

Incorporating Site-Level Knowledge to Extract Structured Data from Web Forums*

Jiang-Ming Yang[†], Rui Cai[†], Yida Wang[‡], Jun Zhu[§], Lei Zhang[†], and Wei-Ying Ma[†]

[†]Microsoft Research, Asia. {jmyang, ruicai, leizhang, wyma}@microsoft.com

[‡]CSSAR, Chinese Academy of Sciences. wangyida@cssar.ac.cn

[§]Dept. Computer Science and Technology, Tsinghua University. jun-zhu@mails.tsinghua.edu.cn

ABSTRACT

Web forums have become an important data resource for many web applications, but extracting structured data from unstructured web forum pages is still a challenging task due to both complex page layout designs and unrestricted user created posts. In this paper, we study the problem of structured data extraction from various web forum sites. Our target is to find a solution as general as possible to extract structured data, such as *post title*, *post author*, *post time*, and *post content* from any forum site. In contrast to most existing information extraction methods, which only leverage the knowledge inside an individual page, we incorporate both *page-level* and *site-level* knowledge and employ Markov logic networks (MLNs) to effectively integrate all useful evidence by learning their importance automatically. *Site-level* knowledge includes (1) the linkages among different object pages, such as *list pages* and *post pages*, and (2) the inter-relationships of pages belonging to the same object. The experimental results on 20 forums show a very encouraging information extraction performance, and demonstrate the ability of the proposed approach on various forums. We also show that the performance is limited if only page-level knowledge is used, while when incorporating the site-level knowledge both precision and recall can be significantly improved.

Categories and Subject Descriptors

H.3.m [Information Storage and Retrieval]: Miscellaneous - Data Extraction; Web; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval - Information filtering; I.5.1 [Pattern Recognition]: Models - Statistical

General Terms

Algorithms, Performance, Experimentation

Keywords

Web forums, Structured data, Information extraction, Site-level knowledge, Markov logic networks (MLNs)

*This work was performed when the 3rd and the 4th authors were visiting Microsoft Research, Asia.

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

1. INTRODUCTION

The rapid growth of Web 2.0 is making web forums (also named bulletin or discussion board) an important data resource on the Web. The strong driving force behind web forums is the power of users and communities. With millions of users' contribution, plenty of highly valuable knowledge and information have been accumulated on various topics including recreation, sports, games, computers, art, society, science, home, health, etc. [1]. As a result, recent years have witnessed more and more research efforts trying to leverage information extracted from forum data to build various web applications. Some exemplary applications include extracting question-answer pairs for QnA service [5]; collecting review comments for business intelligence [6], and discovering expertise networks in online communities [18].

To use forum data, the fundamental step in most applicants is to extract structured data from unstructured forum pages represented in HTML format by removing useless HTML tags and noisy content like advertisements. Only after extracting such structured data can we further exploit forum data to discover communities, find emerging topics, model user interests, etc. However, automatically extracting structured data is not a trivial task due to both complex page layout designs and unrestricted user created posts. This problem has become a major hindrance for efficiently using web forum data.

In this paper, we study the problem of structured data extraction from web forum sites. Our target is to find a solution as general as possible to extract structured data such as *post title*, *post author*, *post time*, and *post content* from any forum site. Because the number of forums is very large, at least tens of thousands, the task is very challenging. However, no matter how complex the forum page layouts are, forum sites have some intrinsic characteristics which make it possible to find a general solution. The most relevant work for structuring web forum data is web data extraction, which has been an active research topic in recent years [2, 7, 10, 11, 17, 19–21]. In general, web data extraction approaches can be classified into two categories: *template-dependent* and *template-independent*.

Template-dependent methods, as the name implies, utilize a wrapper as an extractor for a set of pages which were generated based on the same layout template. A wrapper is usually represented in the form of a regular expression or a tree structure. Such a wrapper can be manually constructed, semi-automatically generated by interactive learning [19], or even discovered fully automatically [17]. However, for web forums, different forum sites usually employ

different templates. Even for those forums built with the same forum software, such as vBulletin and phpBB, they still have various customized templates¹. Even worse, most forum sites periodically update their templates to provide better user experience. Therefore, the cost of both generating and maintaining wrappers for so many (may be tens of thousands of) forum templates is extremely high and makes it impractical in real applications. Furthermore, wrapper-based methods also suffer from noisy and unrestricted data in forums. For example, posts in technical forums usually contain user-created HTML scripts, which often mislead the wrapper and degenerate the data extraction performance.

To provide a more general solution for web data extraction, template-independent methods have been proposed recently and have shown their feasibilities to handle web pages with different layout characteristics [2, 11, 20, 21]. These approaches treat data extraction as a segmentation problem, and employ probabilistic models to integrate more semantic features and sophisticated human knowledge. Therefore, template-independent methods have little dependence on specific templates. Moreover, some template-independent approaches have the advantage of doing data extraction and attribute labeling simultaneously [8, 11, 21]. In practice, most existing template-independent methods depend on features inside an individual page; and separately inference each input page for data extraction. For applications like customer review extraction, page-level information is sufficient and the single page-based inference is also practical. However, as we will show in the experiments, for forum data extraction only adapting page-level information is not enough to deal with both complex layout designs and unrestricted user created posts in web forums.

Actually, besides page-level features, information related to the organization structure of a forum site can also help data extraction. In forum data extraction, we usually need to extract information from several kinds of pages such as *list page* and *post page*, each of which may correspond to one kind of *data object*. Pages of different objects are linked with each other. For most forums, such *linkages* are usually statistically stable, which can support some basic assumptions and provide additional evidence for data extraction. For example, if a link points to a page of *user profile*, the anchor text of this link is very likely an *author name*. Second, the *interrelationships* among pages belonging to the same object can help verify the misleading information existing in some individual pages. For example, although user-submitted HTML codes on some post pages may bring ambiguities in data extraction, the joint inference cross multiple post pages can help the extractor distinguish such noise. The *linkages* and *interrelationships*, both of which are dependent on the site-structure information beyond a single page, are called *site-level knowledge* in this paper.

In this paper, we propose a *template-independent* approach specifically designed for structured data extraction on web forums. To provide a more robust and accurate extraction performance, we incorporate both page-level information and site-level knowledge. To do this, we need to know what kinds of page objects a forum site has, which object a page belongs to, and how different page objects are connected with each others. These information can be obtained by reconstructing the *sitemap* of the target forum.

