Automatic Construction of Wrappers for Semi-structured Documents

練 偉 業
Lin Wai-yip

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Systems Engineering and Engineering Management

摘要

信息抽取的目的是從文字檔案中辨認出特定或有用的文字片段。隨著近年萬維網及電子郵件的普及，網上電子化的信息激烈增長。但是，大部份的網上信息都是以半結構化的格式存在，因此不能利用標準查詢語言去查詢這些資料。以致近來有不少科學研究致力探討信息抽取的工作及信息整合系統。

我們發展了一個信息抽取系統「HISER」來處理半結構化的文件。依靠著用户提供的少數訓練實例，「HISER」能夠為特定的信息源自動學習一個層次記錄結構及抽取規則。在層次記錄結構的學習中，我們會自動為信息源發掘一個層次記錄結構，它代表了那些出現在某信息源內的屬性項的關係。隨後，對於在層次記錄結構中的每個節點，抽取規則會自動被歸納。這設計能夠處理缺掉的屬性項、多重值的屬性項及屬性項出現的不同次序。實驗結果顯示，我們的「HISER」系統能夠在沒有使用複雜的自然語言處理技術的情況下，準確地從半結構化文件中抽取適當的信息。

我們亦探討了使到某信息源的wrapper適用於其他未見過的信息源的問題，爲了達到這個目的，我們利用一個機器學習的技巧 — 支援向量機器(SVM)來自動從未見過的信息源中標籤訓練實例。 實驗結果顯示， 這技巧能從未見過的信息源中準確地標籤訓練實例。

Abstract

The goal of information extraction (IE) is to identify specific or useful text fragments from a textual document. The amount of information available electronically such as the World-Wide Web (WWW or Web), emails, and newsgroups has been growing tremendously in recent years. Much of the data available is in semi-structured format that cannot be queried using standard query languages. There has been much interest in Internet IE tasks and information integration systems.

We have developed an information extraction system, known as HISER, to handle semi-structured documents. Based on a few user-labelled training examples, HISER can automatically learn a hierarchical record structure and extraction rules tailored to an information source. In hierarchical record structure learning, we try to automatically discover the record structure that represents the relationship among the attribute items for the records appeared in an information source. Extraction rules are then induced for each node in the hierarchical record structure to handle the extraction task. HISER can handle missing attribute items, multi-valued attribute items, and attribute items appearing in unrestricted orders. Experiments show that our HISER system is able to accurately extract information from semi-structured documents without using sophisticated natural language processing techniques.

We also investigate the problem of adapting a learned wrapper to unseen information sources. To achieve this goal, we develop a technique to automate the process of annotating training instances for unseen information sources based on a machine learning technique, Support Vector Machines (SVM). Experiments show that it can accurately annotate the training examples of the target attribute items for unseen information sources.

Acknowledgments

There are many people whom I would like to thank for their support and contributions to this research. Firstly, I would like to express my sincere gratitude to my research advisor, Prof. Wai Lam. His constant encouragement and advises contributed a great deal in this research. I would also like to thank Prof. Kai Pui Lam and Prof. Chun Hung Cheng for their helpful suggestions and comments on improving the quality of my work.

Next, I would like to express my deep gratitude to Kathy and my family for their support and encouragement. Thanks to my best friends, Patrick, Philip, Connie, Sally and Fan in the Department of Systems Engineering and Engineering Management, CUHK. Their encouragement, companionship, support and help made my life in CUHK delightful and unforgetable.

Lin Wai Yip.

June 2001.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter gives a brief introduction of information extraction and brings out some key issues of information extraction from semi-structured documents. We have developed a novel framework, known as HISER (HIerarchical record Structure and Extraction Rule learning) [46] to handle semi-structured documents. The motivation and contributions of our research are discussed.

1.1 Information Extraction

The goal of information extraction (IE) is to identify specific or useful text fragments from a textual document such as a newswire article or a Web page [17]. IE aims at transforming unstructured text or semi-structured text into structured data. The extracted data can be stored in a database. The information can then be queried using standard database query languages or used for other intelligent processing without having to deal with the raw text documents. For example, extracting information from Web vendors allows online comparison-shopping agents to find the best bargain [18, 69]. IE systems seem to be a promising way to handle different types of text documents ranging from highly structured text to free text [48]. A key element of such IE systems is a set of extraction rules or patterns, tailored to a particular document collection, that can identify the appropriate fragments of the document.

For free text, a conventional IE problem is to fill template slots from natural language texts. A typical example of extraction from free text is the "Latin American terrorism" domain used in the Message Understanding Conferences (MUCs) [4, 5]. The target fields in this domain are the perpetrator names, victim names, instruments and location of attack from a collection of newswire articles on Latin American terrorism. Another example is the Question-Answering track in the Text REtrieval Conference (TREC) [62, 64] that aims at finding a short phrase or sentence that precisely answers a user's question from a text collection. To extract relevant information from free text, information retrieval and linguistic techniques are usually needed.

For structured text, the information is organized in highly-structured and rigid format such as tabular form. The contents can be correctly extracted based on a fixed ordering of the relevant contents and some uniform syntactic clues such as labels and mark-up tags that delimit the strings to be extracted [13, 30, 40].

Unlike free text or structured text, semi-structured text is ungrammatical and does not follow any rigid format. An example of such kind of text is a Web page, for which grammar and good style are often sacrificed. In addition, the attribute items of interest may be missing, contain multiple values, or appear in unrestricted orders. These features make information extraction from semi-structured texts a challenging task.

1.2 IE from Semi-structured Documents

The amount of information available electronically such as the World-Wide Web (WWW or Web) has been growing dramatically in recent years. Integrating data may significantly increase the utility of the Web and create many new applications. For examples, integrating information from Web vendors allows online comparison-shopping agents [18, 69] to find the best bargain for the users. Another example is the use of the information extracted from the Web to aid the discovery of prediction rules [26, 50] by using data-mining techniques and construct the schema for semi-structured data [27, 51].

Much of the data available is in the form of semi-structured documents that cannot be queried using standard query languages and thus not accessed by other applications. There has been much interest on Internet IE tasks and information integration systems [3, 13, 38, 45]. Most documents on the Web can be regarded as semi-structured documents. For example, HTML [66] documents contains mark-up tags and some content with free text between the tags. Moreover, the tag organization is usually structured, for example, in a hierarchical fashion. The structured tags together with free text constitute the essential content of semi-structured documents. A typical problem is that mark-up tags define how the content is to be displayed in a browser, but provide virtually no insight into their semantic meaning. It makes information extraction from semi-structured documents a challenging task.

