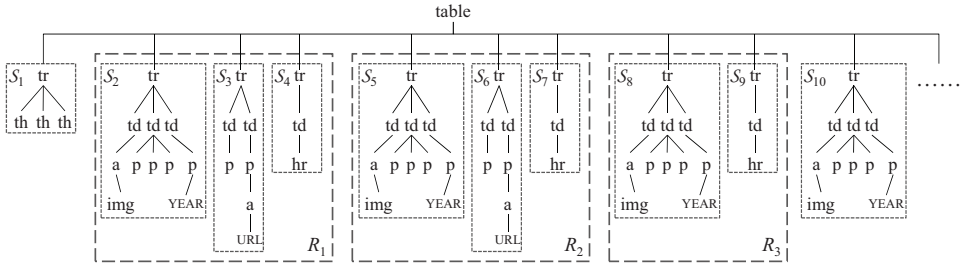
Fig. 1. Flat and nested record regions. S_i is a sub DOM tree, R_i is a data record.

cessing a new page, the heuristic criteria, such as the similarity between the subtree groups in MDR, DEPTA, and RST, as well as the tag path similarity in TPC [Miao et al. 2009], are applied to determine whether a fragment of Web page should be recognized as a record region. However, the heterogeneous characteristics of Web data make it infeasible to use some heuristic criteria to accurately capture different types of formatting manners. Moreover, MDR and DEPTA have limitations brought in by their assumption on the length of generalized nodes, i.e., all generalized nodes in the same record region must have the same number of subtrees. This criterion will fail when handling some cases in which the records contain different number of subtrees such as the record region in Fig. 2(c). Noticing this limitation, ViPER only calculates the similarity for single subtree pairs and constructs a similarity matrix. However, some heuristic rules are employed to process this matrix to detect the record region as well as record boundaries.

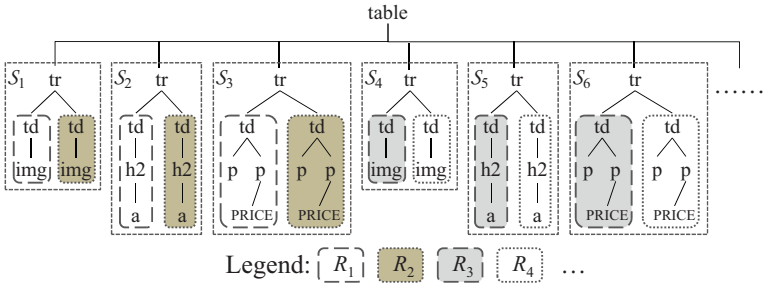
Statistical models were also exploited in Web data extraction in some existing works [Yang et al. 2009; Zhu et al. 2006]. Zhu et al. proposed a model based on Hierarchical Conditional Random Field (HCRF) to conduct record detection as well as attribute labeling and achieved some good performance in tackling product record extraction [Zhu et al. 2006]. The overall design of HCRF in [Zhu et al. 2006] focuses extensively on product records and it involves some specific product-oriented labels such as product name and price, although it may be able to utilize the model in general record extraction after necessary modifications on label and feature design. Another issue is that the authors assume that the boundaries of the visual blocks obtained from VIPS [Cai et al. 2003] are coincident with the boundaries of the records with multiple subtrees. However, the page rendering operation may encounter troubles when the separated cascading style sheet (CSS) and JavaScript files are not available.

(a) A page fragment of a complicated flat record region (a record region with its data records arranged one by one and each record is composed of several sub DOM trees).

(b) A page fragment of an intertwined record region (a record region with the records whose attributes intertwined with other records' attributes).



(c) DOM tree of (a).



(d) DOM tree of (b).

Fig. 2. Complicated flat and intertwined record regions. S_i is a sub DOM tree, R_i is a data record.

Consequently, the boundaries of the visual blocks may not be reliable. Furthermore, HCRF only defines local similarity based features between adjacent nodes in a sibling sequence. It thus cannot exploit the global regularity of the subtree sequence in a particular record region as exemplified in Figs. 1 and 2.

In this paper, we present a Structure Knowledge Oriented Global Analysis (Skoga, pronounced as [səʊgɑː]) framework which adopts a uniform manner to perform robust detection of different kinds of data records and record regions, namely, *flat record region* as exemplified in Fig. 1(a), *nested record region* as exemplified in Fig. 1(b), *complicated flat record region* as exemplified in Fig. 2(a), and *intertwined record region* as exemplified in Fig. 2(b). One major component of Skoga framework is a DOM structure knowledge driven detection model which can conduct a global analysis on the DOM

structure addressing the major limitations of existing methods and achieve effective detection. Let us consider a Web page and its corresponding DOM tree structure. The extraction of data records from the page is equivalent to identifying, in its DOM tree, the subtree corresponding to the record region and the subtrees corresponding to the data records. For the flat record region in Fig. 1(a) with its DOM given in Fig. 1(c), our framework can quantitatively assign the highest score to a recognition that correctly identifies $\mathcal{S}_2$, $\mathcal{S}_3$, etc. as data records and the subtree corresponding to the <table> node as a record region. With respect to the nested record region in Fig. 1(b) with its DOM given in Fig. 1(d), our framework can quantitatively identify the subtrees corresponding to the <td> nodes as data records. At the same time, it identifies the subtrees corresponding to the <tr> nodes as *subregions* but not data records even though these subtrees have similar structures. For the complicated flat record region in Fig. 2(a) with its DOM given in Fig. 2(c), whose records have variable number of subtrees, our framework detects the *complicated flat records* by recognizing the *beginning segments* such as $\mathcal{S}_2$ and the *inside segments* such as $\mathcal{S}_3$ of the records in the subtree sequence with global analysis. With respect to the intertwined example in Fig. 2(b) with its DOM given in Fig. 2(d), our framework first detects the *composite records*, each of which is composed of three subtrees (e.g., $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$) and contains two *intertwined records* (e.g., R_1 and R_2). Then, the intertwined records are assembled from the detected composite records with an assembling stage of our framework. Although the subtree $\mathcal{S}_1$'s in Figs. 1(c) and 2(c) are also part of the regions, they are not part of any record and should be regarded as *region note* providing only some note information of the region and records.

The DOM structure knowledge in Skoga framework consists of background knowledge as well as statistical knowledge capturing different characteristics of data records and record regions. Specifically, the background knowledge encodes the semantics of labels indicating general constituents of data records and regions. In addition, it captures some logical relations governing certain structural constraints among the labels to be assigned to the nodes in the DOM structure. The statistical knowledge is represented by some carefully designed features that capture different characteristics of a single node or a node group in the DOM, such as the similarity of the neighboring subtrees, the similarity of one subtree with its siblings, etc. To allow different impacts for different features, there is a weight associated with each feature. The feature weights in the DOM structure knowledge are determined using a development data set via a parameter estimation algorithm based on structured output Support Vector Machine model [Tsochantaridis et al. 2005]. This model can tackle the inter-dependency among the labels on the nodes of the DOM structure and its superiority in Web IE from structured representation of Web pages was also investigated in other tasks [Zhao et al. 2011]. Another advantage is that our model can capture long range features in a sibling sequence such as the occurrence-related features that exploit the global regularity of the sequence. In our proposed parameter estimation algorithm, a record region oriented loss function is designed so that the acquired statistical knowledge can deal with multiple regions in one page. The development data set was arbitrarily collected from different Web sites covering different kinds of record regions such as the examples given in Figs. 1 and 2. An optimization method based on divide and conquer principle is developed making use of the DOM structure knowledge to quantitatively infer the best record and region recognition for a page. Extensive experiments have been conducted on four data sets. The experimental results demonstrate that our framework achieves higher accuracy compared with state-of-the-art methods.

In summary, the contributions of the paper are as follows:

- We propose a framework which can perform robust detection of different kinds of data records and record regions with a DOM structure knowledge driven detection model by conducting a global analysis on the DOM structure.
- The DOM structure knowledge is carefully designed and it consists of background knowledge and statistical knowledge. The former can encode the semantics of labels indicating general constituents of data records and regions. And the latter can capture different characteristics of a single node or a node group in the DOM.
- We develop a parameter estimation algorithm based on structured output Support Vector Machine model to determine the feature weights in the statistical knowledge. This algorithm can tackle the inter-dependency among the labels on the nodes of the DOM structure.
- An optimization method based on divide-and-conquer principle is developed making use of the DOM structure knowledge to quantitatively infer the best record and region recognition for a page.

The remainder of this paper is organized as follows. In Section 2, the overview of Skoga framework is introduced. The design of DOM structure knowledge is presented in Section 3. The inference of optimal label assignment is presented in Section 4. The determination of feature weights of the DOM structure knowledge is presented in Section 5. After that, the assembling method of intertwined data records is given in Section 6. Then, the experimental results as well as discussions are presented in Section 7. Section 8 provides more comprehensive discussions on related works. We conclude our work and propose some future directions in Section 9.

2. OVERVIEW OF SKOGA FRAMEWORK

In our proposed Structure Knowledge Oriented Global Analysis (Skoga) framework, the goal of record region detection and data record extraction is tackled by identifying appropriate portions in the DOM structure as record regions as well as data records in a region. It can be formulated as a problem of assigning suitable labels to the nodes of the DOM tree. Taking the flat region in Fig. 1(a) with its DOM tree given in Fig. 1(c) as an example, the label "REC-S" (namely, *rec*ord composed of a *s*ingle subtree) should be assigned to the <tr> tags which are the roots of the subtrees $\mathcal{S}_2$, $\mathcal{S}_3$, etc. Furthermore, the label "REGION" should be assigned to the root <table> tag of the table, and the label "REGNOT" (namely, *reg*ion *not*e node as explained in Section 1) should be assigned to the <tr> tag of $\mathcal{S}_1$. For the nested region in Fig. 1(b) with its DOM tree given in Fig. 1(d), the label "REC-S" should be assigned to the <td> tags. Also the label "SUBREG" (namely, *subreg*ion of records) should be assigned to the <tr> tags which are the roots of the subtrees $\mathcal{S}_1$, $\mathcal{S}_2$, etc., and the label "REGION" should be assigned to the root <table> tag.

Formally, let $\mathbf{x}$ denote the DOM tree of a particular Web page, and a single node in $\mathbf{x}$ is denoted by x. Let $\mathbf{y}$ denote a label assignment for $\mathbf{x}$, and a single label is denoted by $y \in \mathcal{Y}$ where $\mathcal{Y}$ is the set of all possible labels. To achieve the goal of record region detection and data record extraction, we can formulate it as an optimization problem via a global objective function to obtain $\mathbf{y}^*$ such that:

$$\mathbf{y}^* = \underset{\mathbf{y}}{\operatorname{argmax}}\, F(\mathbf{x}, \mathbf{y}; \mathbf{w}), \tag{1}$$

where F is an objective function that evaluates the fitness of $\mathbf{y}$ for $\mathbf{x}$ with the guidance of the DOM structure knowledge $\mathbf{w}$. Such design facilitates a global analysis on the DOM structure to achieve accurate detection of records and regions. The DOM structure knowledge is composed of background knowledge and statistical knowledge. The background knowledge encodes the semantics of labels indicating general constituents of data records and regions. In addition, it captures some logical relations governing certain structural constraints among the labels to be assigned to the nodes of the DOM

structure. The statistical knowledge consists of the design of features capturing different characteristics of a single node or a node group in the DOM. Furthermore, these features are able to distill the difference among different types of record regions so as to identify them accurately. Some examples of the features are the structure feature of a single node, the similarity of the neighboring subtrees, the similarity of one subtree with its siblings, etc. To allow different impacts for different features, each feature is associated with a weight.

Fundamentally, some existing methods such as DEPTA [Zhai and Liu 2006], ViPER [Simon and Lausen 2005], FiVaTech [Kayed and Chang 2010], and RST [Bing et al. 2011] can also be represented in the form of $F(\mathbf{x}, \mathbf{y}; \mathbf{w})$. For example, MDR and DEPTA calculate the similarity between two neighboring generalized nodes in a particular region derived from the DOM tree $\mathbf{x}$. According to the degree of satisfaction on some predefined heuristic criteria such as similarity threshold, a certain label assignment $\mathbf{y}$ is returned. The criteria they employ can also be regarded as a simple kind of DOM structure knowledge. Different from these methods, our model conducts a global analysis on the fitness of $\mathbf{y}$ for $\mathbf{x}$ driven by the DOM structure knowledge and it takes the inter-dependency among the labels on the nodes into consideration. To infer the best label assignment $\mathbf{y}^*$ for $\mathbf{x}$, an efficient optimization method is developed using divide-and-conquer principle in polynomial time.

Let us return to the complicated flat region given in Fig. 2(c). Our model is able to detect the data records accurately by assigning the label "REC-B" (*b*eginning segment of a *rec*ord) to the subtrees $\mathcal{S}_2$, $\mathcal{S}_5$, $\mathcal{S}_8$, etc., and the label "REC-I" (*i*nside segment of a *rec*ord) to the subtrees $\mathcal{S}_3$, $\mathcal{S}_4$, $\mathcal{S}_6$, $\mathcal{S}_7$, $\mathcal{S}_9$, etc. The challenges raised by the variable number of subtrees in these records are tackled with the global analysis on the characteristics of the subtree sequence in this record region. Specifically, all possible label sequences are evaluated during the inference of the best label assignment for the sequence $\mathcal{S}_1, \ldots, \mathcal{S}_n$. Finally, the label sequence " 'REGNOT', 'REC-B', 'REC-I', 'REC-I', 'REC-B', 'REC-I', 'REC-I', 'REC-B', 'REC-I', 'REC-B', ..." achieves the highest value for the global objective function. With respect to the intertwined example given in Fig. 2(d), our model assigns the label "REC-B" to the subtrees $\mathcal{S}_1$, $\mathcal{S}_4$, etc., and the label "REC-I" to the subtrees $\mathcal{S}_2$, $\mathcal{S}_3$, $\mathcal{S}_5$, $\mathcal{S}_6$, etc. Thus, the composite data records such as $\mathcal{S}_{1:3}$ and $\mathcal{S}_{4:6}$ are detected. Each segment <tr> in the composite records contains the constituents of two intertwined data records. And then, our Skoga framework invokes a separate assembling stage to assemble the intertwined records R_1, R_2, R_3, and R_4 from the detected composite records in Fig. 2(d).