A sitemap is a directed graph in which each *vertex* represents one page object and each *arc* denotes a linkage between two vertices. In addition, we can also identify vertices of *list*, *post*, and *user profile* from most forum sitemaps automatically [4]. After that, we collect three kinds of evidence for information extraction: 1) *inner-page* features which cover both the semantic and layout information on an individual page; 2) *inter-vertex* features which describe the linkage-related observations; and 3) *inner-vertex* features which characterize the interrelationships among pages in one vertex. Finally, we leverage the power of Markov logic networks (MLNs) [13] to combine all these evidences statistically for inference. By integrating all the evidence and learning the corresponding importance, MLNs can handle uncertainty and tolerate imperfect and contradictory knowledge [12, 15]. The experimental results on 20 forums show a very encouraging performance, and also demonstrate the generalization ability of our approach on different forums. We also show that the performance is limited if only page-level knowledge is used, while incorporating the site-level knowledge both the precision and recall can be significantly improved.

This paper is organized as follows. We briefly review some related research efforts in Section 2, and clarify some basic concepts and our goals in Section 3. The overview of the proposed approach is introduced in Section 4, and the algorithm details are described in Section 5. Experimental evaluations are reported in Section 6, and in the last section we draw conclusions and point out some future directions.

2. RELATED WORKS

In recent years, web data extraction has become an active research topic since more and more applications now rely on resources from the Internet. As we have introduced in the Introduction, those related research efforts are either *template-dependent* [7, 10, 17, 19] or *template-independent* [2, 11, 20, 21].

Template-dependent methods, or wrapper-based methods, usually focus on data extraction from a limited number of Websites. Most of these approaches utilize the structure information on the DOM² tree of a HTML page to characterize a wrapper [17, 19]. That is, sub-trees with similar structures are most likely to represent similar data records. However, inducing robust and effective wrappers is not a trivial task as DOM trees are usually complex [7]; and some approaches need manual interaction to improve the performance [19]. Furthermore, even targeting at only a few Websites, the wrapper maintenance is still a hard problem [10]. Therefore, template-dependent methods are not practical for our goal of data extraction from general Web forums.

Template-independent methods target at providing more general solutions which are insensitive to the templates of Websites. Most of these methods are based on probabilistic models, and try to integrate semantic information and human knowledge in inference. For example, relational Markov networks was utilized in [2] to extract protein names from biomedical text; CRF was adopted to extract tables from plain-text government statistical reports; and two new models, 2D-CRF and hierarchical CRF, were proposed in [20, 21] to detect and label product reviews from web pages. These methods have achieved promising performance and

¹<http://www.vbulletin-faq.com/skins-styles.htm>

²http://en.wikipedia.org/wiki/Document_Object_Model

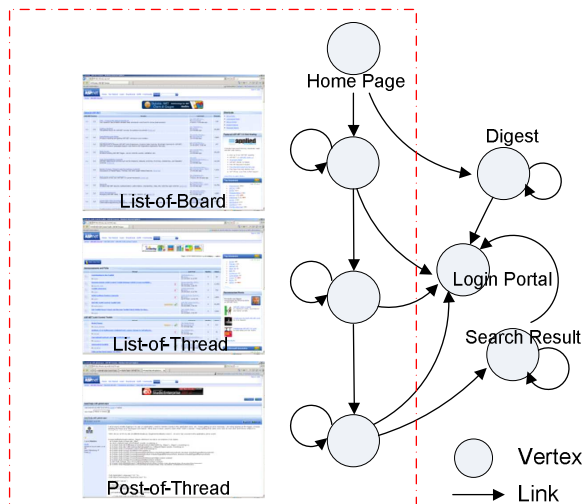


Figure 1: An illustration of the sitemap of the ASP.NET forum. This graph has been simplified for a clear view.

also shown their flexibilities. However, as we argued in the introduction section, most of them only adopt information within a single page/document, and thus cannot fully leverage the extra site-structure information of web forums.

Actually, we have noticed that some work has tried to use site-level information for data extraction. In [9], tables containing link items were discovered by evaluating the content redundancies in the content of a table item and the detailed page that the link points to, because the detailed page should contain additional information about the related link item. To do this, the algorithms in [9] need both the table page and all its detailed pages for inference. In contrast, our method doesn't rely on detailed pages. Instead, it is based on sitemap which can provide much richer site-structure information but with more compact representation. Moreover, we only need a few randomly sampled pages to re-construct the sitemap of a target site [4].

Markov logic networks [13] is a general probabilistic model for modeling relational data. MLNs have been applied to joint inference under different scenarios, such as segmentation of citation records [12] and entity resolution [15]. We'll give a more detailed introduction to MLNs in Section 5.

3. PROBLEM SETTING

To make a clear presentation and facilitate the following discussions, we first explain some concepts in this paper.

- **SITEMAP.** A *sitemap* is a directed graph consisting of a set of vertices and the corresponding links. Each vertex represents a group of forum pages which have similar page layout structure; and each link denotes a kind of linkage relationship between two vertices. Fig. 1 provides an illustration of the *sitemap* for the ASP.NET forum (<http://forums.asp.net>). For vertices, we can find that each vertex is related to one kind of pages in the forum, as shown in Fig. 1 with typical pages and labels. In this paper, we are interested in extracting information from the vertices of “list-of-board”, “list-of-thread”, and “post-of-thread”, which are related to user-created content and marked within the red-dashed rectangle in Fig. 1. Such information is very general as most forums have these vertices and the

linkages among these vertices are also stable. The vertices out of the rectangle usually provide supportive functions for a forum; and are out of the scope of this paper.

- **LIST PAGE.** For users' convenience, a well-organized forum site consists of a tree-like directory structure containing topics (commonly called threads) at the lowest end and posts inside threads. For example, the tree of the ASP.NET forum is a four-level structure shown in the red-dashed tree in Fig. 1. Pages from branch nodes on the tree are called *list pages*, such as the “list-of-board” and “list-of-thread” in Fig. 1. *List pages* within the same node share the same template and each page contains a sets of *list records*. The corresponding *post title* in the *list record* will help users navigate to pages in its children nodes on the tree. Therefore, the goal of data extraction on such *list pages* is to extract the *post title* of every *list record*.
- **POST PAGE.** Pages in the leaf node on the tree are called *post pages*, which contain detailed information of user posts. Each post usually consists of fields such as *post author*, *post time*, and *post content*, which are the goal of data extraction in this paper.

At last, we formally define the problem of Web forum data extraction as:

DEFINITION 1. Given a DOM tree³, data record extraction is the task of locating the minimum set of HTML elements that contains the content of a data record and assigning the corresponding labels to the parent node of these HTML elements. For a list page or post page containing multiple records, all the data records need to be identified.