Browsing and keyword searching are two paradigms [8] for retrieving semi-structured Web data. Despite the fact that they are intuitive, they present some limitations and restrictions. The major limitations of browsing as a search technique are that it is not cost-effective to follow huge amount of hyperlinks, and it is easy to get lost in the hyperspace. Keyword searching returns vast amount of data, which cannot be easily handled by the users. Although new standards such as Extensible Markup Language (XML) [65] will simplify Internet information extraction tasks, the technique of wrapper induction will continue to play an important role. The reason is that

XML requires Web information sources to accept such standards. Still, large amount of data available electronically does not follow such standards. So, XML will make wrapper construction simpler but will not eliminate the need of wrappers.

A number of different methods have been proposed to extract relevant information from semi-structured data such as Web documents. Unlike many natural language processing (NLP) systems, their works do not deal with in-depth understanding of the content of the text-based information. These methods rely on the syntactic structure identification marked by mark-up tags. Many of these techniques make use of wrappers. A wrapper usually contains some rules or extraction patterns that are able to extract attribute items from a document collection written in a specific format. The extraction rules comprise some contextual patterns appearing just before and just after the target text field.

Traditionally, wrappers are constructed manually [13, 30]. The disadvantages of hand-coded wrappers are many. They are tedious to create, costly and time-consuming to build and maintain, and they require high level of expertise. Nevertheless, wrappers have been constructed with some degree of automation [3, 6]. There has been substantial research on specialized programming languages and graphical user interface to assist manual construction of wrappers. Automatic wrapper can be constructed by using machine learning techniques. In automatic wrapper construction, wrappers are learned from attribute annotations provided by users. These annotations are treated as training samples or examples. Wrapper induction generates delimiter-based extraction rules that do not rely on sophisticated NLP techniques.

A number of wrapper learning methods [31, 40, 49] have been proposed. However, many of them can only handle documents with simple structures. Some methods require the attribute items to be extracted be presented in a fixed order. Some methods cannot handle missing attribute items. Most methods would fail for attribute items organized in a hierarchical fashion. Existing wrapper learning methods need various forms of assumptions or description about the document structure. In reality, the internal structure and the record boundaries [22] of many semi-structured information sources such as Web documents are not known in advance.

Although there has been some progress on wrapper induction recently, portability is another challenge to automated wrappers. Many semi-structured information sources, such as Internet sites, often have their own layout and formatting regularities. Furthermore, the layout may change from time to time. These factors lead to difficulties in maintaining and porting wrappers. Many existing works only tackle the problem of wrapper learning for information extraction from the same source. Users are required to provide training examples for each information source when the wrapper is constructed initially, or when the formatting regularities are altered.

1.3 Thesis Contributions

To cope with the above problems and minimize the user effort involved, we have developed a novel system, known as HISER (HIerarchical record Structure and Extraction Rule learning) [46]. Based on a few user-labelled training examples, HISER can automatically learn a hierarchical record structure and extraction rules tailored to an information source. To further reduce the user effort of providing training examples, HISER is able to adapt a learned wrapper to unseen information sources.

For automatic wrapper construction, HISER employs a two-stage learning task, namely, hierarchical record structure learning and extraction rule induction. In hierarchical record structure learning, we try to automatically generate a representation of the relationship among the attribute items for the records appeared in an information source. Based on the learned structure, extraction rules will be induced for each node to handle the extraction task. Instead of constructing a single extraction rule that takes into accounts all possible attribute item orderings and arbitrary levels of embedded data, novel extraction rules are constructed to deal with the tasks of extracting each attribute item from its parent in the hierarchical record structure. Our design can handle missing attribute items, multi-valued attribute items, and attribute items appearing in unrestricted orders.

In the extraction rule induction process, HISER learns expressive extraction rules that make use of limited syntactic and semantic information. We incorporate both syntactic generalization and semantic generalization so that highly accurate and more expressive rules can be learned. In most of the previous wrapper learning approaches, only the surrounding contexts of the target attribute items are considered when constructing the wrappers. In addition to the surrounding contexts, the content of the attribute item itself is considered to enrich the expressiveness of the rules. Our HISER system is able to accurately extract information from semi-structured documents without using sophisticated NLP techniques such as semantic tagging or part-of-speech tagging. While some of our description may use Web documents as examples, the technique does not confine to Web environment. It can be readily applied to other semi-structured information sources.

Another feature of our HISER system is to adapt a learned wrapper to unseen information sources. To achieve this goal, we develop a wrapper adaptation approach based on a machine learning paradigm, known as Support Vector Machines (SVM). Novel feature selection and candidate seeking techniques are developed to automate the process of annotating training instances for unseen information sources. The layout and formatting of semi-structured documents differ among different information sources and thus the extraction rules of one information source usually cannot be applied to other information sources. We observe that the semantic content of the attribute items of interest remains largely unchanged. In addition, the surrounding tags of some attribute items are often similar across different information sources. These regularities enable HISER to tackle the problem of wrapper maintenance and adaptation because those previously-labelled training examples can be an approximation to the training examples that a user may provide for new unseen information sources. The patterns as well as the characteristics of the surrounding contexts and the content of the previously-labelled training examples are learned. Eventually, the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources can be predicted. We present encouraging experimental results of applying HISER to a range of semi-structured information sources for wrapper learning as well as wrapper adaptation.

1.4 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 reviews some related work on information extraction from semi-structured documents. Chapter 3 provides an overview of the proposed information extraction system, HISER. Chapter 4 describes the automatic hierarchical record structure construction. Chapter 5 describes the representation of extraction rule and the rule induction algorithm. Chapter 6 reports the experiments on the wrapper learning of HISER on three semi-structured document domains. Chapter 7 describes the methodology of adapting a learned wrapper to unseen information sources. Chapter 8 reports the experiments on the wrapper adaptation of HISER. Finally, we present our conclusions and future work in Chapter 9.

Chapter 2

Related Work

This chapter reviews and discusses some related research work and similar systems on information extraction from semi-structured documents. With the tremendous amount of information that is now available, advanced tools or systems have been developed to extract relevant data from different kinds of information sources.