The feature weights in the DOM structure knowledge are determined using a development data set via a parameter estimation algorithm. To allow better generalization capability of the estimated feature weights when tackling the heterogenous Web pages, the maximum margin principle with soft margin is employed. Specifically, the parameter estimation algorithm is developed based on structured output Support Vector Machine (SVM) model [Tsochantaridis et al. 2005]. The label output of a node in the DOM structure exhibits a tight interaction with the labels of its connected nodes such as its parent node and siblings. The structured output SVM model is able to tackle the inter-dependency among the labels on the nodes of the DOM structure so as to provide more accurate labeling solutions. Furthermore, a record region oriented loss function is designed to penalize the missing of record regions. Therefore, the acquired statistical knowledge can effectively identify multiple regions, if exist, in a single page. The development data set was arbitrarily collected from different Web sites covering different kinds of record regions such as the examples given in Figs. 1 and 2. An optimization method, namely hierarchical Viterbi algorithm, is developed based on divide-and-conquer principle, which makes use of the DOM structure knowledge

to quantitatively infer the optimal label assignment of record and region recognition from a page in polynomial time.

Once the DOM structure knowledge including the feature weights is determined, Skoga framework can be directly applied to detect common kinds of record regions and data records as illustrated above from any Web sites and domains without the need of labeled data or training. As a result, our framework is more robust when processing different types of record regions so that it can achieve better effectiveness and higher efficiency. Note that we do not generate any wrapper and Skoga is site-independent. Another characteristic of Skoga is that when there is a need to detect application oriented record regions and data records which are different from the common kinds of regions or records, Skoga can conduct a training process with the application-specific labeled data to generate a tailor-made detection model for the intended application.

3. DESIGN OF DOM STRUCTURE KNOWLEDGE

3.1. Background Knowledge

As mentioned above, each node in the DOM tree is assigned a label. We design eight types of labels denoted as $\mathcal{Y}$ to capture general constituents of record regions and data records. The details of the labels are given as follows.

`REGION`: The DOM node with this label is the root node of a subtree corresponding to a record region. The region should contain either a set of data records or a set of subregions. A data record under this region may be composed of several subtrees of the current region, such as the examples given in Fig. 2.

`SUBREG`: The DOM node with this label is the root node of a subtree corresponding to a subregion. A subregion should contain a set of data records such as the examples given in Fig. 1(b). Each data record under this subregion may be composed of several subtrees of the current subregion.

`REC-S`: The DOM node with this label is the root node of a subtree corresponding to a complete data record and it may be composed of a group of components. For example, each data record in Fig. 1(d) contains an image <img> and a link <a>.

`REC-B`: The DOM node with this label is the root node of a subtree corresponding to the beginning segment of a data record such as $\mathcal{S}_2$, $\mathcal{S}_5$ in Fig. 2(c) and $\mathcal{S}_1$, $\mathcal{S}_4$ in Fig. 2(d).

`REC-I`: The DOM node with this label is the root node of a subtree corresponding to an inside segment of a data record such as $\mathcal{S}_3$, $\mathcal{S}_4$ in Fig. 2(c) and $\mathcal{S}_2$, $\mathcal{S}_3$ in Fig. 2(d). Note that a segment node (labeled with `REC-B` or `REC-I`) may be composite and contain constituents of several data records such as $\mathcal{S}_1$, $\mathcal{S}_2$, etc. in Fig. 2(d).

`REGNOT`: The DOM node with this label is the root node of a subtree containing some explanation information on the data records in a record region or a subregion, such as $\mathcal{S}_1$'s in Figs. 1(c) and 2(c). Note that such node is not a part of any data record.

`RECCMP`: The DOM node with this label is the root node of a subtree corresponding to a component of a data record. Each constituent with any granularity in a data record or record segment can be regarded as a component. Thus, each descendant node under a data record or record segment has this label, such as the nodes <img> and <a> in Fig. 1(d).

`OTHNOD`: The DOM node with this label is the root node of a subtree corresponding to any other portion located outside record regions. Therefore, such node is not a part of any record region.

Fig. 3 shows an example of label assignment for the record region in Fig. 2(a) with DOM structure given in Fig. 2(c). Our label design allows broad and general types of data records and record regions. We also design some logical relations among the labels based on the common understanding of data records and record regions. Let x_p denote

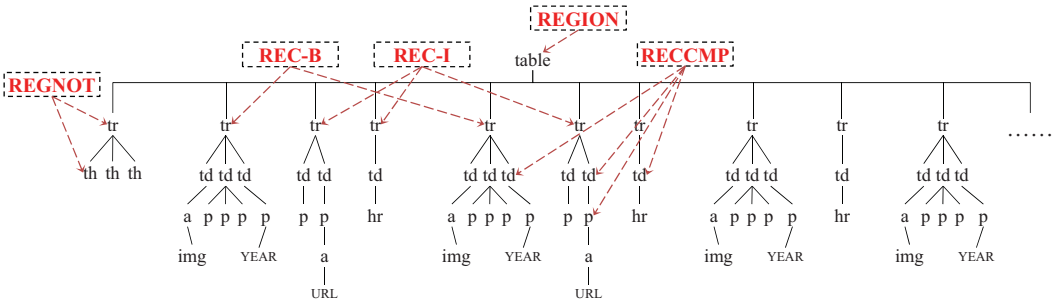


Fig. 3. Label assignment for Fig. 2(c).

a particular node and x_c denote one child node of x_p. Let $\mathbf{y}$ denote the label assignment of the DOM $\mathbf{x}$ from where x_p and x_c originate; $\mathbf{y}(x_p)$ and $\mathbf{y}(x_c)$ denote the labels of x_p and x_c given by $\mathbf{y}$. The logical relations, namely BK1 to BK8, are presented as follows.

- BK1: $\mathbf{y}(x_c) = \texttt{REGION} \rightarrow \mathbf{y}(x_p) = \texttt{OTHNOD}$
- BK2: $\mathbf{y}(x_c) = \texttt{SUBREG} \rightarrow \mathbf{y}(x_p) \in \{\texttt{REGION}, \texttt{SUBREG}\}$
- BK3: $\mathbf{y}(x_c) = \texttt{REC-S} \rightarrow \mathbf{y}(x_p) \in \{\texttt{REGION}, \texttt{SUBREG}\}$
- BK4: $\mathbf{y}(x_c) = \texttt{REC-B} \rightarrow \mathbf{y}(x_p) \in \{\texttt{REGION}, \texttt{SUBREG}\}$
- BK5: $\mathbf{y}(x_c) = \texttt{REC-I} \rightarrow \mathbf{y}(x_p) \in \{\texttt{REGION}, \texttt{SUBREG}\}$
- BK6: $\mathbf{y}(x_c) = \texttt{REGNOT} \rightarrow \mathbf{y}(x_p) \in \{\texttt{REGION}, \texttt{SUBREG}, \texttt{REGNOT}\}$
- BK7: $\mathbf{y}(x_c) = \texttt{RECCMP} \rightarrow \mathbf{y}(x_p) \in \{\texttt{REC-S}, \texttt{REC-B}, \texttt{REC-I}, \texttt{RECCMP}\}$
- BK8: $\mathbf{y}(x_c) = \texttt{OTHNOD} \rightarrow \mathbf{y}(x_p) = \texttt{OTHNOD}$

Taking BK3 as an example, if a particular node is labeled as `REC-S`, we only need to consider two candidate labels for its parent node, namely, `REGION` and `SUBREG`. These logic formulae are employed to provide guidance and constraints for the inference algorithm (presented in Section 4) when labeling a new DOM tree so that the inference results are more robust against noise and the inference algorithm is more efficient. These logic formulae are also used in the feature weight estimation algorithm (presented in Section 5) in which the inference for the optimal and the second optimal label assignments is needed during the optimization of the cost function.

3.2. Statistical Knowledge

As mentioned above, our Skoga framework captures the statistical knowledge represented as features and their corresponding weights. This knowledge is utilized to quantitatively evaluate the fitness of a label assignment $\mathbf{y}$ for the DOM tree $\mathbf{x}$ of a particular Web page via an objective function F. Let $\mathbf{\Psi}(\mathbf{x}, \mathbf{y})$ denote the combined feature representation of $\mathbf{x}$ and its label assignment $\mathbf{y}$. Thus, the objective function F in Equation 1 is formulated as:

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) \equiv \langle \mathbf{\Psi}(\mathbf{x}, \mathbf{y}), \mathbf{w} \rangle, \tag{2}$$

which is the linear combination of the features in $\mathbf{\Psi}(\mathbf{x}, \mathbf{y})$ with their corresponding weights given in $\mathbf{w}$. To exploit the tree structure and capture the inter-dependency among labels, we design three types of features in $\mathbf{\Psi}(\mathbf{x}, \mathbf{y})$, namely, single node features, sibling features, and parent-children features.

Single node features are summarized from a single DOM node. Let $\mathbf{\Phi}(x)$ denote the feature vector related to a particular node x. The combined feature map of x and its label $y \in \mathcal{Y}$ is defined as:

$$\mathbf{\Psi}(x, y) \equiv \mathbf{\Phi}(x) \otimes \mathbf{\Lambda}^c(y), \tag{3}$$

Table I. The features used in the statistical knowledge.

Single Node Features
Tag features: These features indicate whether the current node is a special tag type, such as <table>,   , <dl>, etc. These tags have higher chance to be used in formatting data records. Although the examples in Figs. 1 and 2 are rooted at <table>, it should be noted that our framework is not tag-dependent and the tag features only add some contribution in the overall evaluation.
Text appearance features: These features summarize the general characteristics of the text content contained by the current node, such as the fraction of anchor text, the number of different fonts used, etc.
Structure features: These features capture the aggregated characteristics of the subtree rooted at the current node, such as the number of child nodes, the number of different tags among its children, standard deviation of child subtrees' height, etc.
Special functionality features: These features capture some special characteristics that are often observed in data records, such as URL string, cash symbol, image, etc.
Sibling Features
Structure similarity: This feature is defined as the similarity of the subtrees rooted at x_{c_t} and $x_{c_{t+1}}$. Normally, the neighboring records as well as subregions as exemplified in Fig. 1 share higher similarity, while the neighboring record segments as exemplified in Fig. 2 share less similarity.
Skeleton similarity: Different from the structure similarity, these features capture the similarity of the skeletons of the subtrees rooted at x_{c_t} and $x_{c_{t+1}}$. Skeleton refers to the top level structure of a subtree, for instance, 2-layer or 3-layer skeletons. Skeleton similarity can overcome the dissimilarity caused by optional fields, such as the <a> node in the bottom right portion of R_2 in Fig. 1(c).
Text similarity features: These features summarize the text similarity between the contents of x_{c_t} and $x_{c_{t+1}}$. For example, the text overlapping feature indicates the fraction of the overlapping words between the text contents. The text length difference feature indicates the length difference of the text contents.
Parent-children Features
Occurrence based on structure similarity: The number of occurrences of x_c (or $\langle x_{c_t}, x_{c_{t+1}} \rangle$) among its sibling sequence based on structure similarity. For example, the subtree structure corresponding to the data records in Fig. 1(c) occurs many times in the child sequence of the table.
Occurrence difference between siblings: The difference of the occurrence number between x_{c_t} and $x_{c_{t+1}}$. This feature can help us capture the beginning of the record sequence more precisely, such as the ones given in Figs. 1(c) and 2(c).
Occurrence interval: These features capture the characteristics of the intervals between two successive occurrences of x_{c_t} (or $\langle x_{c_t}, x_{c_{t+1}} \rangle$). For example, the interval length feature is defined as the average length of the intervals. The standard deviation feature indicates whether the intervals are regular, such as the subtree structure corresponding to the beginning segment of the records (i.e., $\mathcal{S}_2$, $\mathcal{S}_5$, etc.) in Fig. 2(c) which occurs regularly.
Occurrence span: The number of siblings spanned by the first occurrence and the last occurrence of x_c (or $\langle x_{c_t}, x_{c_{t+1}} \rangle$). Normally, records occupy a large fraction of the subtree sequence in a record region.
Occurrence-based pivot score: This feature is defined to capture the beginning segment of a record and it is calculated as: $ps(x_{c_t}) = \begin{cases} ps(x_{c_k}) & if\ x_{c_t}\ is\ a\ reoccurence\ of\ x_{c_k}\ (k < t) \\ f(x_{c_t}) \frac{1}{I\_var(x_{c_t})+\sigma} - \max\{ps(x_{c_1}), \ldots, ps(x_{c_{t-1}})\} & otherwise \end{cases}$, where $f(x_{c_t})$ is the number of occurrence of x_{c_t}; $I\_var(x_{c_t})$ is the variance of the occurrence intervals; $\sigma = 0.01$ is used to avoid zero denominator. Considering the example in Fig. 2(c), $ps(\mathcal{S}_2)$ is large, $ps(\mathcal{S}_3)$ and $ps(\mathcal{S}_4)$ are small, $ps(\mathcal{S}_5)$ is also large since $\mathcal{S}_5$ is a reoccurrence of $\mathcal{S}_2$.

where $\otimes$ is the operator of tensor multiplication. $\mathbf{\Lambda}^c(y)$ is the canonical representation of the label y:

$$\mathbf{\Lambda}^c(y) \equiv (\delta(y_1, y), \delta(y_2, y), \cdots, \delta(y_{|\mathcal{Y}|}, y)), \tag{4}$$

where δ is an indicator function which has the value 1 if $y_i = y$ and the value 0 otherwise. From Equation 3, it can be seen that each single feature is mapped to a dimension according to the label y of x. We design four types of single node features to depict different characteristics of a single node and they are described in the first section of Table I. Note that Skoga is tag-independent and the tag related features only add some contribution in the overall evaluation.