4. SYSTEM OVERVIEW

The flowchart of our method is illustrated in Fig. 2, which mainly consists of three parts: (a) offline sitemap recovering; (b) feature extraction; and (c) joint inference for the pages with same template.

The goal of the first step is to automatically estimate the sitemap structure of the target site using a few sampled pages. In practice, it was found that sampling around 2000 pages is enough to re-construct the sitemap of most forum sites [4]. After that, pages with similar layout structures are further clustered into groups (vertices), as shown with the green dashed ellipse in Fig. 2. Then, all possible links among various vertices are established, if in the source vertex there is a page having an out-link pointing to a page in the target vertex. As introduced in [4], each link is described by both the URL pattern and the location (the region where the corresponding out-links located). At last, since some long list or long thread may be divided into several individual pages connected by page-flipping links, we can archive them together by detecting the page-flipping links and treat them as a single page. This greatly facilitates the following data extraction. For more details of this step, please refer to the previous work in [4, 16].

The second part is in charge of features extraction. There are three kinds of features according to their generation source. (1) Inner-page features which leverage the relations among the elements inside a page, such as the size and location of each elements, the inclusion relation among elements,

³<http://en.wikipedia.org/wiki/HTML>

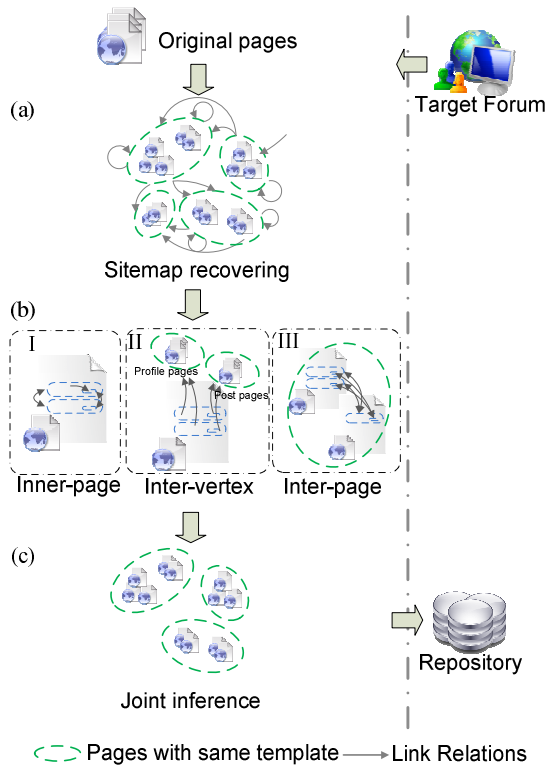


Figure 2: The flowchart to the proposed approach, which consists of three parts: (a) sitemap recovering; (b) feature extraction; and (c) joint inference.

the sequence order among elements and so on; (2) Inter-template features which are generated based on the above site-level knowledge. Links with similar functions usually navigate to the same vertex on the sitemap, such as the *list title* which usually navigates to the vertex containing post pages. We can get the function for each link based on its location. This is a very helpful feature to tag right labels to the corresponding elements; and (3) Inter-page features. For pages in a given vertex, records with same semantic labels (title, author, and etc.) should be presented in same locations in these pages. We introduce such features to improve the feature extraction results of pages in the same vertex.

To combine these features effectively, we utilize the Markov Logic Networks (MLNs) [13] to model the aforementioned relation data. By the joint inference of pages inside one same vertex, we can integrate all the three features and compute maximum a posteriori (MAP) probability of all query evidences.

5. MODULAR DETAILS

5.1 Markov Logic Networks

Markov Logic Networks (MLNs) [13] are a probabilistic extension of a first-order logic for modeling relation data. In MLNs, each formula has an associated weight to show how strong a constraint is: the higher the weight is, the greater the difference in log probability between a world that satisfies the formula and one that does not, other things being equal. In this sense, MLNs soften the constraints of a first-order logic. That is, when a world violates one formula it is less probable, but not impossible. In a first-order logic,

if a world violates one constraint it will have probability zero. Thus, MLN is a more sound framework since the real world is full of uncertainty, noise imperfect and contradictory knowledge [13].

An MLN can be viewed as a template for constructing Markov Random Fields [12]. With a set of formulas and constants, MLNs define a Markov network with one node per ground atom and one feature per ground formula. The probability of a state x in such a network is given by:

$$P(X = x) = \frac{1}{Z} \prod_i \phi_i(x_{\{i\}})^{n_i(x)} \quad (1)$$

where Z is a normalization constant, $n_i(x)$ is the number of true groundings of F_i in x , $x_{\{i\}}$ is the state (truth values) of the atoms appearing in F_i , and $\phi_i(x_{\{i\}}) = e^{w_i}$, w_i is the weight of the i^{th} formula.

Eq. (1) defines a generative MLN model, that is, it defines the joint probability of all the predicates. In our application of forum page segmentation, we know the evidence predicates and the query predicates *a priori*. Thus, we turn to the discriminative MLN. Discriminative models have the great advantage of incorporating arbitrary useful features and have shown great promise as compared to generative models [8, 14]. We partition the predicates into two sets—the evidence predicates X and the query predicates Q . Given an instance x , the discriminative MLN defines a conditional distribution as follows:

$$P(q|x) = \frac{1}{Z_x(\mathbf{w})} \exp \left(\sum_{i \in F_Q} \sum_{j \in G_i} w_i g_j(q, x) \right), \quad (2)$$

where F_Q is the set of formulas with at least one grounding involving a query predicate, G_i is the set of ground formulas of the i^{th} first-order formula, and $Z_x(\mathbf{w})$ is the normalization factor. $g_j(q, x)$ is binary and equals to 1 if the j^{th} ground formula is true and 0 otherwise.

As defined in [21], with the conditional distribution in Eq. (2) web data extraction is a task to compute maximum a posteriori (MAP) probability of q and extract data from this assignment q^* :

$$q^* = \arg \max_q P(q|x) \quad (3)$$

In this paper, we mainly focus on extracting the following six objects, *list record*, *list title*, *post record*, *post author*, *post time*, and *post content*. The atomic extraction units are HTML elements. Thus, in our MLN model, we define the corresponding query predicates q as, $IsListRecord(i)$, $IsTitleNode(i)$, $IsPostRecord(i)$, $IsAuthorNode(i)$, $IsTimeNode(i)$, and $IsContentNode(i)$, respectively, where i denotes the i^{th} element. The evidence x are the features of the HTML elements. In a discriminative MLN model as defined in Eq. (2), the evidence x can be arbitrary useful features. In this paper, the features include three parts: inner-page features (e.g., the size and location of each element), inner-page features (e.g., the order among some time-like elements), and inter-template features (e.g., the alignment relation among elements). With these predefined features, we define some rules or the formulas in MLNs, such as the *post record* element must contains *post author*, *post time* and *post content* nodes, etc. These formulas represent some kinds of relations among HTML elements. With these formulas, the resultant MLN can effectively capture the mutual dependencies

among different extractions and thus achieve globally consistent joint inference.