2.1 Existing Approaches

Information extraction systems seem to be a promising way to deal with certain types of text documents ranging from highly structured to free text. A key element of such IE systems is a set of extraction rules or patterns that identify relevant information to be extracted. However, IE systems are costly and time-consuming to build and maintain. As an alternative, IE systems are constructed by employing machine learning methods [48].

To extract relevant information from free text as in the Message Understanding Conferences (MUCs) [4, 5] and handle Question-Answering problem in the Text REtrieval Conference (TREC) [64], natural-language processing techniques are required. Natural-language processing (NLP) systems [11, 33, 37, 52, 54, 61, 62] apply linguistic techniques such as part-of-speech tagging, parsing, lexical semantic tagging and syntactic analysis.

As the amount of information available on the World-Wide Web has been growing dramatically in recent years, there has been much interest on software agents and information integration [1, 42, 68]. Integrating the data from diverse sources may significantly increase the utility of the Web and create many new applications. For example, integrating information from Web vendors allows online comparison-shopping agents [18] to find the best bargain for the users. Much of the data available on the Web is in semi-structured format that cannot be queried using standard query languages and thus the data is not easily accessible by other applications. There has been much interest in Internet IE tasks and information integration systems [3, 13, 38, 45].

A variety of research has been devoted to issues on directly querying semi-structured data from Web information sources in a database-like fashion

[2, 7, 9, 21, 41, 47]. These approaches are concerned with the development of data models like OEM (Object Exchange Model) and query languages for semi-structured data, defining formal semantics for such query languages, and efficiently implementing these languages.

Many different methods have been proposed to extract relevant information from semi-structured documents. Many of these techniques make use of wrappers [6, 13, 24, 31, 40, 53, 49]. A wrapper usually contains some rules or extraction patterns that are able to extract attribute items from a document collection. Traditionally, wrappers are constructed manually [13, 30]. There has been substantial research on specialized programming languages and graphical user interface to assist manual construction of wrappers [3, 20, 28, 30, 32, 55, 59]. These methods rely on human experts rather than machine learning techniques to construct wrappers. But manual construction is costly, time-consuming, error-prone, and labor-intensive. Recently, a number of different methods have been proposed to automatically construct wrappers for extracting relevant information from semi-structured Web documents.

WIEN [40] is an extraction system which performs automatic wrapper induction by considering different wrapper classes using a machine learning technique. Compared to the manual wrapper construction, WIEN has the advantage of dramatically reducing both the time and effort required to build a wrapper for an information source. But it uses only the delimiters that immediately preceding and following the actual data. It assumes that there is a unique multi-slot rule that can be applied for all documents in a given Web site, and does not allow the use of semantic classes. As a result, it fails to handle sites with missing attribute items, multiple attribute values, and different attribute item permutations commonly found in Web documents. Besides, WIEN learns the separators of the attribute items by searching common prefixes or suffixes at the character level, it needs quite a lot of training examples and the rule expressiveness is quite limited.

SoftMealy [31] uses a wrapper induction algorithm that expresses the extraction rules as finite transducers. The rules are more expressive than those of WIEN because they contain wild-cards and they can handle missing attribute items, multi-valued attribute items, and attribute items appearing in unrestricted orders. In order to deal with missing attribute items, multi-valued attribute items, and attribute items appearing in unrestricted orders, SoftMealy needs to have training examples that include all possible combinations of the attribute items. Another limitation is the inability to use delimiters that do not immediately precede and follow the relevant items.

WHISK [60] is an information extraction system which can handle highly structured texts as well as free texts. Depending on the nature of the text, WHISK generates context-based patterns and delimiter-based patterns for free text and structured texts respectively. WHISK allows the use of syntactic analysis and semantic tagging and it is capable of using multiple landmarks for multi-slot extraction rule. Thus the extraction rules are obviously more expressive than WIEN's. But extracting a particular attribute item is not independent of other relevant attribute items. As a result, the form of information it can handle is rather constrained. It fails to handle records with different number of attribute items or attribute items organized in hierarchical structure.

SRV [23] generates first-order logic extraction patterns based on attribute-value tests and the relational structure of the documents. It does not require prior syntactic analysis. The rule representation is expressive and able to incorporate orthographic features and other information such as semantic class and part of speech when available. However, the rules are single-slot and applying an extraction rule to unseen documents would involve considering a huge number of candidate phrases.

RAPIER [10] learns single-slot extraction pattern rules that use limited syntactic information and semantic class information. The extraction pattern consists of three distinct slots: the target phrase itself, a pre-filler pattern before the target phrase and a post-filler patterns after it. The pre-filler and post-filler play the role of left and right delimiters of the item, while the filler pattern of the target phrase describes the structure of the information to be extracted. RAPIER's rules specify an ordered list of items to be matched for each of these fields, with lexical constraints, semantic constraints, and part of speech constraints on each item. The main disadvantages are that the extraction rules are single-slot and it requires absolute orderings of tokens by using an ordered list of items for each field.

In short, RAPIER and SRV can generate only single-slot rules, which is a serious limitation for a significant number of domains. They treat each attribute item as a separate task and target for learning, without addressing the problem of combining the attribute items to produce a complete record. Besides, they need huge amount of training examples. But RAPIER and SRV are capable of imposing a richer set of constraints than WHISK by using orthographic features, token length and link grammars.

STALKER [49] is a wrapper learning method that can extract content from documents with hierarchical structure. The extraction rules are single-slot. It uses the embedded content tree to group together the individual attribute items to assemble a multi-slot record. Besides, each item is extracted independently of its siblings in the embedded content tree. A major disadvantage is the requirement of providing a description of the structure of the document.

Some methods focus on extracting information from tables in Web documents. Lim and Ng [44] constructed the content tree of the data contents in a given HTML table without requiring the internal structure of the table to be known beforehand. The content tree captures the intended hierarchy of the data contents in the table. This approach can be used by existing wrappers and integrators for extracting hierarchical data from HTML tables. Wang et al. [67] proposed a semantic search approach capable of extracting information from general tables. Semantic ontology allows it to read tables in the same knowledge domain with different layouts. In addition, a system of layout syntax and a set of transformation rules are defined to transform tables into databases without losing their semantic meanings. The major disadvantage is that these approaches can only handle data formatted in HTML tables.