Sibling features capture the relations between two neighboring nodes. Let x_{c_t} and $x_{c_{t+1}}$ be the root nodes of a pair of neighboring sibling subtrees, and their labels are

y_{c_t} and $y_{c_{t+1}}$ respectively. The combined feature map of the pair $\langle x_{c_t}, x_{c_{t+1}} \rangle$ and the corresponding labels is defined as:

$$\mathbf{\Psi}(\langle x_{c_t}, x_{c_{t+1}} \rangle, y_{c_t}, y_{c_{t+1}}) \equiv \mathbf{\Phi}(\langle x_{c_t}, x_{c_{t+1}} \rangle) \otimes \mathbf{\Lambda}^c(y_{c_t}) \otimes \mathbf{\Lambda}^c(y_{c_{t+1}}), \tag{5}$$

where $\mathbf{\Phi}(\langle x_{c_t}, x_{c_{t+1}} \rangle)$ represents the features summarized from this sibling pair, $\otimes$ and $\mathbf{\Lambda}^c(\cdot)$ are defined as above. We design three types of sibling features and they are described in the second section of Table I.

Parent-children features are designed to capture the relations between a particular node and its entire sibling sequence. Let $\langle x_p, x_c \rangle$ denote a parent-child pair, and their labels are y_p and y_c respectively. The combined feature map of the pair $\langle x_p, x_c \rangle$ and the corresponding labels is defined as:

$$\mathbf{\Psi}(\langle x_p, x_c \rangle, y_p, y_c) \equiv \mathbf{\Phi}(\langle x_p, x_c \rangle) \otimes \mathbf{\Lambda}^c(y_p) \otimes \mathbf{\Lambda}^c(y_c), \tag{6}$$

where $\mathbf{\Phi}(\langle x_p, x_c \rangle)$ represents the features summarized from this parent-child pair. Similarly, the feature map for the triple $\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle$ and the corresponding labels can be defined as:

$$\mathbf{\Psi}(\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle, y_p, y_{c_t}, y_{c_{t+1}}) \equiv \mathbf{\Phi}(\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle) \otimes \mathbf{\Lambda}^c(y_p) \otimes \mathbf{\Lambda}^c(y_{c_t}) \otimes \mathbf{\Lambda}^c(y_{c_{t+1}}), \tag{7}$$

where $\mathbf{\Phi}(\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle)$ represents the features summarized from the triple. We design five types of parent-children features and they are described in the third section of Table I.

The combined feature representation of the DOM tree $\mathbf{x}$ and its label assignment $\mathbf{y}$ is the combination of the above types of feature maps:

$$\mathbf{\Psi}(\mathbf{x}, \mathbf{y}) \equiv \begin{pmatrix} \sum_x \mathbf{\Psi}(x, y)^T \\ \sum_{\langle x_{c_t}, x_{c_{t+1}} \rangle} \mathbf{\Psi}(\langle x_{c_t}, x_{c_{t+1}} \rangle, y_{c_t}, y_{c_{t+1}})^T \\ \sum_{\langle x_p, x_c \rangle} \mathbf{\Psi}(\langle x_p, x_c \rangle, y_p, y_c)^T \\ \sum_{\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle} \mathbf{\Psi}(\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle, y_p, y_{c_t}, y_{c_{t+1}})^T \end{pmatrix}^T. \tag{8}$$

As shown above, different features are combined and the difference of their impacts will be captured by the corresponding weights in $\mathbf{w}$ determined via a parameter estimation algorithm with the guidance of a development data set.

4. FINDING OPTIMAL LABEL ASSIGNMENT

One major task in Skoga framework is to find the optimal label assignment of a DOM tree by maximizing $F(\mathbf{x}, \mathbf{y}; \mathbf{w})$ as in Equation 1 with F defined in Equation 2. As a result, the aim is to solve the following optimization problem:

$$\begin{aligned} \mathbf{y}^* &= \operatorname*{argmax}_{\mathbf{y}} F(\mathbf{x}, \mathbf{y}; \mathbf{w}) \\ &= \operatorname*{argmax}_{\mathbf{y}} \langle \mathbf{\Psi}(\mathbf{x}, \mathbf{y}), \mathbf{w} \rangle. \end{aligned} \tag{9}$$

It is referred to as the inference task. The inference algorithm of the entire DOM tree can be tackled with divide-and-conquer principle. Specifically, a hierarchical Viterbi algorithm is designed to obtain the optimal label assignment in polynomial time. It conducts the inference in a bottom-up manner and starts from the subtree whose height is 1. After that, the intermediate results are utilized in the inference of their parent trees.

4.1. Inference for Bottom Subtrees

We first describe the inference algorithm for the subtrees whose height is 1. Let x_p denote the parent node and x_{c_t} denote a child node under x_p. Let $\mathbf{y}_p = \{y_{p,1}, y_{p,2}, \cdots\}$

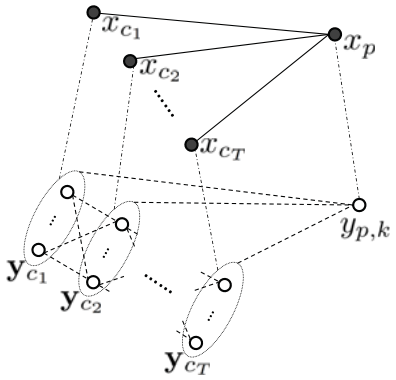


Fig. 4. Inference lattice structure of a subtree whose height is 1.

denote the candidate labels of x_p and $\mathbf{y}_{c_t} = \{y_{c_t,1}, y_{c_t,2}, \cdots\}$ denote the candidate labels of x_{c_t}. Take the tree $\mathbf{x} = \{x_p, x_{c_1}, x_{c_2}, \cdots, x_{c_T}\}$ in Fig. 4 as an example, where the filled nodes represent the root node and the leaf nodes respectively. The unfilled nodes represent the label candidates of the corresponding DOM nodes. T is the total number of the children. We first fix the label of the root x_p to be $y_{p,k}$, and infer the optimal labels for the child sequence from their candidate label sets. The inference mechanism considering different combinations of labels can be represented as a lattice structure. After that, we enumerate all possible candidate labels of x_p in $\mathbf{y}_p$ to obtain the global optimal labeling.

Let $\hat{F}^{p,k}_{c_t,i}$ denote the intermediate maximum objective value on $\mathbf{x} = \{x_p, x_{c_1}, \cdots, x_{c_t}\}$ achieved by an assignment that assigns the label $y_{p,k}$ to the root x_p and $y_{c_t,i}$ to the child x_{c_t}. Then, $\hat{F}^{p,k}_{c_{t+1},j}$ is calculated as follows:

$$\hat{F}^{p,k}_{c_{t+1},j} = \max_{y_{c_t,i} \in \mathbf{y}_{c_t}} \{\hat{F}^{p,k}_{c_t,i} + \Delta F^{p,k}_{(c_t,i),(c_{t+1},j)}\}, \tag{10}$$

where $\Delta F^{p,k}_{(c_t,i),(c_{t+1},j)}$ denotes the increment of the objective value when taking the node $x_{c_{t+1}}$ into account with label $y_{c_{t+1},j}$, and it is calculated as:

$$\Delta F^{p,k}_{(c_t,i),(c_{t+1},j)} = \langle \Delta\mathbf{\Psi}(x_p, x_{c_t}, x_{c_{t+1}}, y_{p,k}, y_{c_t,i}, y_{c_{t+1},j}), \mathbf{w} \rangle, \tag{11}$$

where $\Delta\mathbf{\Psi}$ is the change of the feature vector including the single node features of $x_{c_{t+1}}$, the sibling features of $\langle x_{c_t}, x_{c_{t+1}} \rangle$, and the parent-children features of $\langle x_p, x_{c_{t+1}} \rangle$ and $\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle$. To be precise, $\Delta\mathbf{\Psi}$ is represented as:

$$\Delta\mathbf{\Psi}(x_p, x_{c_t}, x_{c_{t+1}}, y_{p,k}, y_{c_t,i}, y_{c_{t+1},j}) = \begin{pmatrix} \mathbf{\Psi}(x_{c_{t+1}}, y_{c_{t+1},j})^T \\ \mathbf{\Psi}(\langle x_{c_t}, x_{c_{t+1}} \rangle, y_{c_t,i}, y_{c_{t+1},j})^T \\ \mathbf{\Psi}(\langle x_p, x_{c_{t+1}} \rangle, y_{p,k}, y_{c_{t+1},j})^T \\ \mathbf{\Psi}(\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle, y_{p,k}, y_{c_t,i}, y_{c_{t+1},j})^T \end{pmatrix}^T. \tag{12}$$

The contributed objective value by x_p and x_{c_1} is included in $\hat{F}^{p,k}_{c_1,i}$, which is calculated as:

$$\hat{F}^{p,k}_{c_1,i} = \left\langle \begin{pmatrix} \mathbf{\Psi}(x_{c_1}, y_{c_1,i})^T + \mathbf{\Psi}(x_p, y_{p,k})^T \\ \mathbf{0} \\ \mathbf{\Psi}(\langle x_p, x_{c_1} \rangle, y_{p,k}, y_{c_1,i})^T \\ \mathbf{0} \end{pmatrix}^T, \mathbf{w} \right\rangle. \tag{13}$$

Recall that the labels of x_p and its child x_{c_t} should satisfy the logic formulae in Section 3.1. Taking these formulae into consideration in the calculation of Equation 10, when a particular $y_{c_{t+1},j}$ together with $y_{p,k}$ violates any formula, it can be automatically pruned. Similarly, Equation 13 is also constrained by these logic formulae.

For a pre-assigned label $y_{p,k}$ of x_p, the maximum objective value that can be achieved by labeling its child nodes is denoted by $\hat{F}^{p,k}$, which is calculated as:

$$\hat{F}^{p,k} = \max_{y_{c_T,i} \in \mathbf{y}_{c_T}} \hat{F}^{p,k}_{c_T,i}, \tag{14}$$

Finally, the maximum objective value achieved by the optimal label assignment can be obtained by enumerating all possible $y_{p,k}$ for x_p:

$$\hat{F} = \max_{y_{p,k} \in \mathbf{y}_p} \hat{F}^{p,k}. \tag{15}$$

The time complexity of the above inference for a bottom single-depth tree can be derived from that of the standard Viterbi algorithm [Ryan and Nudd 1993]. For each candidate label of the root x_p, the standard Viterbi algorithm is invoked to infer the optimal labels for the children. Therefore, the time complexity is $\mathbf{O}(|\mathcal{Y}|^3 * T)$, where $\mathcal{Y}$ is the set of all possible labels. Note that the above time complexity analysis has not taken the logical relations (see Section 3.1) into consideration. Incorporating the logical relations makes the inference algorithm even more efficient.

4.2. Recursive Inference for Higher Subtrees

In the previous subsection, each child node is assumed to be a leaf. When processing the higher level subtrees, the intermediate results from the lower level subtrees rooted at each child x_{c_t} are taken into consideration. Let $\hat{F}^{c_t,i}$ denote the optimal value achieved in labeling the subtree rooted at x_{c_t} which is labeled with $y_{c_t,i}$. The objective value increment $\Delta F^{p,k}_{(c_t,i),(c_{t+1},j)}$ is calculated as:

$$\Delta F^{p,k}_{(c_t,i),(c_{t+1},j)} = \hat{F}^{c_{t+1},j} + \langle \Delta \mathbf{\Psi}'(x_p, x_{c_t}, x_{c_{t+1}}, y_{p,k}, y_{c_t,i}, y_{c_{t+1},j}), \mathbf{w} \rangle, \tag{16}$$

where $\Delta \mathbf{\Psi}'$ is calculated as:

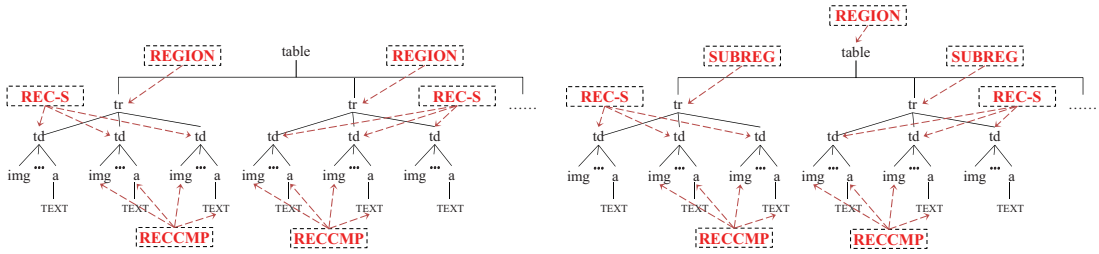
$$\Delta \mathbf{\Psi}'(x_p, x_{c_t}, x_{c_{t+1}}, y_{p,k}, y_{c_t,i}, y_{c_{t+1},j}) = \begin{pmatrix} \mathbf{0} \\ \mathbf{\Psi}(\langle x_{c_t}, x_{c_{t+1}} \rangle, y_{c_t,i}, y_{c_{t+1},j})^T \\ \mathbf{\Psi}(\langle x_p, x_{c_{t+1}} \rangle, y_{p,k}, y_{c_{t+1},j})^T \\ \mathbf{\Psi}(\langle x_p, x_{c_t}, x_{c_{t+1}} \rangle, y_{p,k}, y_{c_t,i}, y_{c_{t+1},j})^T \end{pmatrix}^T. \tag{17}$$

Similarly, $\hat{F}^{p,k}_{c_1,i}$ is calculated as:

$$\hat{F}^{p,k}_{c_1,i} = \hat{F}^{c_1,i} + \left\langle \begin{pmatrix} \mathbf{\Psi}(x_p, y_{p,k})^T \\ \mathbf{0} \\ \mathbf{\Psi}(\langle x_p, x_{c_1} \rangle, y_{p,k}, y_{c_1,i})^T \\ \mathbf{0} \end{pmatrix}^T, \mathbf{w} \right\rangle. \tag{18}$$

With the above recursive property, the inference is conducted layer by layer starting from the bottom of the DOM tree. After the recursion is finished at the root of the DOM tree, the global optimal value of F is obtained.

Fig. 5 shows the recursive inference procedure for the nested region in Fig. 1(b). When inferring the labels of the subtrees rooted at <tr>'s, referring to Fig. 5(a), the optimal label for the <tr>'s is "REGION" and the optimal label for the <td>'s is "REC-S". Currently, the label "SUBREG" for <tr> is not the optimal choice. When coming to the



(a) Inference results for the subtrees rooted at <tr>'s. (b) Inference result for the nested region <table>.

Fig. 5. Recursive inference for the nested region in Fig. 1(b).

higher level <table>, referring to Fig. 5(b), the label of <tr>'s favors towards "SUBREG" and <table> tends to be labeled as "REGION".