Note that in the above general definition, we treat all the HTML elements identically when formulating the query and evidence predicates. However, in practice, HTML elements can show obviously different and non-overlapping properties. For example, the elements staying at the leaves of a DOM tree are quite different from the inner nodes. Only the elements at leaf nodes can be a *post author* or a *post time*; only the inner elements can be *list record* or a *post record*. Thus, we group these elements into three non-overlapping groups (see Section 5.2 for more details). This can be implemented in an MLN model by defining them as different types. In this way, we can significantly reduce the number of possible groundings when MLN is performing inference. Also, this prior grouping knowledge could potentially reduce the ambiguity in the model and thus achieve better performance.

5.2 Features

To accelerate the training and inference process, the DOM tree elements are divided into the following three categories according to their attributes and our requirement:

- Text element (t). This kind of elements always acts as leaves in DOM trees and ultimately contains all the extracted information. For some plain text information like *post time*, we should identify this kind of elements in extraction.
- Hyperlink element (h). This kind of elements corresponds to hyperlinks in a web page which usually have tags $\langle a \rangle$ in HTML files. Web pages inside a forum are connected to each other through hyperlinks. For example, *list pages* and *post pages* are linked together by hyperlinks of *post titles* pointing from the former to the latter. Further observation shows that inside a forum site, some required information such as *post title* and *post author*, is always enveloped in hyperlink elements.
- Inner element (i). All the other elements located inside a DOM tree are defined as inner elements. In practice, the *list records* or *post records* and *post contents* are always embraced in inner elements.

In MLNs model, we will treat the above three kinds of evidences separately to accelerate the training and inference process. In the following, we will represent the above three kinds of evidences as t , h , and i , respectively. We list the corresponding features in Table 1.

5.2.1 Inner-page features

The inner-page features leverage the relations among elements inside a page; and are listed in Table 1. This corresponds to the part (I) in Fig. 2. We describe these features from four aspects:

The time feature: To extract time information, we first get candidates whose content is short and containing string like mm-dd-yyyy, dd/mm/yyyy, or some specific terms like Monday and January. This evidence can be presented as $IsTimeFormat(t)$ for each text element t . Similarly, we can introduce another evidence $ContainTimeNode(i)$.

The inclusion relation: Data records usually have inclusion relations. For example, a *list record* should contain a *list title* which can be represented as $HasLongestLink(i, h)$; a *post content* should be contained in a *post record* and usually contains a large ratio of text which can be represented as $HasDescendant(i, i')$ and $ContainLongText(i)$.

The alignment relation: Since data is generated from database and represented via templates, data records with the same label may appear repeatedly in a page. If we can identify some records with high confidence, we may assume other records aligned with them should have the same label. There are two methods to generate the alignment information: (1) By rendering via a web browser, we can get the location information of each element [3]. Two elements are aligned with each other if they are aligned similarly in vertical or horizontal direction. (2) By recursively matching their children nodes pairs by pairs [17], we define the similarity measurement including the comparison of nodes' tag types, tag attributes, and even the contained texts. We can represent the alignment relation similarity on i , h , and t as $InnerAlign(i, i')$, $HyperAlign(h, h')$, and $TextAlign(t, t')$. We can get the similar alignment relation if an element is aligned with its sibling nodes and represent as $IsRepeatNode(i)$.

Time Order: The order of the *post time* is quite special. Since *post records* are generated sequentially along timeline, the *post time* should be sorted ascendently or descendantly. This helps us to distinguish other noisy time content such as users' registration time, and get the right information. If the time information in the same location satisfies the ascent or descendant order, we represent it as $UnderSameOrder(t)$.

5.2.2 Inter-vertex features

The inter-vertex features are generated based on the site-level knowledge. In a sitemap, the pages inside a given vertex usually have similar functions, as shown in Fig. 1. If we can navigate to a vertex containing post pages via a given link in a list page, this link probably represents the title of a thread. We represent this as $IsPostLink(h)$, $HasPostLink(i, h)$, and $ContainPostlink(i)$. Similarly, if we can navigate to a vertex containing profile pages via a given link, this link probably contains a user name. We represent this as $IsAuthorLink(h)$, $HasAuthorLink(i, h)$, and $ContainAuthorlink(i)$. For each given page, we can map it to one vertex and get the function of each link in this page based on the location of this link. These features are also listed in Table 1 and correspond to the part (II) in Fig. 2.

5.2.3 Inner-vertex features

In general, for different pages from the same vertex in the sitemap of a forum, the records of the same semantic labels (title, author and etc.) should be presented in a same DOM path. We introduce these alignment features to further improve the results within a set of pages belonging to a same template. This feature can be leveraged for three kinds of elements i , h , and t , respectively. We represent them as $InnerAlignIV(i, i')$, $HyperAlignIV(h, h')$, and $TextAlignIV(t, t')$. The details are also listed in Table 1 and correspond to the part (III) in Fig. 2.

Table 1: The proposed inner-page, inter-vertex, and inner-vertex features.

Type	Feature	Description
Inner-Page	$IsTimeFormat(t)$	The text string in text node t appears as time-format.
	$ContainTimeNode(i)$	There exist one text element t , t is contained in inner node i and $IsTimeFormat(t)$ is true.
	$HasLongestLink(i, h)$	The hyperlink node h is embraced in the inner node i and its text content is longer than any other hyperlink embraced in i .
	$HasDescendant(i, i')$	The inner node i' is one of the descendants of the inner node i .
	$ContainLongText(i)$	The inner node i has several text elements in its sub DOM tree which contain long periods of text contents.
	$InnerAlign(i, i')$	The location of inner node i and its sub DOM tree structure are similar with those of another inner node i'
	$HyperAlign(h, h')$	The location of hyperlink node h and its sub DOM tree structure are similar with those of another hyperlink node h' .
	$TextAlign(t, t')$	The location of text node t and its sub DOM tree structure are similar with those of another text node t' .
	$IsRepeatNode(i)$	There is at least one sibling of the inner node i which has a similar sub DOM tree.
Inter-Vertex	$IsPostLink(h)$	The hyperlink node h navigates to post-of-thread vertex.
	$HasPostLink(i, h)$	The hyperlink node h is embraced in the inner node i and $IsPostLink(h)$ is true.
	$ContainPostlink(i)$	There exist one hyperlink element h which is contained in the inner node i and $IsPostLink(h)$ is true.
	$IsAuthorLink(h)$	The hyperlink node h navigates to the author profile vertex.
	$HasAuthorLink(i, h)$	The hyperlink node h is embraced in the inner node i and $IsAuthorLink(h)$ is true.
	$ContainAuthorLink(i)$	There exist one hyperlink element h which is contained in the inner node i and $IsAuthorLink(h)$ is true.
Inner-Vertex	$InnerAlignIV(i, i')$	The inner node i in one page shares similar DOM path and tag attributes along the path with another inner node i' in another page.
	$HyperAlignIV(h, h')$	The hyperlink node h in one page shares similar DOM path and tag attributes along the path with another hyperlink node h' in another page.
	$TextAlignIV(t, t')$	The text nodes t in one page shares similar DOM path and tag attributes along the path with another text node t' in another page.