Freitag introduced an approach, called BWI [24], to build a trainable information extraction system. Like many wrapper induction techniques, it learns relatively simple contextual patterns identifying the beginning and ending of relevant text fields. BWI makes use of boosting technique to improve the performance of a simple machine learning algorithm. It repeatedly reweights the training examples so that subsequent patterns handle training examples missed by previous rules.

Recently, some methods make use of the Hidden Markov models (HMMs) for information extraction tasks. Hidden Markov modeling is a statistical machine learning technique. Seymore et al. [58] explored the use of HMMs to learn model structure from data and how to make the best use of labelled and unlabelled data. Freitag and McCallum [25] demonstrated the ability of shrinkage to improve the performance of HMMs for information extraction. Methods using HMMs need considerable amount of training examples, which may not be suitable for Web information extraction.

Adaptability is another challenge to automatically building wrappers for Internet sites. Kushmerick introduced RAPTURE [39], a domain-independent and heuristic algorithm for solving the wrapper verification problem. It uses well-motivated heuristics to compute the similarity between a wrapper's observed and expected output. This solution can partially solve the problem of wrapper maintenance.

2.2 Limitations of Existing Approaches

Existing wrapper learning methods need various forms of assumptions or description about the relationship among the attribute items of interest. In reality, the internal structure of many semi-structured information sources such as Web documents is not known in advance. In many existing methods, users are required to provide extra information about the document structure, which may not be effective.

SRV and RAPIER generate single-slot extraction rules, but there is no specific technique to group the individual attribute items together to produce a complete record, which is a serious limitation for a significant number of domains. WIEN, SoftMealy and WHISK construct a single extraction rule that takes into accounts all possible attribute item orderings and arbitrary levels of embedded data. These methods can handle semi-structured documents with attribute items organized in a simple structure only. As a result, they are ineffective in handling missing attribute items, multiple attribute values, and various attribute item permutations commonly found in semi-structured documents.

WIEN, SoftMealy, and approaches discussed in [12, 14, 15, 21, 29] specifically focus on Web documents only. Some methods [44, 67] just handle data formatted in HTML tables only. The extraction capability of all these methods is rather restricted.

Many existing works only tackle the problem of wrapper learning for information extraction from the same source. Users are required to provide training examples for an information source when the wrapper is constructed, or when the formatting regularities are altered. Another limitation is that some existing approaches such as SRV, RAPIER and methods using HMMs, need huge amount of training examples, which may not be feasible and efficient.

2.3 Our HISER Approach

To cope with the above problems and minimize the burden of users, we develop a novel system, known as HISER (HIerarchical record Structure and Extraction Rule learning) [46]. Based on a few user-labelled training examples, HISER can automatically learn a hierarchical record structure and highly accurate extraction rules tailored to an information source without any prior processing or post-processing. To further reduce the burden on users of providing training examples, HISER is able to adapt a learned wrapper to unseen information sources.

Our HISER approach employs a two-stage learning task for wrapper learning, namely, hierarchical record structure learning and extraction rule induction. First, we try to automatically generate a representation of hierarchical structure for the records appearing in an information source. Extraction rules will then be induced for each node in the learned structure to handle the extraction task. Instead of constructing a single extraction rule that takes into account all possible attribute item orderings and arbitrary levels of embedded data, simpler extraction rules are constructed to deal with the tasks of extracting each attribute item from its parent in the hierarchical record structure. In the extraction rule induction process, we incorporate both syntactic generalization and semantic generalization so that highly accurate and more expressive rules can be learned. In addition to the surrounding contexts, the content of the attribute item itself is considered to enrich the expressiveness of the rules.

Our extraction rules are single-slot. Unlike SRV and RAPIER, the single-slot nature of our extraction rules does not possess a limitation because HISER uses the hierarchical record structure to group individual attribute items together. HISER is capable of extracting data from documents that contain arbitrarily complex combinations of embedded lists and items. In addition, each attribute is extracted independently of its siblings in the hierarchical record structure. The different orderings of the attribute items does not require one extraction rule for each existing permutation of the attribute items to be extracted.

To further reduce the user effort of providing training examples, HISER is able to adapt a learned wrapper to unseen information sources. To achieve this goal, we develop a wrapper adaptation approach based on a machine learning paradigm, known as Support Vector Machines (SVM). Novel feature selection and candidate seeking techniques are developed to automate the process of annotating training examples for unseen information sources. The patterns as well as the characteristics of the surrounding contexts and the content of the attribute instances are learned. Eventually, the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources can be predicted. Experimental results show that HISER offers effective performance for automatic wrapper construction and encouraging results for wrapper adaptation.

Chapter 3

System Overview

The system framework of our HISER system (HIerarchical record Structure and Extraction Rule learning) [46] will be presented in this chapter. Illustrative examples will be also provided.