Given that the inference of all lower level subtrees has been done, the time complexity of the higher level subtree inference is $\mathbf{O}(|\mathcal{Y}|^3 * T')$ where T' is the child number of this subtree. Therefore, the time complexity of the entire DOM tree inference can be decomposed into those of all child sequences in any layer of the DOM tree. Let T_i denote the length of a particular child sequence, the overall time complexity for the entire DOM tree is computed as $\mathbf{O}(\sum_i |\mathcal{Y}|^3 * T_i) = \mathbf{O}(|\mathcal{Y}|^3 * |\mathbf{x}|)$ where $|\mathbf{x}|$ denotes the total number of nodes in the DOM x.

4.3. Backtracking for the Optimal Label Assignment

The backtracking of the optimal label for the entire DOM tree is carried out with a top-down manner starting from the root of the tree. The best label of the root node is obtained by:

$$y_{p,*} = \underset{y_{p,k} \in \mathbf{y}_p}{\operatorname{argmax}} \hat{F}^{p,k}. \tag{19}$$

Let $y_{c_t,*}$ denote the label of x_{c_t} in the optimal label assignment for the child sequence with the root node x_p labeled with $y_{p,*}$. Thus, the optimal label of the last child x_{c_T} is:

$$y_{c_T,*} = \underset{y_{c_T,i} \in \mathbf{y}_{c_T}}{\operatorname{argmax}} \hat{F}^{p,*}_{c_T,i}. \tag{20}$$

By backtracking, $y_{c_t,*}$ is obtained as:

$$y_{c_t,*} = \underset{y_{c_t,i} \in \mathbf{y}_{c_t}}{\operatorname{argmax}} \{\hat{F}^{p,*}_{c_t,i}. + \Delta F^{p,*}_{(c_t,i),(c_{t+1},*)}\} \tag{21}$$

After the optimal label $y_{c_t,*}$ of each x_{c_t} is obtained, we can repeat the same procedure as given in Equations 20 and 21 to obtain the optimal labels for the children of x_{c_t}.

The above backtracking can be implemented by keeping backward pointers during the recursive calculation of the global optimal objective value. Thus, given the optimal label $y_{p,*}$ of the DOM root, we can obtain the optimal label assignment for the entire DOM tree by backtracking through backward pointers in a top-down manner.

4.4. Second Optimal Label Assignment

Owning to the fact that the feature weight estimation is conducted with a maximum margin principle based algorithm, the second optimal label assignment is needed when maximizing the margin.

4.4.1. Second Optimal Inference for Bottom Subtrees. Returning to the example in Fig. 4, recall that its optimal label assignment is denoted as $(y_{p,*}, y_{c_1,*}, y_{c_2,*}, \cdots, y_{c_T,*})$. The

second optimal objective value is achieved by one of the following cases: (1) The label of x_p is not $y_{p,*}$ and the labels of the children also change accordingly. Let $\ddot{F}^{p,\bar{*}}$ denote the achieved value in this case; (2) The label of x_p is still $y_{p,*}$, but the label sequence of the children is different from $(y_{c_1,*}, y_{c_2,*}, \cdots, y_{c_T,*})$. Let $\ddot{F}^{p,*}$ denote the achieved value in this case. Then the second optimal objective value is obtained by:

$$\ddot{F} = \max\{\ddot{F}^{p,\bar{*}}, \ddot{F}^{p,*}\}. \tag{22}$$

$\ddot{F}^{p,\bar{*}}$ and $\ddot{F}^{p,*}$ are calculated by Equations 23 and 24:

$$\ddot{F}^{p,\bar{*}} = \max_{y_{p,k} \in \mathbf{y}_p \setminus y_{p,*}} \hat{F}^{p,k}, \tag{23}$$

$$\ddot{F}^{p,*} = \max\{\max_{y_{c_T,i} \in \mathbf{y}_{c_T} \setminus y_{c_T,*}} \hat{F}^{p,*}_{c_T,i}\ ,\ \ddot{F}^{p,*}_{c_T,*}\}. \tag{24}$$

The first term of Equation 24 is the best value achieved by labeling the last child with another label other than $y_{c_T,*}$. The second term of Equation 24 denotes the second optimal value achieved by still labeling the last child with $y_{c_T,*}$, which can be recursively calculated. Let $\ddot{F}^{p,*}_{c_{t:T},*}$ denote the second optimal value achieved by labeling the child sequence from t to T with the optimal labels $(y_{c_t,*}, \cdots, y_{c_T,*})$. $\ddot{F}^{p,*}_{c_{t:T},*}$ is recursively calculated as:

$$\ddot{F}^{p,*}_{c_{t:T},*} = \max\{\ddot{F}^{p,*}_{c_{t-1:T},*}\ ,\ \max_{y_{c_{t-1},i} \in \mathbf{y}_{c_{t-1}} \setminus y_{c_{t-1},*}} \{\hat{F}^{p,*}_{c_{t-1},i} + \Delta F^{p,*}_{(c_{t-1},i),(c_t,*)}\} + \Delta F^{p,*}_{c_{t:T},*}\}, \tag{25}$$

where the second term is composed of two parts, namely, the partial second optimal value achieved up to the child x_{c_t} which is labeled with $y_{c_t,*}$, and the objective value increment $\Delta F^{p,*}_{c_{t:T},*}$ achieved by the optimal labels $(y_{c_t,*}, \cdots, y_{c_T,*})$. $\Delta F^{p,*}_{c_{t:T},*}$ is calculated as:

$$\Delta F^{p,*}_{c_{t:T},*} = \sum_{t'=t}^{T-1} \Delta F^{p,*}_{(c_{t'},*),(c_{t'+1},*)}. \tag{26}$$

Since we assume that each child node is a leaf, the termination condition of the recursion in Equation 25 is $t = 2$, i.e., the label of the first child changes and the remaining children's labels do not change:

$$\ddot{F}^{p,*}_{c_{2:T},*} = \max_{y_{c_1,i} \in \mathbf{y}_{c_1} \setminus y_{c_1,*}} \{\hat{F}^{p,*}_{c_1,i} + \Delta F^{p,*}_{(c_1,i),(c_2,*)}\} + \Delta F^{p,*}_{c_{2:T},*}. \tag{27}$$

4.4.2. Second Optimal Inference for Higher Subtrees. When processing the higher level subtrees, we can take the lower level subtrees rooted at each child x_{c_t} into consideration. Thus, the global second optimal value may be achieved in the case that all labels of x_p and its children x_{c_t}'s are still the same as those for the global optimal value, and the labels of some descendants of a certain x_{c_t} change. Following the above notation, this value is denoted as $\ddot{F}^{p,*}_{c_{1:T},*}$, which is calculated as:

$$\ddot{F}^{p,*}_{c_{1:T},*} = \max_{x_{c_t}} \{\hat{F} - \hat{F}^{c_t,*} + \ddot{F}^{c_t,*}\}, \tag{28}$$

where $\ddot{F}^{c_t,*}$ denotes the second optimal value achieved in labeling the subtree rooted at x_{c_t} which is still labeled with $y_{c_t,*}$. $\ddot{F}^{c_t,*}$ can be calculated by Equation 24. Finally, the global second optimal objective value $\ddot{F}$ is calculated as:

$$\ddot{F} = \max\{\ddot{F}^{p,\bar{*}}, \ddot{F}^{p,*}, \ddot{F}^{p,*}_{c_{1:T},*}\}. \tag{29}$$

ALGORITHM 1: Finding feature weights via structured output SVM learning.

1: **initialization**: $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, C, ε, $\forall i : S_i \leftarrow \emptyset$
2: **repeat**
3: **for** $i = 1, \cdots, n$ **do**
4: $H(\mathbf{y}) \equiv (1 - \langle \delta\mathbf{\Psi}_i(\mathbf{y}), \mathbf{w}\rangle)\Delta(\mathbf{y}_i, \mathbf{y})$ *//cost function*
5: $\mathbf{y}^* = \text{argmax}_{\mathbf{y} \in \mathcal{Y}^{|\mathbf{x}_i|}} H(\mathbf{y})$ *//cutting plane*
6: $\xi_i = \max\{0, \max_{\mathbf{y} \in S_i} H(\mathbf{y})\}$
7: **if** $H(\mathbf{y}^*) > \xi_i + \varepsilon$ **then**
8: $S_i \leftarrow S_i \cup \{\mathbf{y}^*\}$
9: update α's and $\mathbf{w}$ with $\cup_i S_i$
10: **end if**
11: **end for**
12: **until** no S_i has changed during iteration

Thus, $\ddot{F}$ can be obtained with a recursive manner starting from the bottom of the DOM tree. Similarly, backward pointers are kept for backtracking the second optimal label assignment.

5. STATISTICAL KNOWLEDGE ACQUISITION

In this section, we discuss the determination of the feature weights of the statistical knowledge using a development data set via a parameter estimation algorithm based on structured output Support Vector Machine (SVM) model [Tsochantaridis et al. 2005]. This model can tackle the inter-dependency among the labels on the nodes of the DOM structure. Meanwhile, the maximum margin principle of SVM allows the determined feature weights a better generalization capability to harness the heterogeneity of Web content.

5.1. Finding Feature Weights via Structured Output SVM Learning

Let $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ denote a set of development data instances, and $\Delta(\mathbf{y}_i, \mathbf{y})$ denote the loss caused by assigning the label assignment $\mathbf{y}$ to $\mathbf{x}_i$. The quadratic program form of the SVM model with slack re-scaled by the loss is:

$$\min_{\mathbf{w},\xi} \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$

$$s.t.\ \forall i, \forall \mathbf{y} \in \mathcal{Y}^{|\mathbf{x}_i|} \setminus \mathbf{y}_i : \langle \delta\mathbf{\Psi}_i(\mathbf{y}), \mathbf{w}\rangle \geq 1 - \frac{\xi_i}{\Delta(\mathbf{y}_i, \mathbf{y})}, \tag{30}$$

where ξ_i is the slack variable of $\mathbf{x}_i$, $C > 0$ is a tradeoff constant of the two parts, and $\langle \delta\mathbf{\Psi}_i(\mathbf{y}), \mathbf{w}\rangle = F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}) - F(\mathbf{x}_i, \mathbf{y}; \mathbf{w})$ is the margin between the objective values of $\mathbf{y}_i$ and $\mathbf{y}$. Tsochantaridis et al. proposed a cutting plane based algorithm to solve this optimization problem in its dual formulation [Tsochantaridis et al. 2005]. It selects a subset of constraints from the exponential full set $\mathcal{Y}^{|\mathbf{x}_i|}$ to ensure a sufficiently accurate solution. The procedure of finding the feature weights is briefly summarized in Algorithm 1. S_i is the working set of selected constraints for the instance $\mathbf{x}_i$, α's are the Lagrange multipliers, and ε is the precision parameter. The algorithm proceeds by finding the most violated constraint for $\mathbf{x}_i$ involving $\mathbf{y}^*$ (refer to Line 5). If the margin violation of this constraint exceeds the current ξ_i by more than ε (refer to Line 7), the working set S_i of $\mathbf{x}_i$ is updated. α's and $\mathbf{w}$ are also updated with the updated working set accordingly. We refer the reader to [Tsochantaridis et al. 2005] for more details of the algorithm.

5.2. Region-oriented Loss

In record region detection and data record extraction, we wish to avoid the missing of data record regions since it will result in false negative predictions on all data records in this region. As observed in the existing works [Simon and Lausen 2005], one major difficulty of record region detection is to find out all the regions in a given Web page. Some existing methods only reported the largest region and missed the others [Zhao et al. 2005]. To tackle this issue, we define a loss function that penalizes the missing of record regions.

Let $\mathbf{x}_{reg}$ denote the set of root nodes of the subtrees corresponding to record regions in a Web page, the loss function is defined as:

$$\Delta(\mathbf{y}_i, \mathbf{y}) \equiv \exp\{\frac{\sum_{x \in \mathbf{x}_{reg}} \bar{\delta}(\mathbf{y}_i(x), \mathbf{y}(x))}{|\mathbf{x}_{reg}|}\}, \tag{31}$$

where $\bar{\delta}$ is an indicator function which has the value 0 if $\mathbf{y}_i(x) = \mathbf{y}(x)$ and the value 1 otherwise. The loss function Δ is monotonically increasing with respect to the number of wrongly labeled regions. Note that we do not adopt zero diagonal loss function. The reason is that our loss function does not consider all the nodes of a DOM tree and it concentrates on the essential parts, namely, record regions.

5.3. Cost Function Optimization

In the learning procedure as depicted in Algorithm 1, it is required to optimize the cost function in Line 4 for finding the most violated constraint corresponding to $\mathbf{y}^*$:

$$\mathbf{y}^* = \underset{\mathbf{y} \in \mathcal{Y}^{|\mathbf{x}_i|}}{\operatorname{argmax}} H(\mathbf{y}). \tag{32}$$

When $H(\mathbf{y}^*) \leq 0$, $\mathbf{y}^*$ will not be added into the working set S_i since the margin $\langle \delta\mathbf{\Psi}_i(\mathbf{y}^*), \mathbf{w} \rangle$ is larger than or equal to 1 (refer to Line 7). Thus, we only need to consider the cases when $H(\mathbf{y}^*) > 0$, i.e., $\langle \delta\mathbf{\Psi}_i(\mathbf{y}^*), \mathbf{w} \rangle < 1$.

We first conduct inference for $\mathbf{x}_i$ based on the current $\mathbf{w}$. Let $\hat{\mathbf{y}}$ denote the label assignment that achieves the optimal objective value $F(\mathbf{x}_i, \hat{\mathbf{y}}; \mathbf{w})$ denoted as $F_i(\hat{\mathbf{y}})$ for short. If $\langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle \geq 1$, we directly move to $\mathbf{x}_{i+1}$. The reason is that given $F_i(\hat{\mathbf{y}}) \geq F_i(\mathbf{y}^*)$, $\langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle = F_i(\mathbf{y}_i) - F_i(\hat{\mathbf{y}})$ and $\langle \delta\mathbf{\Psi}_i(\mathbf{y}^*), \mathbf{w} \rangle = F_i(\mathbf{y}_i) - F_i(\mathbf{y}^*)$, we have $\langle \delta\mathbf{\Psi}_i(\mathbf{y}^*), \mathbf{w} \rangle \geq \langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle \geq 1$. If $\langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle < 1$, we utilize $\hat{\mathbf{y}}$ to derive $\mathbf{y}^*$.