5.3 Formulas

In this section, we introduce the detail formulas used in the two models of list pages and post pages, respectively.

5.3.1 Formulas of list page

Here we assume that a *list record* should be an inner nodes, a *list title* should be contained in a hyperlink node. In order to extract them accurately, we introduce some rules which are presented as following formulas. There are two kinds of rules which basically present the relations among the queries and the evidences. To help readers understand easily, we draw the relations for *list record* and *list title* in Fig. 3.

(1) **Formulas for identifying *list record*.** A *list record* usually contains a link of *list title* which also appears repeatedly. We can identify a *list record* if a candidate element is aligned with a known *list record* inside a page or aligned with a known *list record* in another page of the same vertex. As shown in Fig. 3:

$$\forall i \text{ ContainPostLink}(i) \wedge \text{IsRepeatNode}(i) \quad (4)$$

$$\Rightarrow \text{IsListRecord}(i)$$

$$\forall i, i' \text{ IsListRecord}(i) \wedge \text{InnerAlign}(i, i') \quad (5)$$

$$\Rightarrow \text{IsListRecord}(i')$$

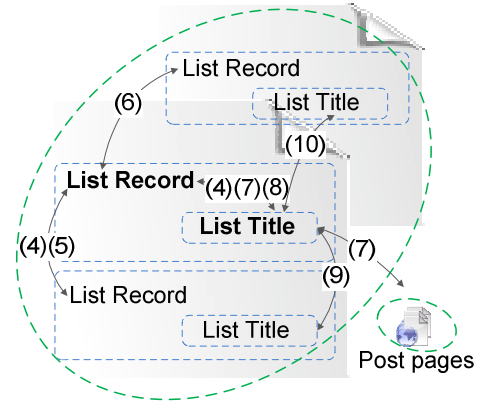


Figure 3: An illustration of the formulas of list page, in which each number in parentheses denotes the corresponding formula in the paper.

$$\forall i, i' \text{ IsListRecord}(i) \wedge \text{InnerAlignIV}(i, i') \quad (6)$$

$$\Rightarrow \text{IsListRecord}(i')$$

(2) **Formulas for identifying *list title*.** A *list title* usually contains a link to the vertex of post pages and is contained in *list record*. The formula (8) will be useful when

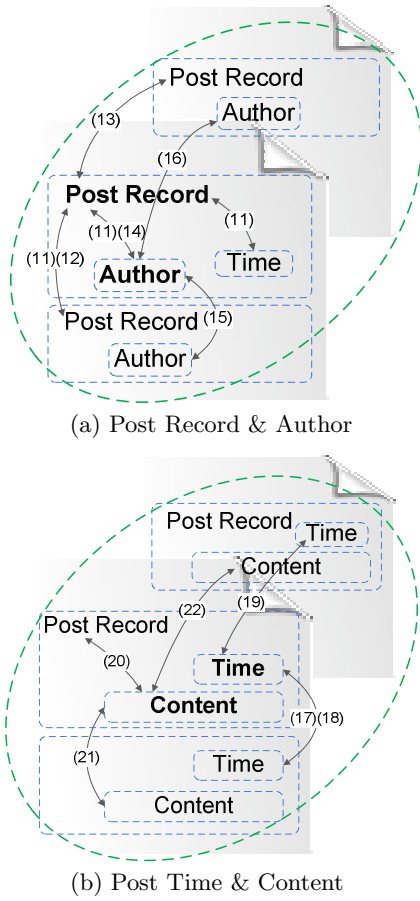


Figure 4: An illustration of the formulas of post page, in which each number in parentheses denotes the corresponding formula in the paper.

we do not have site level information. We can also identify a *list title* if a candidate element is aligned with a known *list title* inside a page or aligned with a known *list title* in another page of the same vertex. It is shown in Fig. 3.

$$\forall i, h \text{ IsListRecord}(i) \wedge \text{HasPostLink}(i, h) \quad (7)$$

$$\Rightarrow \text{IsTitleNode}(h)$$

$$\forall i, h \text{ IsListRecord}(i) \wedge \text{HasLongestLink}(i, h) \quad (8)$$

$$\Rightarrow \text{IsTitleNode}(h)$$

$$\forall h, h' \text{ IsTitleNode}(h) \wedge \text{HyperAlign}(h, h') \quad (9)$$

$$\Rightarrow \text{IsTitleNode}(h')$$

$$\forall h, h' \text{ IsTitleNode}(h) \wedge \text{HyperAlignIV}(h, h') \quad (10)$$

$$\Rightarrow \text{IsTitleNode}(h')$$

5.3.2 Formulas of post page

The *post record* and *post content* should be contained in inner nodes, while a *post author* should be contained in a hyperlink node and a *post time* always appear in a text node as time-format. We can identify our required information by inferring these predicates and try to establish some rules to describe the required elements according to their own evidences. To help readers understand easily, we also draw the relations among *post record*, *post author*, *post time*, and *post content* respectively in the Fig. 4.