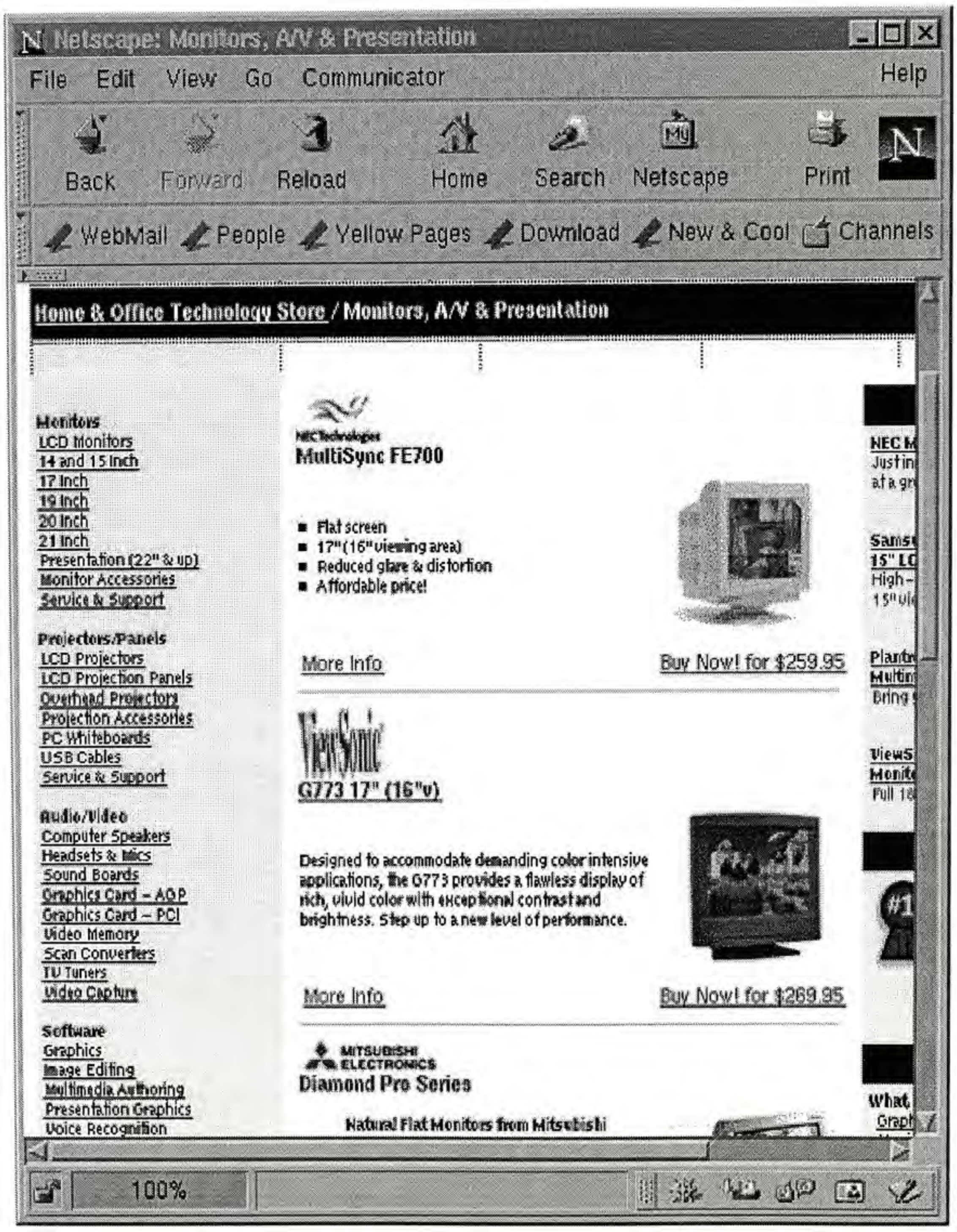
3.1 HIerarchical record Structure and Extraction Rule learning (HISER)

Our HISER framework has a wrapper learning component. The wrapper learning component is composed of two modules. The first module is to induce a hierarchical record structure for an information source automatically. The second module is an extraction rule induction algorithm. Once a wrapper is learned, it can be used for extracting information for a particular information source. Apart from the wrapper learning component, HISER has a wrapper adaptation component. This wrapper adaptation component is able to automatically annotate training examples for unseen information sources. This adaptation component will be invoked if we adapt a wrapper to unseen information sources.

In order to extract the items of interest from a particular information source, our wrapper uses the hierarchical record structure and sets of extraction rules. For each node in the hierarchical record structure, our wrapper needs extraction rules to extract that particular node from its parent. For a list node, the extraction rules will be first applied iteratively to extract the list elements. After that, the extraction rules of the children will be applied to the list elements to extract individual attribute items. A typical user starts with annotating samples of attribute items by highlighting some texts on the training page. This annotation process could be done via a graphical user interface. HISER takes these samples as input and conducts the wrapper learning process. The output of learning will be a wrapper composed of a hierarchical structure description of the document records and sets of extraction rules. The information embodied in a wrapper can be viewed as attribute item extraction knowledge and it can be used for extracting records and attribute items from this particular information source.

Figure 3.1 shows an example of a Web page containing information of electronic appliances in a product catalog. The corresponding excerpt of this HTML page is shown in Figure 3.2. The attribute items of interest are the name, description, feature, and price of the product. As shown in Figure 3.1, two products can be extracted. The first product contains a name "MultiSync FE700", a list of features "Flat screen", "17$''$ (16$''$ viewing area)", "Reduced glare & distortion" and "Affordable price!", and price "259.95". Another product contains a name "G773 17$''$ (16$''$v)", description "Designed to accommodate demanding color intensive applications ...... new level of performance." and price "269.95".

Another kind of semi-structured document is shown in Figure 3.3. It is a seminar announcement from the Chinese University of Hong Kong (CUHK). Unlike HTML document, it looks like raw texts. The contents of interest are not enclosed with mark-up tags, they appear as free text with no clear separator like HTML tags. Extracting contents from this kind of documents is more challenging. The attribute items of interest are the topic, speaker, venue, date, start time and end time of the seminar. Refer to the seminar announcement in Figure 3.3, the topic is "Generalized Steiner Problems and Other Variants", the speaker is "Professor Moshe Dror", the venue is "Rm 122, HSH Engineering Building CUHK", and it is held from "4:00pm" to "5:00pm" on "May 18th , 2000, Thursday".


Netscape: Monitors, A/V & Presentation
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print
WebMail People Yellow Pages Download New & Cool Channels
100%

Home & Office Technology Store / Monitors, A/V & Presentation

Monitors
LCD Monitors
14 and 15 Inch
17 Inch
19 Inch
20 Inch
21 Inch
Presentation (22" & up)
Monitor Accessories
Service & Support

Projectors/Panels
LCD Projectors
LCD Projection Panels
Overhead Projectors
Projection Accessories
PC Whiteboards
USB Cables
Service & Support

Audio/Video
Computer Speakers
Headsets & Mics
Sound Boards
Graphics Card - AGP
Graphics Card - PCI
Video Memory
Scan Converters
TV Tuners
Video Capture

Software
Graphics
Image Editing
Multimedia Authoring
Presentation Graphics
Voice Recognition

NEC Technologies
MultiSync FE700

- Flat screen
- 17"(16" viewing area)
- Reduced glare & distortion
- Affordable price!

More Info Buy Now! for $259.95

ViewSonic
G773 17" (16"v)

Designed to accommodate demanding color intensive applications, the G773 provides a flawless display of rich, vivid color with exceptional contrast and brightness. Step up to a new level of performance.