PROPOSITION 1. *If all regions in $\mathbf{x}_{reg}$ are wrongly labeled in $\hat{\mathbf{y}}$ and given $\langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle < 1$, then we have $\mathbf{y}^* = \hat{\mathbf{y}}$.*

PROOF. Note that the loss function Δ is monotonically increasing with respect to the number of wrongly labeled regions. Then $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$ is maximized when all regions in $\mathbf{x}_{reg}$ are wrongly labeled. In addition, $\hat{\mathbf{y}}$ maximizes F so that $1 - \langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle$ is also maximized. Taking $\Delta(\mathbf{y}_i, \hat{\mathbf{y}}) > 0$ and $\langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle < 1$ into consideration, $H(\hat{\mathbf{y}})$ is the maximum. □

PROPOSITION 2. *Let $\mathbf{y}'$ be a label assignment that has less than or equal number of wrongly labeled regions than $\hat{\mathbf{y}}$ and given $\langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle < 1$, then we have $H(\mathbf{y}') \leq H(\hat{\mathbf{y}})$.*

PROOF. Because $\hat{\mathbf{y}}$ maximizes F, $F_i(\mathbf{y}') \leq F_i(\hat{\mathbf{y}})$. Thus, we have $(1 - \langle \delta\mathbf{\Psi}_i(\mathbf{y}'), \mathbf{w} \rangle) \leq (1 - \langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle)$. In addition, $0 < \Delta(\mathbf{y}_i, \mathbf{y}') \leq \Delta(\mathbf{y}_i, \hat{\mathbf{y}})$ and $(1 - \langle \delta\mathbf{\Psi}_i(\hat{\mathbf{y}}), \mathbf{w} \rangle) > 0$, therefore $H(\mathbf{y}') \leq H(\hat{\mathbf{y}})$. □

Based on Proposition 2, we only need to optimize $H(\mathbf{y})$ among the label assignments that have more wrongly labeled regions than $\hat{\mathbf{y}}$, at the same time satisfying $\langle \delta\mathbf{\Psi}_i(\mathbf{y}), \mathbf{w} \rangle < 1$. Let $\mathbf{x}^{\times}_{reg}$ denote the regions labeled wrongly in $\hat{\mathbf{y}}$. The procedure of deriving $\mathbf{y}^*$ is summarized as Proposition 3.

ALGORITHM 2: Assembling intertwined data records.

1: **input**: data record $\mathbf{R} = \{\mathcal{S}_{i..j}\}$,
2: where $i < j$ for each $\mathcal{S}_{i..j}$
3: **output**: intertwined data records $\mathbf{R}'$
4: $c \leftarrow 0$, $\mathbf{R}' \leftarrow \emptyset$
5: **for** $\mathcal{S}_{i..j} \in \mathbf{R}$ **do**
6: **if** $is\_composite\_record(\mathcal{S}_{i..j})$ **then**
7: $c \leftarrow c + 1$
8: **end if**
9: **end for**
10: **if** $c/|\mathbf{R}| \geq \theta'$ **then**
11: **for** $\mathcal{S}_{i..j} \in \mathbf{R}$ **do**
12: $\mathbf{S}_k \leftarrow$ subtrees of $\mathcal{S}_k$, where $i \leq k \leq j$
13: $\mathcal{S}_l^k$ denotes the l-th subtree of $\mathcal{S}_k$
14: $size \leftarrow \max_{k=i}^{j} |\mathbf{S}_k|$
15: **for** $l = 1, \cdots, size$ **do**
16: $\mathbf{R}' \leftarrow \mathbf{R}' \cup \{\mathcal{S}_l^i \ldots \mathcal{S}_l^j\}$
17: **end for**
18: **end for**
19: **end if**

1: ***proc*** $is\_composite\_record(\mathcal{S}_{i..j})$
2: $s_1 \leftarrow 0$, $s_2 \leftarrow 0$, $c_1 \leftarrow 0$, $c_2 \leftarrow 0$
3: $\mathbf{S}_k \leftarrow$ subtrees of $\mathcal{S}_k$, where $i \leq k \leq j$
4: **for** $k = i, \cdots, j$ **do**
5: **for** $\mathcal{S}_l^k, \mathcal{S}_r^k \in \mathbf{S}_k$ **and** $l \neq r$ **do**
6: $s_1 \leftarrow s_1 + sim(\mathcal{S}_l^k, \mathcal{S}_r^k)$
7: $c_1 \leftarrow c_1 + 1$
8: **end for**
9: **for** $t = i + 1, \cdots, j$ **do**
10: **for** $\mathcal{S}_l^k \in \mathbf{S}_k$ **and** $\mathcal{S}_r^t \in \mathbf{S}_t$ **do**
11: $s_2 \leftarrow s_2 + sim(\mathcal{S}_l^k, \mathcal{S}_r^t)$
12: $c_2 \leftarrow c_2 + 1$
13: **end for**
14: **end for**
15: **end for**
16: **if** $s_1/c_1 \geq \theta$ and $s_2/c_2 < \theta$ **then**
17: **return true**
18: **else**
19: **return false**
20: **end if**

PROPOSITION 3. *Let* $\{\mathbf{x}'_{reg}\}$ *be all subsets of* $\mathbf{x}_{reg}$ *having more elements than* $\mathbf{x}^{\times}_{reg}$*, and let* $\mathbf{y}'$ *be the label assignment that achieves the largest* F *value when all regions in* $\mathbf{x}'_{reg}$ *are wrongly labeled and all regions out of* $\mathbf{x}'_{reg}$ *are correctly labeled. The label* $\mathbf{y}^*$ *maximizing* $H(\mathbf{y})$ *is from* $\cup_{\mathbf{x}'_{reg}}\{\mathbf{y}'\} \cup \{\hat{\mathbf{y}}\}$.

We can first fix the loss value to a constant by selecting an $\mathbf{x}'_{reg}$ and removing the label REGION from the candidate label set of the regions in $\mathbf{x}'_{reg}$, meanwhile the label REGION is pre-assigned to the other regions out of $\mathbf{x}'_{reg}$. Then, the term $(1 - \langle \delta\mathbf{\Psi}_i(\mathbf{y}), \mathbf{w}\rangle)$ in H is maximized with the inference algorithm and $\mathbf{y}'$ is obtained. If $\langle \delta\mathbf{\Psi}_i(\mathbf{y}'), \mathbf{w}\rangle \geq 1$, $\mathbf{y}'$ is pruned. Otherwise, $\mathbf{y}'$ is one candidate for finding $\mathbf{y}^*$. The same procedure is repeated for all the other $\mathbf{x}'_{reg}$. Typically, the number of regions in a page is limited, normally, no more than 5. So the above enumeration method can work in affordable time in practice. To save the computational time, some existing intermediate results in the computation of $\hat{\mathbf{y}}$ can be reused.

Note that it is possible that $\hat{\mathbf{y}}$ is the same as $\mathbf{y}_i$. To tackle this issue, we infer the second best label assignment and use it instead of $\hat{\mathbf{y}}$ to go through the same procedure as above.

6. ASSEMBLING INTERTWINED RECORDS

Intertwined records, also known as *non-continuous records* in DEPTA [Zhai and Liu 2006] and *cross records* in [Zheng et al. 2009], refer to records whose attributes intertwine together with other records' attributes. One example and its DOM tree are given in Figs. 2(b) and 2(d) respectively. Each record has 3 attributes, namely, image, title, and price, and these attributes are scattered in 3 successive <tr>'s, such as $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$. Such subtree group, e.g., $\mathcal{S}_{1..3}$, is named *composite record*.

Our Skoga framework described above is able to detect the composite records together with the complicated flat records using a global analysis. To determine and assemble possible intertwined records, Skoga invokes the method depicted in Algorithm 2. Let $\mathbf{R} = \{\mathcal{S}_{i..j}\}$ denote such a set of data records from the same record region. We first judge whether $\mathbf{R}$ is a set of composite data records. In Line 6 of the main proce-

dure on the left-hand side, each $\mathcal{S}_{i..j}$ is passed to the sub-procedure $is\_composite\_record$ depicted on the right-hand side to determine whether it is a composite record. Referring to Line 16 of $is\_composite\_record$, if the average similarity s_1/c_1, calculated as in Lines 5 to 8, of the subtree pairs from a single $\mathcal{S}_k$ ($i \leq k \leq j$) is greater than or equal to a threshold θ and the average similarity s_2/c_2, calculated as in Lines 9 to 14, of the subtree pairs with two subtrees from $\mathcal{S}_k$ and $\mathcal{S}_t$ ($k \neq t$) is less than θ, we are confident to determine that $\mathcal{S}_{i..j}$ is a composite record. If the ratio of composite records in $|\mathbf{R}|$ is no less than a ratio threshold θ', as shown in Line 10 of the main procedure, we conclude that $\mathbf{R}$ is a composite record set. Then, the intertwined records can be easily assembled from the composite records, as shown in Lines 11 to 18 of the main procedure. Returning to the example given in Figs. 2(b) and 2(d), two records (image 1, title 1, price 1) and (image 2, title 2, price 2) are assembled from the first three <tr>'s.

7. EXPERIMENTS

7.1. Evaluation Data Sets

In our experiments, four evaluation data sets are used to evaluate the performance of the proposed Skoga framework. The first data set is the testbed collected by Yamada et al. [Yamada et al. 2004]. It can be considered as a benchmark data set [1] which has been used for evaluation in some works such as ViPER [Simon and Lausen 2005] and TPC [Miao et al. 2009]. It is referred to as TB1 in this paper. TB1 has 253 Web pages from 51 Web sites randomly drawn from 114,540 Web pages with search forms of various search engines, such as picture search, product search and document search. One sample data record of each record region in the pages was given by the collectors of this data set. Almost all pages contain flat regions or complicated flat regions. In our experiment, two sites were excluded because of garbled code or ambiguous record annotation. Thus, we used the remaining 49 sites including 243 pages and 4,326 data records in total.

Different from TB1, in which the records are search results in dynamic Web pages generated by server-side programs with predefined templates, the second data set is composed of static Web pages in which the data records are presented in formats exhibiting some regularities. This data set is referred to as TB2 [2]. The pages were collected from different online shopping and university Web sites. The targeted university pages are the ones that contain the faculty list of a particular department. To obtain such kind of pages, we issued a synthetic query in the form of "faculty list site:xxx.edu" to Google and then browsed the result pages to find the ones with record regions. To collect the shopping Web pages, we investigated the online shopping Web sites one by one in an online shopping yellow page `http://www.usaonlineshoppingguide.com/`. There are 37 categories such as "Art Collectibles" and "Baby Stores" in this yellow page, and each category has 10 recommended shopping Web sites on average. We clicked the navigation links in the randomly selected sites to obtain record pages and at most 2 pages were collected from a single site. This data set contains 200 pages and 5,713 data records in total.

We prepared the third data set to examine the performance of Skoga on complicated flat regions and intertwined regions. The pages with flat and complicated flat regions were collected from different online shopping sites and university sites. The pages with intertwined regions were only collected from online shopping sites since very few university Web sites adopt the intertwined manner in presenting the faculty information. This data set was collected in the same manner as above and it is referred to as TB3 [3].

[1] It is publicly available at `http://daisen.cc.kyushu-u.ac.jp/TBDW/`.
[2] It is publicly available at `http://www.se.cuhk.edu.hk/~textmine/`.
[3] It is publicly available at `http://www.se.cuhk.edu.hk/~textmine/`.

TB3 contains 100 pages and 2,158 data records in total. There are 50 pages containing some product lists and at most 5 pages were collected from a single shopping site. The other pages were collected from department sites of different universities.

The last data set contains data records of user-generated content such as reader comments, customer reviews, and forum posts. The pages were collected from news channels, such as BBC, CNN, ABC, etc., online shopping sites, such as Amazon, eBay, AliExpress, etc., and online forums, such as forums.asp.net, forums.d2jsp.org, forums.phpfreaks.com, etc. The posts or reviews are written by the users and embedded in the predefined templates of the corresponding Web sites. We regard each of them as one data record and test whether Skoga framework can accurately extract them. In total, this data set contains 100 pages and 2,318 data records coming from about 30 different Web sites and at most 4 pages were collected from a single site. This data is referred to as TB4.

7.2. Experimental Setup

One of the comparison methods is MDR [Liu et al. 2003] [4] which is able to deal with flat, nested, and intertwined records. DEPTA [Zhai and Liu 2006] is another comparison method in our experiment. DEPTA employs some rendering information to construct the DOM tree and uses tree edit distance instead of string edit distance in the calculation of generalized node similarity. Since no implementation of this method is available, we implemented this method by following the pseudo-code presented in [Zhai and Liu 2006]. To enhance it so that it can take the advantage of the development data set, we develop a parameter estimation process for automatically determining a major model parameter on top of the basic DEPTA algorithm using the development data set. Specifically, the basic DEPTA algorithm has one major parameter that affects the performance of record detection, namely, the threshold τ of the normalized tree distance between generalized nodes. Basically, we varied this parameter and determined the parameter value that achieves the highest performance in the development data set. The found value for τ was 0.36. Considering the heterogeneity of the record regions, the maximum generalized node length used was 12 in our experiments and it is lager than the value 10 in [Zhai and Liu 2006] to obtain better performance although it increases the running time of the program. This enhanced DEPTA is referred to as DEPTA+ in this paper. The authors of FiVaTech [Kayed and Chang 2010] kindly provided us the demo system. Therefore, we can also conduct comparison with FiVaTech on all the evaluation data sets except TB3 since this method is not designed to tackle intertwined record regions. We also compared with another existing method, namely, TPC [Miao et al. 2009] by simply retrieving the experimental results of TPC available on TB1 data set we used.

We employ the commonly used precision, recall, and F-measure as the evaluation metrics. They are calculated as follows:

$$precision = \frac{|\{\text{true data records}\} \cap \{\text{identified data records}\}|}{|\{\text{identified data records}\}|}, \tag{33}$$

$$recall = \frac{|\{\text{true data records}\} \cap \{\text{identified data records}\}|}{|\{\text{true data records}\}|}, \tag{34}$$

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}. \tag{35}$$

[4] MDR is publicly available at `http://www.cs.uic.edu/~liub/WebDataExtraction/`.