(1) **Formulas for identifying *post record*.** A *post record* usually contains a link for *post author* and *post time* and appears repeatedly. We can also identify a *post record* if a candidate element is aligned with a known *post record* inside a page or aligned with a known *post record* in another page of the same vertex. It is shown in Fig. 4(a).

$$\forall i \text{ ContainAuthorLink}(i) \wedge \text{ContainTimeNode}(i) \wedge \quad (11)$$

$$\text{IsRepeatNode}(i) \Rightarrow \text{IsPostRecord}(i)$$

$$\forall i, i' \text{ IsPostRecord}(i) \wedge \text{InnerAlign}(i, i') \quad (12)$$

$$\Rightarrow \text{IsPostRecord}(i')$$

$$\forall i, i' \text{ IsPostRecord}(i) \wedge \text{InterInnerAlign}(i, i') \quad (13)$$

$$\Rightarrow \text{IsPostRecord}(i')$$

(2) **Formulas for identifying *post author*.** A *post author* usually contains a link to the vertex of profile pages and is contained in a *post record*. We can also identify a *post author* if a candidate element is aligned with a known *post author* inside a page or aligned with a known *post author* in another page of the same vertex. It is shown in Fig. 4(a).

$$\forall i, h \text{ IsPostRecord}(i) \wedge \text{HasAuthorLink}(i, h) \wedge \quad (14)$$

$$\Rightarrow \text{IsAuthorNode}(h)$$

$$\forall h, h' \text{ IsAuthorNode}(h) \wedge \text{HyperAlign}(h, h') \quad (15)$$

$$\Rightarrow \text{IsAuthorNode}(h')$$

$$\forall h, h' \text{ IsAuthorNode}(h) \wedge \text{HyperAlignIV}(h, h') \quad (16)$$

$$\Rightarrow \text{IsAuthorNode}(h')$$

(3) **Formulas for identifying *post time*.** A *post time* usually contains time-format content and is sorted ascendingly or descendingly. We can also identify a *post time* if a candidate element is aligned with a known *post time* inside a page or aligned with a known *post time* in another page of the same vertex. It is shown in Fig. 4(b).

$$\forall t \text{ UnderSameOrder}(t) \Rightarrow \text{IsTimeNode}(t) \quad (17)$$

$$\forall t, t' \text{ IsTimeNode}(t) \wedge \text{TextAlign}(t, t') \quad (18)$$

$$\Rightarrow \text{IsTimeNode}(t')$$

$$\forall t, t' \text{ IsTimeNode}(t) \wedge \text{TextAlignIV}(t, t') \quad (19)$$

$$\Rightarrow \text{IsTimeNode}(t')$$

(4) **Formulas for identifying *post content*.** A *post content* is usually the descendant of *post record* and do not contain *post time* and *post author*. We can also identify a *post content* if a candidate element is aligned with a known *post content* inside a page or aligned with a known *post content* in another page of the same vertex. It is also shown in Fig. 4(b).

$$\forall i, i' \text{ IsRepeatNode}(i) \wedge \text{HasDescendant}(i, i') \wedge \quad (20)$$

$$\text{ContainLongText}(i') \wedge (\neg \text{ContainTimeNode}(i') \vee$$

$$\neg \text{ContainHyperLinkAuthor}(i')) \Rightarrow \text{IsContentNode}(i')$$

$$\forall i, i' \text{ IsContentNode}(i) \wedge \text{InnerAlign}(i, i') \quad (21)$$

$$\Rightarrow \text{IsContentNode}(i')$$

$$\forall i, i' \text{ IsContentNode}(i) \wedge \text{InnerAlignIV}(i, i') \quad (22)$$

$$\Rightarrow \text{IsContentNode}(i')$$

6. EXPERIMENTS

In this section, we report experimental results by applying our models to extract structured forum data from both list

Table 2: Web Forums in the Experiments

Id	Forum Site	Description
1	forums.asp.net	Commercial technical
2	www.avforum.com/avs-vb	Audio and video
3	www.codeguru.com/forum	Programming
4	www.computerforum.com	Hardware and software
5	bbs.cqzg.cn	General forum(Chinese)
6	boards.cruisecritic.com	Cruise travel message
7	forums.d2jsp.org	Gaming and trading
8	www.devhardware.com/forums	Electronic and hardware
9	www.disboards.com	Disney trip planning
10	www.dpreview.com/forums	Digital photography
11	www.flyertalk.com/forum	Frequent flyers discussion
12	www.gsmhosting.com	Mobile phone message
13	www.howardforums.com	Mobile phones discussion
14	bbs.imobile.com.cn	Mobile phones(Chinese)
15	www.pctools.com/forum	Personal computer tools
16	photography-on-the.net/forum	Digital photography
17	forums.photographyreview.com	Photography review
18	www.phpbuilder.com/board	PHP programming
19	www.sitepoint.com/forums	Web design
20	www.ubuntuforums.org	Official Ubuntu Linux

and post pages. The results show that our models achieve significant improvements in both list data extraction and post data extraction. We also demonstrate where the gains come from, include: the global features from site-level information, the joint optimization for record detection, and the attribute labeling in multiple pages from the same vertex.

6.1 Experiment Setup

Different forums usually have different layout designs. To evaluate the performance of our system on various situations, 20 different forum sites were selected in diverse categories (including bike, photography, travel, computer technique, and some general forums), as listed in Table 2. Some of these forum sites, such as “www.howardforums.com” and “boards.cruisecritic.com” are powered by the popular forum software “vBulletin”; and “bbs.cqzg.cn” and “bbs.imobile.com.cn” are powered by another tool “Discuz!”; while some others, such as “forums.asp.net” and “forums.d2jsp.org” are customized forums. We have presented some sample pages from these sites in Figure 5. Apparently, these pages are quite different in layout designs, which can be used to evaluate the generalization ability of the proposed models.

To set up a consistent data collection for further evaluation and comparison, we first mirrored the above 20 sites using a crawler. For each forum, after removing error pages due to network problems, we kept about 1,000 list pages and 1,000 post pages, which are enough to represent the pages in each forum. Moreover, for a quantitative evaluation, we manually write a wrapper for each site to extract all the targeted data from the downloaded pages as the ground truth.

In the experiment, we adopt the precision, recall, and their harmonic mean-F1 as the criteria of measurements. For the block data extraction, a block is considered as a correctly detected *list record*, *post record*, or *post content* if it is marked with right label and contains more than 95% content. We not require 100% content here because there is some non-important information like “Contact” or “Reply” buttons. For the attribute data extraction, the *post title* in a *list record*, the *post author* and *post time* in a *post record* are

Table 3: Evaluation results of different methods on both list pages and post pages

	Label	Measure	MLNs-P	MLNs-PV	MLNs-PVV
List Pages	Record	Precision	0.687	0.980	0.997
		Recall	0.611	0.626	0.975
		F1	0.647	0.764	0.986
	Title	Precision	0.905	0.905	0.994
		Recall	0.370	0.376	0.976
		F1	0.526	0.531	0.985
Post Pages	Record	Precision	0.932	0.937	0.996
		Recall	0.656	0.695	0.941
		F1	0.770	0.798	0.967
	Author	Precision	0.751	0.750	0.939
		Recall	0.585	0.628	0.969
		F1	0.658	0.684	0.954
	Time	Precision	0.955	0.956	0.952
		Recall	0.919	0.935	0.933
		F1	0.937	0.945	0.942
	Content	Precision	0.824	0.821	0.914
		Recall	0.742	0.814	0.853
		F1	0.781	0.817	0.882

required to be marked with the right labels, as well as the same content with ground truth to be completely extracted.