More Info Buy Now! for $269.95

MITSUBISHI ELECTRONICS
Diamond Pro Series

Natural Flat Monitors from Mitsubishi

Figure 3.1: A sample online electronic appliance catalog.

```
<TR><TD VALIGN=TOP COLSPAN=2><FONT FACE="Geneva, Arial, sans-serif" SIZE=2>
<img src="/nec.gif" WIDTH=47 HEIGHT=24 ALT="NEC" BORDER=0><BR>
<FONT COLOR=#006633><B>MultiSync FE700</B></FONT></TD>
</TR><TR>
<TD><FONT FACE="MS Sans Serif, Geneva, Arial, sans-serif" SIZE=1>
<LI>Flat screen
<LI>17" (16" viewing area)
<LI>Reduced glare & distortion
<LI>Affordable price!
</FONT></TD>
<TD VALIGN=TOP ALIGN=RIGHT>
<img src="/144209.gif" WIDTH=100 HEIGHT=75 BORDER=0 ALT="">
</TD></TR>
<TR><TD COLSPAN=2>
<TABLE WIDTH=100% CELLPADDING=0 CELLSPACING=0>
<TR><TD ALIGN=RIGHT>
<A HREF="/cgi-bin/shoplist_q?nitems=1&citemno=0000144209" STYLE="COLOR:#CC0000">
<FONT SIZE=-1 COLOR=#CC0000>
Buy Now! for $259.95</FONT></TD><BR>
</TR></TABLE></TD></TR>
<TR><TD COLSPAN=2>
<img src="/grayline.gif" WIDTH=305 HEIGHT=1 BORDER=0 ALT="">
</TD></TR>

<TR><TD VALIGN=TOP><FONT FACE="Geneva, Arial, sans-serif" SIZE=2>
<img src="/viewsoni.gif" WIDTH=48 HEIGHT=37 ALT="Viewsonic" BORDER=0><BR>
<A HREF="/prod_page.html?key=0000106019&nonce=guest" STYLE=COLOR:#006633>
<FONT COLOR=#006633><B>G773 17" (16"v)</B></FONT></A></FONT></TD>
</TR><TR>
<TD><FONT FACE="MS Sans Serif, Geneva, Arial, sans-serif" SIZE=1>
Designed to accommodate demanding color intensive applications, the G773 provides a
flawless display of rich, vivid color with exceptional contrast and brightness. Step
up to a new level of performance. </FONT></TD>
<TD VALIGN=TOP ALIGN=RIGHT>
<img src="/106019.gif" WIDTH=100 HEIGHT=75 BORDER=0 ALT="">
</TD></TR>
<TR><TD COLSPAN=2>
<TABLE WIDTH=100% CELLPADDING=0 CELLSPACING=0>
<TR><TD ALIGN=RIGHT>
<A HREF="/cgi-bin/auth/shoplist_q?nitems=1&citemno=0000106019" STYLE="COLOR:#CC0000">
<FONT SIZE=-1 COLOR=#CC0000>
Buy Now! for $269.95</FONT></A></TD><BR>
</TR></TABLE></TD></TR>
```

Figure 3.2: Excerpt of the HTML page shown in Figure 3.1.

```
                * * * Seminar * * *

     Dept. of Systems Engineering & Engineering Management
               Chinese University of Hong Kong

------------------------------------------------------------------------

Title   :       Generalized Steiner Problems and Other Variants

Speaker :       Professor Moshe Dror
                The University of Arizona

Venue   :       Rm 122, HSH Engineering Building CUHK

Date    :       May 18th , 2000, Thursday

Time    :       4:00pm - 5:00pm

Abstract        :

In this paper, we examine combinatorial optimization problems by
considering the case where the set N (the ground set of elements) is
expressed as a union of a finite number of m nonempty distinct subsets
N_{1},...,N_{m}. The term we use is the generalized Steiner problems
coined after the Generalized Traveling Salesman Problem. We have
collected a short list of classical combinatorial optimization problems
and we have recast each of these problems in this broader framework in
an attempt to identify a linkage between these "generalized" problems.
......

Biography       :

Moshe Dror is a Professor in the MIS Department at the College of
Business and Public Administration, University of Arizona, where he also
served for two years as the Head of Decision Sciences (an independent
academic unit of 7-9 faculty which has been subsequently  consolidated
into MIS). He received ......

Enquiry : Dr. H. Yan (2609-8329)
E-mail : yan@se.cuhk.edu.hk
```

Figure 3.3: A sample seminar announcement from CUHK.

3.2 Hierarchical Record Structure

The first module of HISER attempts to automatically learn a hierarchical record structure for an information source based on the attribute item samples given by users. The hierarchical record structure is a representation of the relationship among the attribute items in an information source. Users only need to specify the attribute items of interest and then label them in the training page. They do not need to explicitly specify the hierarchical record structure in the information source or the relationship among the attribute items. The documents can contain single record or multiple records. Based on the target attribute items, a common hierarchical record structure as shown in Figure 3.4 will be inferred automatically for the product catalog shown in Figure 3.1. The representation of this hierarchical record structure indicates that each product contains a product name, a list of features, a description, and a price. These attribute items can appear in different orders. The details of this module will be discussed in Chapter 4.

3.3 Extraction Rule

The second module is the extraction rule induced. Extraction rules will be learned for each node in the hierarchical record structure. An extraction rule consists of three parts: the left pattern component, the target component,

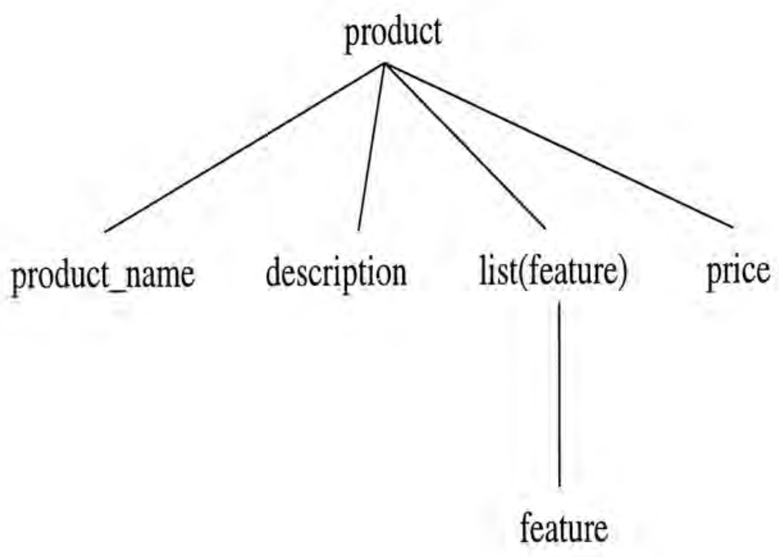


Figure 3.4: Hierarchical record structure S for the product information shown in Figure 3.1.

and the right pattern component. The left pattern component and right pattern component play the roles of left and right delimiters precede and follow the target attribute item respectively, while the target component of the attribute item describes the information to be extracted.