Both micro-averaged and macro-averaged values are reported to provide a more comprehensive perspective on the performance. A micro-averaged value is computed by first aggregating the ground truth data records and the identified data records from all the pages. Then micro-averaged precision, recall, and F-measure are calculated based on these two global sets. In contrast, macro-averaged values are calculated by first calculating precision, recall, and F-measure for each Web page and finally taking the average values of all the pages.

As mentioned before, a separate development data set was collected to conduct the estimation of feature weights of statistical structure knowledge. It was also used to enhance the existing method DEPTA for comparison. This data set contains Web pages originated from different types of sources. Some pages come from the data set 3 in ViNTs [Zhao et al. 2005] [5]. This data set is originated from Omini [Buttler et al. 2001] testbed collected by Buttler et al. which consists of more than 2,000 Web pages collected from 50 Web sites. ViNTs took one random page per Web site to construct its data set 3. The pages in this set are mainly search result pages from different search engines including vertical search engines and general search engines. The regions in these pages are of flat type or complicated flat type. We also collected some other pages from the online shopping Web sites listed in another online shopping yellow page `http://www.toponlineshopping.com/`. There are 22 categories such as "Art & Collectibles" and "Beauty & Fragrances" in this yellow page, and each category has about 6 sub-categories on average. Under each sub-category, we arbitrarily selected 3 recommended Web sites. We directly clicked the navigation links in the home page to obtain data record pages. In this way, we collected 100 data record pages presenting some product lists. About half of these pages contain nested record regions, and the other pages contain other types of data record regions.

A Firefox plug-in program was developed to assist the annotators in preparing the development data instances from the pages in the development data set. The annotators appended the labels as attributes to the DOM nodes inside record regions with the plug-in program. Take the nested region in Fig. 1(d) as an example, after the annotation, each <td> node becomes the form of "<td label='REC-S'>", each <tr> node becomes the form of "<tr label='SUBREG'>", etc.

The parameters C and ε in the feature weight estimation algorithm depicted in Algorithm 1 were set to be 5 and 0.5 respectively. The similarity threshold for counting the occurrence in the parent-children features in Section 3.2 was set to be 0.6 and the similarity measure used is the same as the one proposed in RST [Bing et al. 2011]. In Algorithm 2, the average similarity threshold θ is set to be 0.65 and the ratio threshold θ' is set to be 80%. We ran MDR on the evaluation data sets with the default similarity threshold 60% and extracted the records reported.

7.3. Experimental Results on TB1

TPC [Miao et al. 2009] also conducted experiment on this data set. The authors kindly provided us the Web site IDs in the subset they used, which contains 43 sites out of 49 sites we use. Therefore, without implementing their method, we can still conduct a fair comparison. MDR could not produce output for two Web sites because the MDR program terminated abnormally. We report the results without these two sites.

The experimental results on TB1 are given in Table II. "MDR handled" refers to the subset that MDR program can handle, "TPC reported" refers to the subset used by the TPC method reported in [Miao et al. 2009], and "ALL" refers to the entire set of pages. "Ground" denotes the number of ground truth records. "TP" denotes the number of true positives detected by a method, and "FP" denotes the number of false positives.

[5] It is publicly available at `http://www.data.binghamton.edu:8080/vints/`.

Table II. Experimental results on TB1.

		Ground	TP	FP	P-mi	R-mi	F-mi	P-ma	R-ma	F-ma
MDR handled	MDR	4261	2692	230	0.920	0.640	0.754	0.620	0.636	0.622
	Skoga	4261	4226	27	0.994	0.992	0.993	0.991	0.979	0.983
TPC reported	TPC	3897	NA	NA	NA	NA	NA	0.904	0.931	NA
	Skoga	3897	3873	27	0.993	0.994	0.993	0.989	0.985	0.983
ALL	DEPTA+	4326	3550	704	0.835	0.823	0.829	0.788	0.807	0.802
	FiVaTech	4326	3782	340	0.918	0.874	0.895	0.883	0.889	0.871
	Skoga	4326	4289	49	0.989	0.991	0.990	0.989	0.976	0.975

"P-mi", "R-mi", "F-mi", "P-ma", "R-ma" and "F-ma" are the micro-averaged and macro-averaged precision, recall, and F-measure values respectively. "NA" means that the corresponding result was not reported in [Miao et al. 2009]. In P-ma calculation, if both TP and FP are 0 for a particular page, its precision is set to be 0.

Skoga outperforms MDR, DEPTA+, FiVaTech, and TPC. The recall of Skoga is significantly better than that of MDR. Compared with DEPTA+, Skoga achieves 15% to 20% improvement in both precision and recall. Compared with FiVaTech, the improvements achieved are 7% to 12%. In addition, the precision and recall of Skoga outperform TPC about 8% and 5% respectively. For MDR, the macro-precision is much smaller than the micro-precision. It is because for some pages, although the MDR program terminated normally, it could not give any output. Thus, both true positive and false positive are 0.

The details of the extracted records on TB1 are given in Fig. 6. Skoga produces some false positives for the sites 14 and 21. After checking the pages manually, we found that each page in the site 14 contains several record regions presenting the search results from different sources. The given ground truth by the collectors of TB1 only contains two regions, and regards the others as non-record regions. Some pages in the site 21 contain several recommended books on the right side bar. These books were not regarded as data records in the given ground truth since they are more likely to be advertisement items. Only the books formatted with <table> in the center of the page were included in the ground truth. For this site, MDR only outputs the books annotated. Skoga missed a few results in the sites 14, 18, 23, and 25. The main reason is that some regions adopt a different formatting manner in the first few data records compared with the remaining records. For example, the heading information is included in the first data record. Consequently, this record is identified as `REGNOT`. For the site 36, we find that each field of a record is packed in a single subtree, such as id, title, URL, each of digest sentences, etc. Since different records have different number of subtrees, both DEPTA+ and FiVaTech output some false positives. DEPTA+ wrongly outputs many records in some sites such as 19, 23, etc. It is mainly because DEPTA+ has limitations in region detection and record identification brought in by the constraints on the length of generalized node.

7.4. Experimental Results on TB2

On data set TB2, we conducted comparison with MDR, DEPTA+, and FiVaTech. MDR could not produce output for 13 pages because the MDR program terminated abnormally. The experimental results on TB2 are given in Table III. The headers of the rows and columns have the same meaning as those in Table II. The recall of Skoga is significantly better than that of MDR. The reason is that for quite a few pages the MDR program could not give any output, although it terminated normally. Compared with DEPTA+, Skoga achieves 14% to 21% improvements in both precision and recall, and about 18% improvements in both micro and macro F-measure values. Compared with FiVaTech, improvements achieved are about 6% to 9%.

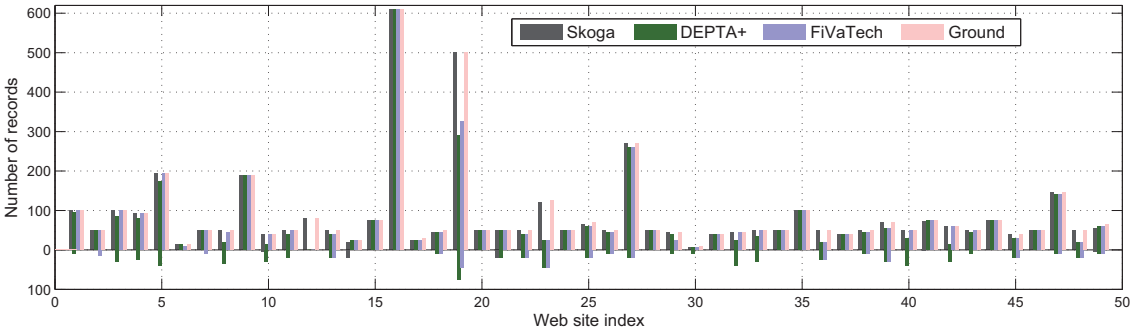


Fig. 6. Extracted records for each site in TB1. TP number and FP number are shown by the bars above and below the axis respectively.

Table III. Experimental results on TB2.

		Ground	TP	FP	P-mi	R-mi	F-mi	P-ma	R-ma	F-ma
MDR handled	MDR	5364	3451	832	0.806	0.643	0.715	0.629	0.637	0.634
	Skoga	5364	4893	441	0.917	0.912	0.915	0.902	0.896	0.897
ALL	DEPTA+	5713	4503	1854	0.708	0.788	0.746	0.697	0.767	0.731
	FiVaTech	5713	4712	804	0.854	0.825	0.840	0.843	0.832	0.834
	Skoga	5713	5325	472	0.919	0.932	0.925	0.906	0.914	0.908

When tackling nested record regions, DEPTA+ first identifies the record region by regarding each subregion, such as <tr>'s in Fig. 1(d), as a generalized node. In the record identifying step of DEPTA+, it attempts to identify the data records by finding lower level generalized nodes from each generalized node in the detected region. The performance of DEPTA+ in this record identifying step is affected by three difficulties. First, if the similarity among the records in the subregion is lower than the required similarity threshold, the entire subregion is regarded as one data record. The second difficulty is caused by the lack of a global analysis. Precisely, the record identification from one subregion does not consider the identification results from other subregions. Therefore, the identification results from different subregions can be significantly different. The third difficulty for DEPTA+ is that, if the neighboring records in one subregion are separated by separator subtrees, DEPTA+ cannot identify the records accurately. Regarding FiVaTech, after checking the pages that cannot be well tackled, we found that FiVaTech suffers from the difficulty brought in by the optional tags in data record templates. FiVaTech does not perform tree matching across multiple layers in peer node recognition so that it may not be able to induce effective wrappers when the templates have more variants.

In general, Skoga can better handle these difficulties with the global analysis that evaluates the entire nested region as a whole. However, the heterogeneity of Web data records also causes some failure cases for Skoga. In some pages from the university sites, the professors are grouped according to their titles and formatted with different formats in the same region. This causes some difficulty for Skoga which adopts a global analysis to favor the regions whose records follow similar formats.

7.5. Experimental Results on TB3

On data set TB3, we conducted comparison with MDR and DEPTA+. FiVaTech was not used for comparison since it is not designed to tackle intertwined record regions. MDR could not produce output for 5 pages because the MDR program terminated abnormally. The experimental results are given in Table IV. The headers of the rows and

Table IV. Experimental results on TB3.

		Ground	TP	FP	P-mi	R-mi	F-mi	P-ma	R-ma	F-ma
MDR handled	MDR	2049	1468	179	0.891	0.716	0.794	0.715	0.694	0.702
	Skoga	2049	1993	42	0.979	0.972	0.976	0.973	0.969	0.970
ALL	DEPTA+	2158	1853	457	0.802	0.859	0.827	0.796	0.833	0.811
	Skoga	2158	2099	54	0.975	0.973	0.974	0.981	0.969	0.972

Table V. Experimental results on TB4.

	Ground	TP	FP	P-mi	R-mi	F-mi	P-ma	R-ma	F-ma
MDR	2318	1809	266	0.872	0.780	0.824	0.857	0.768	0.816
DEPTA+	2318	1979	287	0.873	0.854	0.863	0.858	0.837	0.850
FiVaTech	2318	2056	335	0.860	0.887	0.873	0.859	0.842	0.853
Skoga	2318	2185	96	0.958	0.943	0.950	0.947	0.936	0.938

columns have the same meaning as those in Table II. Compared with DEPTA+, Skoga achieves 11% to 18% improvements in both precision and recall, and more than 14% improvements in both micro and macro F-measure values.

In the detection of intertwined data records, DEPTA+ first detects each intertwined field as one pseudo data record region. In reality, each record in the pseudo-region is a record field. Then it assembles the true data records from the adjacent pseudo-regions. However, the record region detection step in DEPTA+ would face a difficulty in detecting the desirable pseudo-regions in certain kinds of intertwined regions. Taking the region shown in Fig. 2(b) with its DOM given in Fig. 2(d) as an example, with the top-down searching detection manner DEPTA+ first identifies that the table is one record region and it has the generalized nodes $\mathcal{S}_{1..3}$, $\mathcal{S}_{4..6}$, etc. In the next step of identifying data records, DEPTA+ regards each generalized node, e.g., $\mathcal{S}_{1..3}$, as a data record but not a pseudo-region since the subtrees, i.e., $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, are dissimilar to one another. Skoga can tackle this type of intertwined regions well. It first identifies the composite records with the labels "`REC-B`" and "`REC-I`", such as $\mathcal{S}_{1..3}$ and $\mathcal{S}_{4..6}$. Then, the assembling algorithm assembles the true data records from each composite record based on the regularity that each single subtree, e.g., $\mathcal{S}_1$, has several repetitive fields and the neighboring subtrees, e.g., $\mathcal{S}_1$ and $\mathcal{S}_2$, have different repetitive fields. Finally, the correct records can be reassembled.

7.6. Experimental Results on TB4

On data set TB4, we conducted comparison with MDR, DEPTA+, and FiVaTech. The experimental results on TB4 are given in Table V. The headers of the rows and columns have the same meaning as those in Table II. In this data set, most of the data records are composed of a single subtree and the data items are embedded in predefined templates. In general, all methods can achieve good performance except MDR which cannot generate output for a few pages. Skoga outperforms DEPTA+ and FiVaTech by around 8% in micro and macro F-measure values. One major type of difficulty in this data set is caused by the quotations. The users usually quote the record of previous users when giving their own comments, reviews or posts. This behavior results in embedding format of some data records so that the dissimilarity among records is enlarged and the detection accuracy is affected. Another type of difficulty is the content length diversity of this type of user-generated content resulting in different number of <p>'s or 's for paragraphs. Skoga and FiVaTech can overcome the difficulty of repeating content by tandem repeat detection and set detection respectively. MDR and DEPTA+ are not able to tackle this difficulty properly.