6.2 Evaluation of Various Features

Some state-of-the-art approaches, such as [12, 21], mainly focus on jointing web data extraction method for both record detection and attribute labeling with inner-page relations. To see the effect of the site-level features, we construct an MLNs model which does not incorporate the global features, denoted as MLNs-P (inner-page relations) as the baseline method. Similarly, we also construct an MLNs model which incorporates both the inner-page relations and inter-vertex relations, denoted as MLNs-PV. Finally, we construct an MLNs model which integrate all the features including inner-page relations, inter-vertex relations and inner-vertex relations, denoted as MLNs-PVV. We didn’t compare the CRF with our MLNs-based solution as the CRF is actually a special case of MLN by restricting the formulas [12]. A linear-chain CRF can be employed using the similar formulas in Section 5.

And the results are shown in Table 3. We will explain the results in following aspects:

The advantage of joint inference with inner-page relation. The effectiveness of the inner-page relations promises an acceptable performance of the MLNs-P model. For example, there are a lot of datetime fields in a post page, such as the users’ login time, the users’ registration time, the posting time, and even some time content in some posts. It is very hard to extract the exact *post time* information by only checking if a HTML element contains the datetime information. Due to the forum template, the *post time* of each post usually appears in the similar position and satisfies a sequential order. But the orders of other datetime fields besides the real *post time* are usually very random. Similarly, we also verify the label of each data record by checking the position of other data with the same label. This kind of information can help us to filter a lot of noise. By jointing web data extraction method for both record detection and attribute labeling, we can also improve the performance of other data records which are relative to the above data record. For ex-

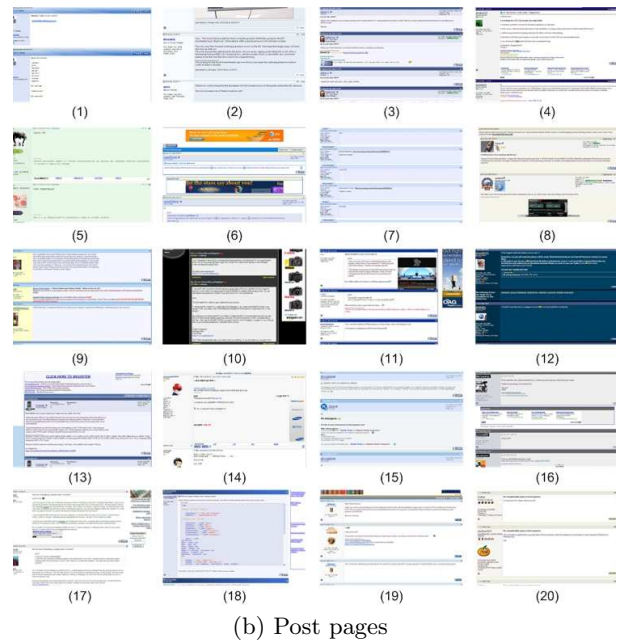
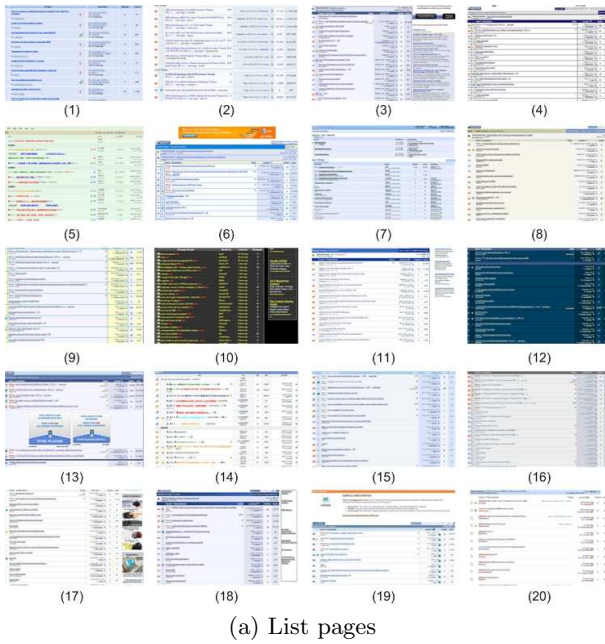


Figure 5: Some screenshots of 20 forum sites for both list pages & post pages.

ample, if we can extract each *post time* exactly, we can also improve the performance of *post record* by the relation that each *post record* should contain one *post time*.

The advantage of inter-template relations. In some situation it is still very hard to judge the label of some data records only with the inner-page relations. For example, we think the *post author* is a link contains some short words. But there may exist a lot of other links with short words, such as “Reply”, “Contact”, some advertisements with short words, and so on. These noises may also appear in the similar position of each *post record*. We have the similar problem for extracting *title* from *alist record*. But we can definitely filter these noises by checking which kind of page the link navigates to. Since the pages are built from templates, for users’ convenience, the links at similar position should have similar functions. Thus we do not need to check the exact page of the link navigate to, we just check the function of the position of a link on the page template. This is the part where the improvement of MLNs-PV comes from.

The advantage of inter-page relations. Some pages may be special, such as some threads having only one *post record*. In this situation, a lot of inner-page relations does not work, such as the *post time*. It will also affect the accuracy of the *post record* and *post content* since they are rely on the result of the *post time*. But we can avoid this situation by serval pages with the same template together and use the inter-page relations to refine the results. This is the reason why MLNs-PVV has the best performance.

6.3 Generalization Ability

The MLNs-PVV model is essentially a supervised method which relies on the training samples. In this section, we would like to evaluate the generalization ability of the MLNs-PVV model. For the above 20 forum sites, we split them into two part: 5 sites for training and 15 sites for testing. Then, we evaluate the performance on the training set and the testing set respectively. The results are shown in Figure 6.

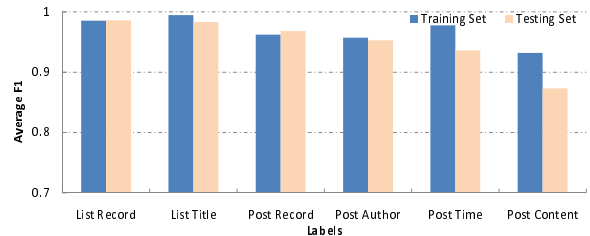


Figure 6: The performance on training set and testing set separately.