For example, to extract the product price from the product catalog shown in Figure 3.1, the following extraction rule will be induced:

left pattern component:

ScanUntil("<FONT SIZE=-1 COLOR=#CC0000>"),

ScanUntil("$").

target component:

Contain(*Numeric*).

right pattern component:

ScanUntil("</FONT>"),

ScanUntil("</TD>"),

ScanUntil("<BR>").

The left pattern component and right pattern component consist of a sequence of token scanning instructions, *ScanUntil()*. They instructs the wrapper to scan the texts until a particular token is matched. After the token scanning instructions, *ScanUntil*("$") and *ScanUntil*("</FONT>") have been successfully executed, the start and end boundaries of the attribute item are marked. A token containing instruction, *Contain()*, in the target component instructs the wrapper to test if the text segment contains the content in the token containing instructions. In order to satisfy the rule, all the token scanning instructions in the left pattern component and right pattern component must be fulfilled. Then the content between the left and right pattern components can be extracted only if it satisfies the requirements of the target component. More details of the extraction rule representation and induction algorithm will be described in Chapter 5.

After the learning process completes, the learned extraction rules can be applied to extract individual attribute items of interest from unseen documents in the same information source. After that, the hierarchical record structure is used to group individual attribute items together to form complete records. Eventually, the extracted records can be returned as structured data.

3.4 Wrapper Adaptation

The third module is automatic annotation of training examples for unseen information sources. Portability is a major challenge for automated wrappers. Many information sources often have their own layout and formatting regularities which may change from time to time. These factors lead to difficulties in adapting wrappers to unseen information sources. Users are required to provide training examples for each source when the first time a wrapper is constructed, or when the formatting regularities are altered.

To alleviate this problem, our HISER system is able to adapt a learned wrapper to unseen information sources. We observe that the semantic content of the attribute items of interest remains largely unchanged. In addition, the surrounding tags of some attribute items are often similar across different information sources. These regularities enable HISER to tackle the problem of wrapper maintenance and adaptation because those previously-labelled training examples can be an approximation to the training examples that a user may provide for new unseen information sources.

To achieve this goal, we develop a technique to automate the process of annotations of training instances for unseen information sources. Support Vector Machines (SVMs), a machine learning technique, is employed to capture the characteristics of the contexts around target attributes as well as the content of the previously-labelled training examples. Eventually, the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources can be predicted. Once the training examples are annotated automatically, a suitable wrapper will be learned for the unseen information sources for extraction. The details of the adaptation module will be described in Chapter 7.

Chapter 4

Automatic Hierarchical Record Structure Construction

This chapter discusses the first module of our HISER system, which is used to induce a hierarchical record structure description for an information source. Automatic construction of hierarchical record structure is proposed in our approach.

4.1 Motivation

Existing wrapper learning methods need various forms of assumptions or description about the relationship among the attribute items of interest. In reality, the internal structure of many semi-structured information sources such as Web documents are not known in advance. Sometimes, users are required to provide extra information about the document structure, which may not be effective. In order to handle these problems, a hierarchical record structure is constructed automatically in HISER.

In hierarchical record structure learning, we try to automatically induce a representation of the relationship among the attribute items of interest in an information source. Users only need to specify the attribute items of interest and then label them in the training page. They do not need to explicitly specify the hierarchical record structure in the information source or the relationship among the attribute items. The documents can contain single record or multiple records. We focus on the record level instead of the whole document because those extraneous contents like advertisements and forms can be ignored.

Most of the previous approaches cannot handle semi-structured documents that contain missing attribute items, multi-valued attribute items, or different orderings of attribute items. In our approach, the hierarchical record structure can have arbitrarily many levels of embedded data, which is able to handle multi-valued attribute items. Moreover, there is no explicit ordering of the nodes in the structure at the same level. These features can handle missing attribute items and different orderings of attribute items in the records, which are quite common in semi-structured documents.

4.2 Hierarchical Record Structure Representation

We make use of a context-free grammar to express the kind of hierarchical record structure as follows:

$$S = (aS) \mid aS \mid SS \mid \epsilon$$

where S is a non-terminal representing a structure, a is a terminal representing an attribute item and ϵ represents an empty element. The bracket indicates a repeated pattern.

In fact, a hierarchical record structure is a tree-like structure in which the leaf nodes are the attribute items of interest. The root node in the structure represents a record which typically consists of a sequence of attribute items. An internal node in the structure represents a certain part of the content of its parent node. Node other than the root node or leaf nodes is regarded as a list, where the element under the list can be a leaf element or another list. Leaf elements are the attribute items of interest. In principle, the hierarchical record structure can have arbitrary levels of embedded data. There is no explicit ordering of the nodes in the structure. These features can handle missing attribute items and different orderings of attribute items in the records, which are quite common in semi-structured documents.

For example, the first two products in Figure 3.1 can be represented as:

$$S1 = (product\_name \quad (feature) \quad price)$$

$$S2 = (product\_name \quad description \quad price)$$

The bracket indicates a repeated pattern. Thus, *(feature)* denotes a list of attribute item "feature". At the same time, *S1* and *S2* can be visualized as tree structures as shown in Figure 4.1.