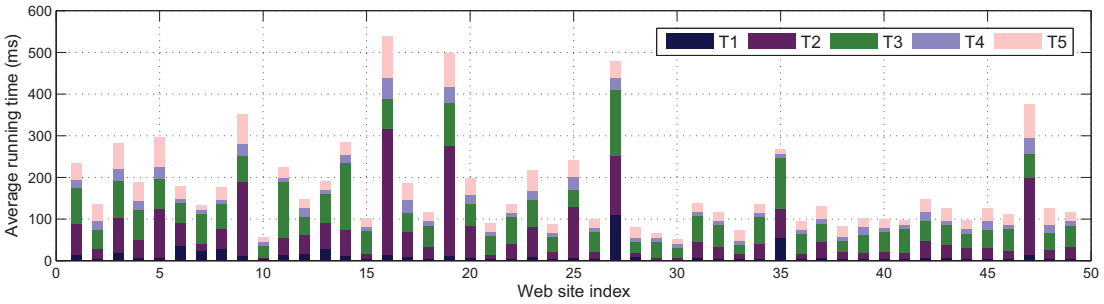


Fig. 7. The running time of Skoga for each site in TB1. The reported time for each site is the average time needed for processing its pages. T1 indicates the time for loading a Web page and building its DOM tree structure. T2 indicates the time for calculating feature values. T3 indicates the time for generating feature vectors and saving the testing instance of a page into the disk. T4 indicates the time for loading a testing instance from the disk and rebuilding tree structure. T5 indicates the time for inferring the optimal label assignment for a testing instance.

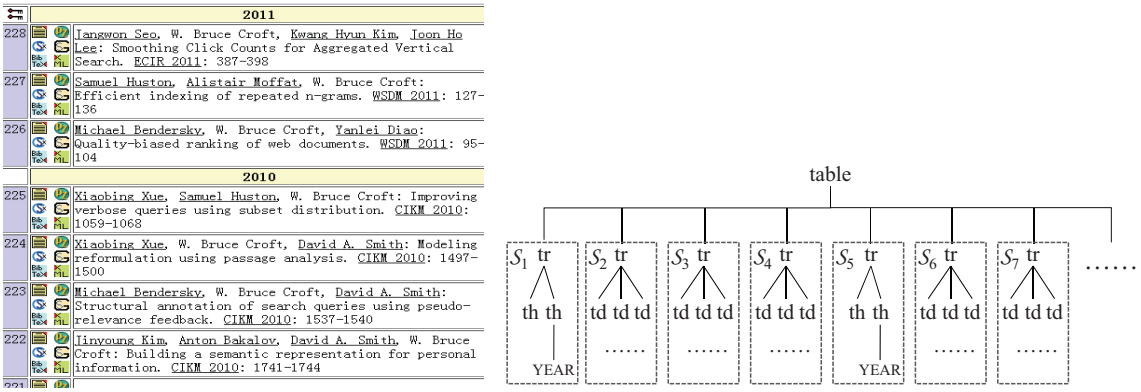
7.7. Running Time

The running time of Skoga framework for each site in TB1 is reported in Fig. 7. The reported time for each site is the average time needed for processing its pages. The implementation of the feature extraction part is in Java and the running time on different steps of this part is depicted by T1, T2 and T3. The inference part is implemented by extending the structured output SVM framework [Tsochantaridis et al. 2005] in C and the inference time is depicted by T4 and T5. The experiment is run on a PC with Quad core CPU @2.66 GHz and 3GB RAM. In fact, the program has one thread and consume relatively low memory, so it does not require powerful computing resource. From Fig. 7, it can be observed that Skoga is very efficient and one page takes around 200ms on average.

7.8. Empirical Case Study

An empirical case study on the page given in Fig. 8(a) with its DOM tree in 8(b) is presented to offer a close look at how our proposed model can effectively tackle this difficult case. The intermediate computational results in our model are also presented to show the procedure how the correct labels are assigned for this studied page. This case study can also provide explanations on the failure of existing methods. In this case study, each publication item should be regarded as one data record. The heading rows of the published year, namely, $\mathcal{S}_1$, $\mathcal{S}_5$, $\mathcal{S}_{20}$, etc., pose challenges for existing methods to conduct accurate record extraction.

Recall that DEPTA+ requires all the generalized nodes in the same region have the same length and they are all adjacent. Thus it recognizes that the first region is composed of several generalized nodes and each of which contains four table rows, such as $\mathcal{S}_{1..4}$, $\mathcal{S}_{5..8}$ and $\mathcal{S}_{9..12}$. In the record identifying step, DEPTA+ identifies data records from each single generalized node in the regions. The main idea is that if a generalized node contains two or more data records, one more iteration of finding finer generalized nodes in the current generalized node can find the data records. It assumes that the lower level finer generalized nodes, i.e., data records, need to satisfy the condition of covering all the data items in the original generalized node. This assumption makes DEPTA+ fail to detect the correct records from the first two generalized nodes. Precisely, the existence of $\mathcal{S}_1$ and $\mathcal{S}_5$ hinders the accurate detection of data records in the corresponding generalized nodes, namely, $\mathcal{S}_{1..4}$ and $\mathcal{S}_{5..8}$.



(a) A page fragment of noisy subtree region. (b) DOM tree of (a).

Fig. 8. Case study on a more challenging record region.

Table VI. Portion of intermediate computational results of Skoga for record detection from the Web page in Fig. 8(a) with the root node <table> labeled with REGION. L is the label ID, specifically, REC-S=3 and REGNOT=6. N is the subtree ID in Fig. 8(b).

L \ N	1	2	3	4	5	6	7	8	9	10	11
3	^{0}7.679 (5.194)	^{6}36.868 (29.156)	^{3}66.059 (29.153)	^{3}95.250 (29.156)	^{3}100.452 (5.194)	^{6}129.630 (29.156)	^{3}158.833 (29.158)	^{3}188.040 (29.173)	^{3}222.156 (34.082)	^{3}246.417 (24.227)	^{3}270.681 (24.229)
6	^{0}7.702 (5.218)	^{6}36.801 (29.180)	^{6}66.055 (29.172)	^{3}95.219 (29.167)	^{3}100.472 (5.218)	^{6}129.559 (29.178)	^{3}158.820 (29.170)	^{3}188.011 (29.183)	^{3}222.175 (34.141)	^{6}246.388 (24.217)	^{3}270.627 (24.216)

L \ N	12	13	14	15	16	17	18	19	20	21	22
3	^{3}294.947 (24.230)	^{3}319.214 (24.231)	^{3}348.401 (29.151)	^{3}372.673 (24.236)	^{3}400.330 (27.622)	^{3}429.528 (29.163)	^{3}453.779 (24.218)	^{3}489.429 (35.617)	^{6}494.646 (5.194)	^{6}518.949 (24.250)	^{3}543.215 (24.231)
6	^{3}294.896 (24.220)	^{3}319.162 (24.221)	^{3}348.378 (29.170)	^{3}372.621 (24.225)	^{3}400.294 (27.627)	^{3}429.508 (29.184)	^{3}453.735 (24.211)	^{3}489.409 (35.693)	^{3}494.699 (5.218)	^{6}518.934 (24.223)	^{3}543.165 (24.221)

L \ N	23	24	25	26	27	28	29	30	31	32	33
3	^{3}572.408 (29.157)	^{3}596.670 (24.227)	^{3}620.935 (24.229)	^{3}642.504 (21.536)	^{3}671.693 (29.156)	^{3}700.898 (29.171)	^{3}730.095 (29.162)	^{3}754.355 (24.226)	^{3}783.556 (29.166)	^{3}807.815 (24.225)	^{3}832.101 (24.253)

This case also hinders the accurate record detection of other existing methods such as FiVaTech and RST. FiVaTech identifies each group of publications in a particular year as one data record. The wrapper induction manner in FiVaTech favors a wrapper that can achieve a record recognition covering longer repeats. Thus, it induces a wrapper that treats each publication as one repetitive field in a detected "record", i.e. a group of publications in one year. Note that the output given by FiVaTech can also be regarded to be correct from a more general perspective on record definition. RST outputs the data records such as $\mathcal{S}_{1..2}$, $\mathcal{S}_3$, $\mathcal{S}_4$, $\mathcal{S}_{5..6}$, $\mathcal{S}_7$, etc. We can see that the heading rows of the published year are fused into the neighboring records and inaccurate data records are reported such as $\mathcal{S}_{1..2}$ and $\mathcal{S}_{5..6}$.

Skoga framework can consider the entire sequence of subtrees in Fig. 8(b) to conduct more accurate record detection with a global analysis. It identifies all data records correctly and also labels all heading rows of the published year with REGNOT. A portion of intermediate computational results of Skoga for record detection is given in Table VI. L is the label ID and N is the subtree ID in Fig. 8(b). The optimal label of the root node <table> is REGION. Following the notations in Section 4, we have $y_{p,*} =$ REGION. In each cell of the table, the superscript is the backtracking pointer, i.e. $y_{c_t,*}$, and the

superscript 0 indicates that the backtracking pointer is ineffective. The value after the superscript is the intermediate maximum objective value, i.e. $\hat{F}^{p,*}_{c_t,i}$. The value in the brackets is the optimal value for the corresponding subtree with the root labeled with the corresponding label, i.e. $\hat{F}^{c_t,i}$. Take the cell at $N = 3$ and $L = 3$ as an example, we have $y_{c_2,*} = 3$, $\hat{F}^{p,*}_{c_3,3} = 66.059$, and $\hat{F}^{c_3,3} = 29.153$. From the intermediate maximum objective values in the last column in the third section of Table VI, we can obtain that $y_{c_{33},*} = 3$. Thus, the optimal label sequence of the subtrees can be obtained by backtracking the pointers starting from $y_{c_{33},*}$. Finally, the subtrees $\mathcal{S}_1$, $\mathcal{S}_5$ and $\mathcal{S}_{20}$ corresponding to the published year rows are correctly labeled with `REGNOT`. The publication records are correctly labeled with `REC-S`.

8. RELATED WORK

The task of record-level extraction from an arbitrary single input page is one active direction in Web IE [Liu et al. 2003; Simon and Lausen 2005; Wang and Lochovsky 2003; Zhai and Liu 2006]. Such data record information is very useful for developing various applications such as online market intelligence [Baumgartner et al. 2009], knowledge base population [Bing et al. 2013], entity semantic network building [Luo et al. 2011], etc. Techniques that address record extraction from a single page can be categorized into the following types: early methods based on heuristics [Buttler et al. 2001; Embley et al. 1999b], repetitive pattern based methods [Chang and Lui 2001; Wang and Lochovsky 2003], similarity-based extraction methods [Liu et al. 2003; Simon and Lausen 2005; Zhai and Liu 2006], tag path based methods [Miao et al. 2009], visual feature based methods [Gatterbauer et al. 2007; Liu et al. 2010; Zhao et al. 2005], and automatic record-level wrapper induction methods [Kayed and Chang 2010]. Methods based on heuristic rules cannot be generalized well. Repetitive pattern based methods such as IEPAD [Chang and Lui 2001] and DeLa [Wang and Lochovsky 2003] show some potential in solving this issue because similar templates used in formatting the records make it feasible to mine some repetitive patterns as clues for locating records in the page. However, one limitation of such pattern mining methods is that they are not robust against optional data and tags appeared in the records.

The similarity-based method tackles this limitation with approximate matching to identify repeating objects. MDR [Liu et al. 2003], DEPTA [Zhai and Liu 2006], and NET [Liu and Zhai 2005] are such techniques, which utilize string or tree edit distance to assess whether two adjacent subtree groups, known as generalized nodes, are repetitions of the same data type. ViPER [Simon and Lausen 2005] is another work which computes the similarity of each pair of single subtrees to detect record region. Then it involves some visual perception to segment the detected regions into records. Since these methods highly depend on the pairwise similarity computation of subtrees or subtree groups, they separate the task into two steps, namely, record region detection and record segmentation. They search the possible record regions in the entire DOM tree with a traversal manner. In contrast, our framework can efficiently analyze the DOM tree structure with a global view and find the regions and data records. Furthermore, our framework is free from their limitations regarding the subtree grouping, for instance, fixed length of generalized node or single subtree pairs. The concept of nested records in NET [Liu and Zhai 2005] is significantly different from nested regions processed in this paper. In our framework, the sub-records nested in a particular record are detected as tandem repeats so that the super record is recognized as a normal data record. The extraction of sub-records can be tackled with simple postprocessing operation. Miao et al. investigated tag paths in a Web page to perform record extraction [Miao et al. 2009]. Their method transforms a DOM tree into pieces of tag paths, and clusters the paths according to the defined similarity measure to detect record

regions. One limitation of this method is that it cannot take into account the record boundary information during region detection. Hence, it needs a separate step to segment records after region detection. Furthermore, their method clusters the tag paths across the entire page and does not consider the proximity relations of the paths. Thus, the same tag path may be used in different blocks of the page, even these blocks are far away from each other.

Although ViPER [Simon and Lausen 2005] and the work by Miao et al. [Miao et al. 2009] utilize some visual information from rendered Web page to assist record segmentation, they depend on the tag structure to detect record regions. In contrast, ViNTs [Zhao et al. 2005] utilizes the visual information first to identify content regularities and then combines them with the tag structure regularities to generate wrappers. ViNTs cannot separate horizontally arranged records, e.g., the records in a nested region, and identify multiple regions. Zhao et al. enhanced ViNTs in their later work [Zhao et al. 2006] to address the multi-section cases in search engine result pages. Pure visual feature based methods such as VENTex [Gatterbauer et al. 2007] and ViDE [Liu et al. 2010] are effective to extract records from pages with well organized visual features. With the help of visual information of the rendered pages, these methods are able to select some major blocks that may have high potential of containing data records. However, they suffer from two limitations, namely, the inefficiency of Web page rendering, and the difficulty of accurate rendering. For a single Web page in a repository, its related cascading style sheet (CSS) and JavaScript files are normally not cached by the repository. Therefore, it is very likely that this page cannot be correctly rendered. Furthermore, the rendering operation is time consuming. In this paper, we do not use these expensive features although our framework is open to incorporate them.

Kayed and Chang proposed an automatic template induction method called FiVaTech [Kayed and Chang 2010] that can achieve record extraction from a single page. This method does not consider different HTML tags with the same meaning since it is assumed that the template is fixed for the same data type. In addition, level crossing is not permitted in the detection of peer nodes. These assumptions reduce its flexibility and it cannot handle cases with more variants well which are often observed in manually edited record regions. Furthermore, this method induces possible wrappers at each level of the DOM tree leading to high computational cost.