The performances on the training set and testing set is quite similar. It is the features but not training set size affect the performance. The model even has a little bit worse of *post record* detection in the training set. This is because some individual sampled pages in the training set contain noise. The model performs better for the detection of *post time* and *post content*. In general, the performance is good for all of these sites.

Since the performance of the MLNs-PVV model may rely on the number of forum sites or number of sampled pages in training, we first trained the MLNs-PVV model with 1 ~ 5 forum sites separately and evaluate their performance on the other 15 sites. In addition, we trained our model with 10 ~ 50 pages from each training site and evaluate their performance on the other 15 sites (we only show the result when there are three sites in the training set). To get an average performance, we repeated the above process for 5 times and get the results which are shown in Figure 7. In general, the performance becomes stable when we had 2 sites and 20 pages from each site. This indicates that our model has satisfied generalization ability in practice.

7. CONCLUSION

In this paper we have presented a template-independent approach to extract structured data from web forum sites.

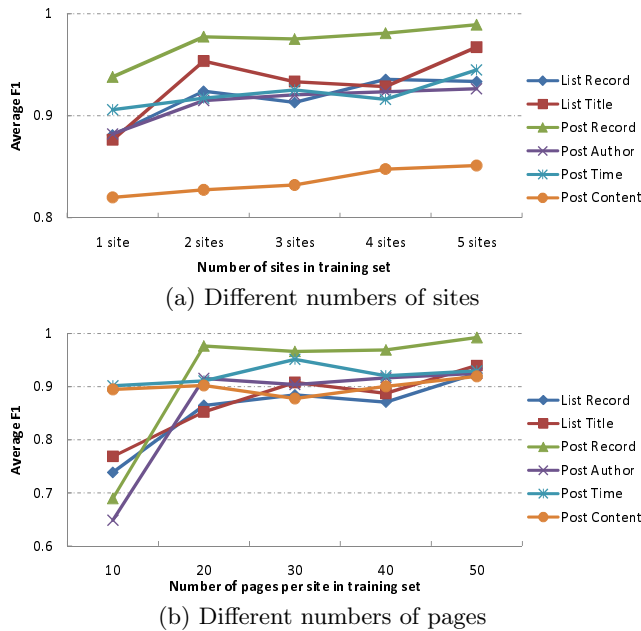


Figure 7: Training with different number of sites or number of pages.

Most existing methods basically depend on features inside an individual page and ignore site-level information. However, as we have shown in the experiments, only considering page-level information is not enough to deal with both complex layout designs and unrestricted user created posts in web forums and the performance are not good enough.

Our method introduces an automatic approach to extract site-level information with the reconstructed sitemap. The site-level knowledge includes (1) linkages among pages inter vertexes of the sitemap should have certain functionalities, such as the title link between list page and post page; and (2) interrelationships of pages sharing the similar layout design, such as the post contents appear in the same DOM path of all post pages. In this way, we can leverage the mutual information among pages inner or inter vertexes of the sitemap. Finally, we incorporate both the page-level and site-level knowledge and employ Markov logic networks (MLNs) to effectively integrate all useful evidences by learning their importance automatically. The experimental results on 20 forums show very encouraging performance of information extraction, and demonstrate the generalization ability of the proposed approach on various forums.

8. REFERENCES

- [1] Big boards. <http://directory.big-boards.com/>, 2008.
- [2] R. Bunescu and R. J. Mooney. Collective information extraction with relational Markov networks. In *Proc. 42nd ACL*, pages 439–446, Barcelona, Spain, July 2004.
- [3] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Block-based Web search. In *Proc. 27th SIGIR*, pages 456–463, Sheffield, UK, July 2004.
- [4] R. Cai, J.-M. Yang, W. Lai, Y. Wang, and L. Zhang. iRobot: An intelligent crawler for Web forums. In *Proc. 17th WWW*, pages 447–456, Beijing, China, April 2008.
- [5] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun. Finding question-answer pairs from online forums. In *Proc. 31st SIGIR*, pages 467–474, Singapore, July 2008.
- [6] N. Glance, M. Hurst, K. Nigam, M. Siegler, R. Stockton, and T. Tomokiyo. Deriving marketing intelligence from online discussion. In *Proc. 11th SIGKDD*, pages 419–428, Chicago, Illinois, USA, August 2005.
- [7] N. Kushmerick. Wrapper induction: efficiency and expressiveness. *Artificial Intelligence*, 118:15–68, 2000.
- [8] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th ICML*, pages 282–289, Williams College, Williamstown, MA, USA, June 2001.
- [9] K. Lerman, L. Getoor, S. Minton, and C. Knoblock. Using the structure of Web sites for automatic segmentation of tables. In *Proc. SIGMOD*, pages 119–130, Paris, France, June 2004.
- [10] K. Lerman, S. Minton, and C. Knoblock. Wrapper maintenance: A machine learning approach. *Journal of Artificial Intelligence Research*, 18:149–181, 2003.
- [11] D. Pinto, A. McCallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields. In *Proc. 26th SIGIR*, pages 235–242, Toronto, Canada, July 2003.
- [12] H. Poon and P. Domingos. Joint inference in information extraction. In *Proc. 22nd AAAI*, pages 913–918, Vancouver, Canada, July 2007.
- [13] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62 (1-2):107–136, 2006.
- [14] P. Singla and P. Domingos. Discriminative training of markov logic networks. In *Proc. 20nd AAAI*, 2005.
- [15] P. Singla and P. Domingos. Entity resolution with Markov logic. In *Proc. 6th ICDM*, pages 572–582, Hong Kong, China, December 2006.
- [16] Y. Wang, J.-M. Yang, W. Lai, R. Cai, L. Zhang, and W.-Y. Ma. Exploring traversal strategy for Web forum crawling. In *Proc. 31st SIGIR*, pages 459–466, Singapore, July 2008.
- [17] Y. Zhai and B. Liu. Web data extraction based on partial tree alignment. In *Proc. 14th WWW*, pages 76–85, Chiba, Japan, May 2005.
- [18] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise networks in online communities: structure and algorithms. In *Proc. 16th WWW*, pages 221–230, Banff, Alberta, Canada, May 2007.
- [19] S. Zheng, R. Song, J.-R. Wen, and D. Wu. Joint optimization of wrapper generation and template detection. In *Proc. 13th SIGKDD*, pages 894–902, San Jose, California, USA, August 2007.
- [20] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. 2D conditional random fields for Web information extraction. In *Proc. 24th ICML*, pages 1044–1051, Corvallis, Oregon, USA, August 2005.
- [21] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. Simultaneous record detection and attribute labeling in Web data extraction. In *Proc. 12th SIGKDD*, pages 494–503, Philadelphia, PA, USA, August 2006.