Zhu et al. proposed a model based on Hierarchical Conditional Random Field (HCRF) to conduct record detection as well as attribute labeling and achieved good performance in tackling product record extraction [Zhu et al. 2006]. The overall design of HCRF in [Zhu et al. 2006] focuses extensively on product records and it involves some specific product-oriented labels such as product name and price. Necessary modifications on label and feature design are needed if one wants to utilize HCRF in general record extraction as we do in this paper. One limitation is that the authors assume that the boundaries of the visual blocks are coincident with the boundaries of the records with multiple subtrees. However, the page rendering operation may encounter troubles when the separated CSS and JavaScript files are not available. Consequently, the boundaries of the visual blocks may not be reliable and the above assumption may not hold for such cases. In our framework, we do not need the above assumption. Another limitation is that HCRF only defines local similarity based features between adjacent nodes in a sibling sequence. It thus cannot exploit the global regularity of the subtree sequence in a particular record region as we do in our model via long range features in a sibling sequence such as the occurrence-related features. Yang et al. proposed a model based on Markov Logic Networks to tackle the task of post information extraction from Web forums [Yang et al. 2009], which is also investigated by other researchers [Song et al. 2010]. Considering the specialized properties of the

forum data, the authors utilized site level knowledge and defined some specific features to conduct accurate extraction. Thus, the model cannot be readily applied to deal with general record extraction. Some other researchers employed predefined domain ontology [Embley et al. 1999a] or automatically generated domain ontology [Su et al. 2009] to assist the record extraction task. The reader can refer to some surveys [Chang et al. 2006; Sleiman and Corchuelo 2012] for more comprehensive information on the existing works of data record extraction.

Another branch of structured Web data extraction mainly depends on manually-constructed wrappers [Arocena and Mendelzon 1999; Liu et al. 2000]. These methods are difficult to maintain and be applied to different Web sites, because they are very labor intensive. Semi-automatic methods [Hogue and Karger 2005; Hsu and Dung 1998; Kushmerick 2000; Laender et al. 2002; Muslea et al. 2001; Zhai and Liu 2007; Zheng et al. 2007; Zheng et al. 2009], known as wrapper induction, were proposed to tackle this problem. These methods need some labeled pages in the target domain as input to perform the induction. First, the target data or record in a set of training pages are labeled manually. The system then learns the extraction rules from the labeled data automatically, and uses them to extract records from new pages originated from the target domain. To enhance the adaptation capability of the induced wrappers, domain oriented methods were proposed in [Hao et al. 2011; Wong et al. 2009; Wong and Lam 2010]. These methods take some labeled examples of a particular site as input and learn attribute related knowledge of this site. After that, this knowledge is adapted to a new Web site of the same domain to learn new wrappers. Some attempts were made by Zhao et al. [Zhao et al. 2011] in conducting domain-independent Web IE aiming at extracting open-domain attribute name and value pairs from Web pages. They formulated the task as a structured classification problem, which shares some resemblance to our framework, on the structured representation of Web pages. However, our framework targets at extracting different information, namely, data records. Furthermore, we propose a record region oriented loss function as well as a refined training method taking into account this loss function.

It should be noted that the problem setting in our framework is significantly different from the unsupervised instance-based learning data extraction methods, such as RoadRunner [Crescenzi et al. 2001] and EXALG [Arasu and Garcia-Molina 2003]. Their methods tackle the task of site-oriented data extraction by taking several pages coming from the same Web site as input, and extract the underlying template or schema automatically. However, our objective is to detect data records and regions not limited to particular sites and does not require that several pages from the same site are available.

9. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a new framework, named Skoga, to perform robust detection of different kinds of Web data records. Skoga can address some major limitations of existing works. It conducts a quantitative analysis of the DOM structure in a global manner for the detection of record regions and data records with a DOM structure knowledge driven model. To allow different impacts for different features in the structure knowledge, there is a weight associated with each feature which is determined using a development data set via a parameter estimation algorithm based on structured output Support Vector Machine model. An optimization method based on divide-and-conquer principle is developed making use of the DOM structure knowledge to quantitatively infer the best record and region recognition for a page. The experimental results on four evaluation data sets demonstrate the effectiveness of the proposed framework.

Several directions are worth exploring in the future work. One direction is to enhance Skoga by incorporating some reliable visual perception features from the rendered Web pages. Visual perception features such as background color, font size, and margin space are very useful and commonly used in Web page analysis. Skoga is open to incorporate them and it is a worthwhile enhancement for upgrading the accuracy when the real-time performance is not the major concern. Another direction is to transfer the proposed DOM structure knowledge driven model to tackle other Web page understanding tasks, such as the extraction of description details from the pages describing a single object, where the specific labels and features need to be appropriately designed with the characteristics of the tasks considered.

REFERENCES

ARASU, A. AND GARCIA-MOLINA, H. 2003. Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. SIGMOD. 337–348.

AROCENA, G. O. AND MENDELZON, A. O. 1999. Weboql: restructuring documents, databases, and webs. *Theory and Practice of Object Systems 5*, 127–141.

BAUMGARTNER, R., GOTTLOB, G., AND HERZOG, M. 2009. Scalable web data extraction for online market intelligence. *Proceedings of the VLDB Endowment 2*, 1512–1523.

BING, L., LAM, W., AND GU, Y. 2011. Towards a unified solution: data record region detection and segmentation. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. CIKM. 1265–1274.

BING, L., LAM, W., AND WONG, T.-L. 2013. Wikipedia entity expansion and attribute extraction from the web using semi-supervised learning. In *Proceedings of the 6th ACM international conference on Web search and data mining*. WSDM. 567–576.

BUTTLER, D., LIU, L., AND PU, C. 2001. A fully automated object extraction system for the world wide web. In *Proceedings of the the 21st International Conference on Distributed Computing Systems*. ICDCS. 361–370.

CAFARELLA, M. J., HALEVY, A., AND MADHAVAN, J. 2011. Structured data on the web. *Communications of the ACM 54*, 72–79.

CAFARELLA, M. J., HALEVY, A., WANG, D. Z., WU, E., AND ZHANG, Y. 2008. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment 1*, 538–549.

CAI, D., YU, S., WEN, J.-R., AND MA, W.-Y. 2003. VIPS: a Vision-based Page Segmentation Algorithm. Technical report.

CHANG, C.-H., KAYED, M., GIRGIS, M. R., AND SHAALAN, K. F. 2006. A survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering 18*, 1411–1428.

CHANG, C.-H. AND LUI, S.-C. 2001. Iepad: information extraction based on pattern discovery. In *Proceedings of the 10th international conference on World Wide Web*. WWW. 681–688.

CRESCENZI, V., MECCA, G., AND MERIALDO, P. 2001. Roadrunner: Towards automatic data extraction from large web sites. In *Proceedings of the 27th International Conference on Very Large Data Bases*. VLDB. 109–118.

ELMELEEGY, H., MADHAVAN, J., AND HALEVY, A. 2009. Harvesting relational tables from lists on the web. *Proceedings of the VLDB Endowment 2*, 1078–1089.

EMBLEY, D. W., CAMPBELL, D. M., JIANG, Y. S., LIDDLE, S. W., LONSDALE, D. W., NG, Y.-K., AND SMITH, R. D. 1999a. Conceptual-model-based data extraction from multiple-record web pages. *Data & Knowledge Engineering 31*, 227–251.

EMBLEY, D. W., JIANG, Y., AND NG, Y.-K. 1999b. Record-boundary discovery in web documents. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*. SIGMOD. 467–478.

GATTERBAUER, W., BOHUNSKY, P., HERZOG, M., KRÜPL, B., AND POLLAK, B. 2007. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web*. WWW. 71–80.

HAO, Q., CAI, R., PANG, Y., AND ZHANG, L. 2011. From one tree to a forest: a unified solution for structured web data extraction. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. SIGIR. 775–784.

HE, B., PATEL, M., ZHANG, Z., AND CHANG, K. C.-C. 2007. Accessing the deep web. *Communications of the ACM 50*, 94–101.

HOGUE, A. AND KARGER, D. 2005. Thresher: automating the unwrapping of semantic content from the world wide web. In *Proceedings of the 14th international conference on World Wide Web*. WWW. 86–95.

HSU, C.-N. AND DUNG, M.-T. 1998. Generating finite-state transducers for semi-structured data extraction from the web. *Information Systems 23*, 521–538.

KAYED, M. AND CHANG, C.-H. 2010. Fivatech: Page-level web data extraction from template pages. *IEEE Transactions on Knowledge and Data Engineering 22*, 249–263.

KUSHMERICK, N. 2000. Wrapper induction: efficiency and expressiveness. *Artificial Intelligence 118*, 15–68.

LAENDER, A. H. F., RIBEIRO-NETO, B., AND DA SILVA, A. S. 2002. Debye - date extraction by example. *Data & Knowledge Engineering 40*, 121–154.

LIU, B., GROSSMAN, R., AND ZHAI, Y. 2003. Mining data records in web pages. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD. 601–606.

LIU, B. AND ZHAI, Y. 2005. Net – a system for extracting web data from flat and nested data records. In *Proceedings of the 6th international conference on Web Information Systems Engineering*. WISE. 487–495.

LIU, L., PU, C., AND HAN, W. 2000. Xwrap: An xml-enabled wrapper construction system for web information sources. In *Proceedings of the 16th International Conference on Data Engineering*. ICDE. 611–621.

LIU, W., MENG, X., AND MENG, W. 2010. Vide: A vision-based approach for deep web data extraction. *IEEE Transactions on Knowledge and Data Enginering 22*, 447–460.

LUO, X., XU, Z., YU, J., AND CHEN, X. 2011. Building association link network for semantic link on web resources. *IEEE Transactions on Automation Science and Engineering 8,* 3, 482–494.

MADHAVAN, J., KO, D., KOT, L., GANAPATHY, V., RASMUSSEN, A., AND HALEVY, A. 2008. Google's deep web crawl. *Proceedings of the VLDB Endowment 1*, 1241–1252.

MIAO, G., TATEMURA, J., HSIUNG, W.-P., SAWIRES, A., AND MOSER, L. E. 2009. Extracting data records from the web using tag path clustering. In *Proceedings of the 18th international conference on World wide web*. WWW. 981–990.

MUSLEA, I., MINTON, S., AND KNOBLOCK, C. A. 2001. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems 4*, 93–114.

RYAN, M. S. AND NUDD, G. R. 1993. The viterbi algorithm. Technical report.

SIMON, K. AND LAUSEN, G. 2005. Viper: augmenting automatic information extraction with visual perceptions. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. CIKM. 381–388.

SLEIMAN, H. A. AND CORCHUELO, R. 2012. A survey on region extractors from web documents. *IEEE Transactions on Knowledge and Data Engineering 99,* PrePrints.

SONG, X., LIU, J., CAO, Y., LIN, C.-Y., AND HON, H.-W. 2010. Automatic extraction of web data records containing user-generated content. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. CIKM. 39–48.

SU, W., WANG, J., AND LOCHOVSKY, F. H. 2009. Ode: Ontology-assisted data extraction. *ACM Transactions on Database Systems 34*, 12:1–12:35.

TSOCHANTARIDIS, I., JOACHIMS, T., HOFMANN, T., AND ALTUN, Y. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research 6*, 1453–1484.

WANG, J. AND LOCHOVSKY, F. H. 2003. Data extraction and label assignment for web databases. In *Proceedings of the 12th international conference on World Wide Web*. WWW. 187–196.

WONG, T.-L. AND LAM, W. 2010. Learning to adapt web information extraction knowledge and discovering new attributes via a bayesian approach. *IEEE Transactions on Knowledge and Data Engineering 22*, 523–536.

WONG, T.-L., LAM, W., AND CHAN, S.-K. 2006. Collaborative information extraction and mining from multiple web documents. In *Proceedings of the 2006 SIAM International Conference on Data Mining*. SDM. 440–450.

WONG, T.-L., LAM, W., AND CHEN, B. 2009. Mining employment market via text block detection and adaptive cross-domain information extraction. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. SIGIR. 283–290.

YAMADA, Y., CRASWELL, N., NAKATOH, T., AND HIROKAWA, S. 2004. Testbed for information extraction from deep web. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*. WWW Alt. 346–347.

YANG, C., CAO, Y., NIE, Z., ZHOU, J., AND WEN, J.-R. 2010. Closing the loop in webpage understanding. *IEEE Transactions on Knowledge and Data Engineering 22*, 639–650.

YANG, J.-M., CAI, R., WANG, Y., ZHU, J., ZHANG, L., AND MA, W.-Y. 2009. Incorporating site-level knowledge to extract structured data from web forums. In *Proceedings of the 18th international conference on World wide web*. WWW. 181–190.

ZHAI, Y. AND LIU, B. 2006. Structured data extraction from the web based on partial tree alignment. *IEEE Transactions on Knowledge and Data Engineering 18*, 1614–1628.

ZHAI, Y. AND LIU, B. 2007. Extracting web data using instance-based learning. *World Wide Web 10*, 113–132.

ZHAO, B., YIN, X., AND XING, E. P. 2011. Max margin learning on domain-independent web information extraction. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. CIKM. 1305–1310.

ZHAO, H., MENG, W., WU, Z., RAGHAVAN, V., AND YU, C. 2005. Fully automatic wrapper generation for search engines. In *Proceedings of the 14th international conference on World Wide Web*. WWW. 66–75.

ZHAO, H., MENG, W., AND YU, C. 2006. Automatic extraction of dynamic record sections from search engine result pages. In *Proceedings of the 32nd international conference on Very large data bases*. VLDB. 989–1000.

ZHENG, S., SONG, R., WEN, J.-R., AND GILES, C. L. 2009. Efficient record-level wrapper induction. In *Proceeding of the 18th ACM conference on Information and knowledge management*. CIKM. 47–56.

ZHENG, S., SONG, R., WEN, J.-R., AND WU, D. 2007. Joint optimization of wrapper generation and template detection. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD. 894–902.

ZHU, J., NIE, Z., WEN, J.-R., ZHANG, B., AND MA, W.-Y. 2006. Simultaneous record detection and attribute labeling in web data extraction. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD. 494–503.