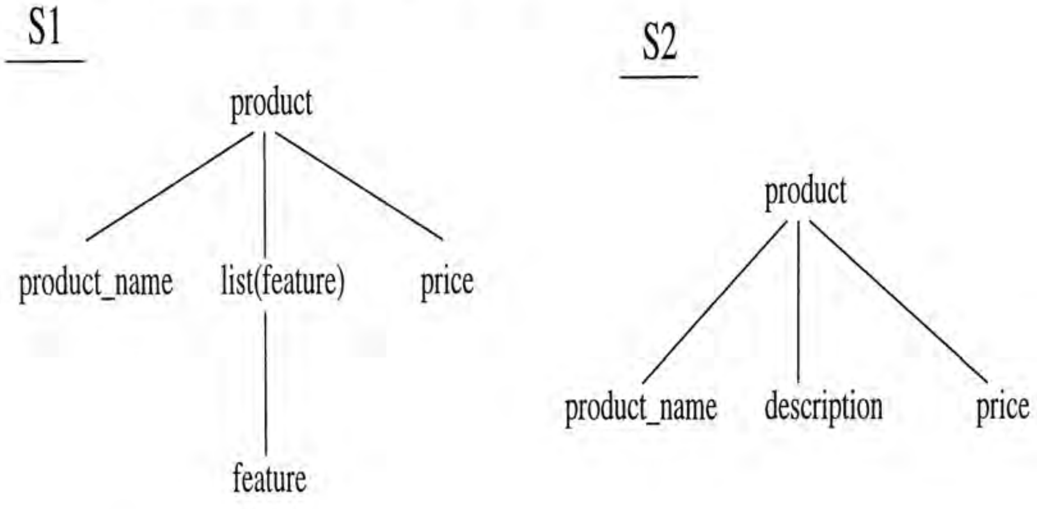


Figure 4.1: Hierarchical record structures *S1* and *S2* for the first two products shown in Figure 3.1.

Similarly, the seminar announcement in Figure 3.3 can be represented as:

$$S3 = (topic \quad speaker \quad venue \quad date \quad start\_time \quad end\_time)$$

At the same time, *S3* can be visualized as a tree structure as shown in Figure 4.2.

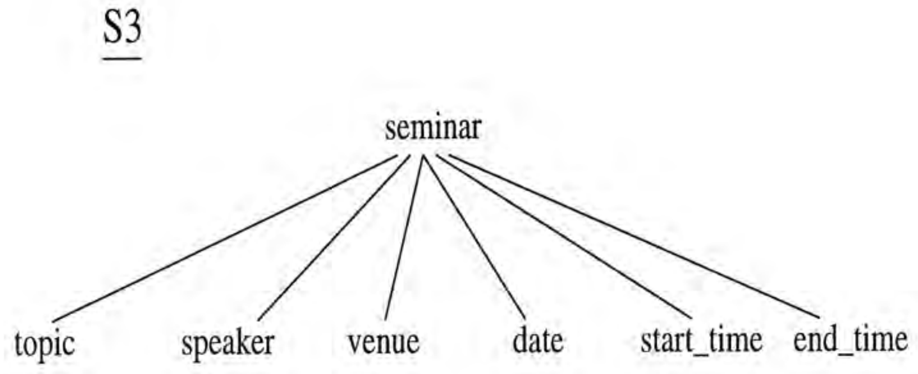


Figure 4.2: Hierarchical record structures *S3* for the seminar announcement shown in Figure 3.3.

4.3 Constructing Hierarchical Record Structure

Each textual document contains a sequence of fields or tokens. A field or token can be a word, number, punctuation, specific ASCII character, HTML tag if dealing with Web documents or domain specific contents like *price* and *product feature*. Before the learning process, users are required to label training samples or examples. This labelling process could be conducted via a graphical user interface. The underlying organization of the document is not visible to the users. A user-labelled example contains the name and the corresponding value of the target attribute item in the textual document. These examples are treated as training examples and collectively as the training set. Based on the user-labelled examples, individual hierarchical structure can be derived for each training record.

We apply the following criteria to determine the hierarchical record structure for a record. While our description targets Web documents, the technique can be readily applied to other semi-structured information sources.

1. Suppose x and y represent sequences of tokens. If x contains y, it implies that x depends on y and x is a container of y. Consequently, there is a hierarchical relation between x and y such that y is a children of x and x is a parent of y.

   Let the notation of "x depends on y" be $x \leftarrow y$. Dependency is anti-symmetric and transitive. By anti-symmetricity, if $x \leftarrow y$ and $y \leftarrow x$, then $x = y$. By transitivity, if $x \leftarrow y$ and $y \leftarrow z$, then $x \leftarrow y \leftarrow z$. In this case, x is an indirect container of z. The notion of dependency plays an important role in determining the hierarchical relationships among the objects in semi-structured data.

2. In addition to the dependency relationship among different objects, some HTML tags have the implicit indications of hierarchical relationship. HTML tags like UL, OL, LI, TABLE, TR, TD show some implicit indications on the hierarchy among the objects.

   For examples, the tag "<TABLE>" allows people to arrange data in tabular form. The data can be text, images, links, forms or other tables. The tag "<TR>" defines a container for a single row of table cells, and the tag "<TD>" defines a cell in a table that contains data. That means the data content enclosed with the tags "<TABLE>" and "</TABLE>" depends on the data content enclosed with the tags "<TR>" and "</TR>", while the data content enclosed with the tags "<TR>" and "</TR>" depends on the data content enclosed with the tags "<TD>" and "</TD>". This kind of HTML background knowledge can also be useful when determining the relationship among different objects.

3. If some particular attribute items occur together more than once, this repeated pattern implies that these attribute items can be grouped together to form a list in the record structure. The concept of list element can be used to handle multi-valued attribute items which are also commonly found in semi-structured documents.
   As shown in Figure 3.1, more than one feature are displayed for the first product. This repeated pattern of the feature can be grouped together to form a list of feature in the hierarchical record structure.

Based on the criteria listed above, the implicit individual hierarchical record structure for each record in the training set can be constructed automatically. After that, the individual hierarchical structures will be integrated to form a common hierarchical record structure. This common hierarchical record structure can represent the relationship among the attribute items for a particular information source. It should cover all the attribute items appearing in all individual structures so as to handle missing attribute items. In order to deal with multi-valued attribute items, the precedence of a list element is higher than that of a simple element in the hierarchical structure. Moreover, there is not a fixed ordering of the attribute items in the hierarchical record structure. The order of the attribute items can be changed arbitrarily so as to handle different orderings of attribute items.

The individual hierarchical record structures shown in Figure 4.1 will be integrated to a common hierarchical record structure as shown in Figure 4.3. This common structure can also be expressed as:

$$S = (product\_name \quad description \quad (feature) \quad price)$$

Although a list of features, i.e. *(feature)*, appears in *S1* but not *S2*, and only *S2* contains a description, eventually, a product name, a list of features, a description, and a price will appear in the common hierarchical record structure *S*. It indicates that each product in this information source contains a product name, a list of features, a description, and a price.

Extracting information based on the hierarchical record sturcture has the

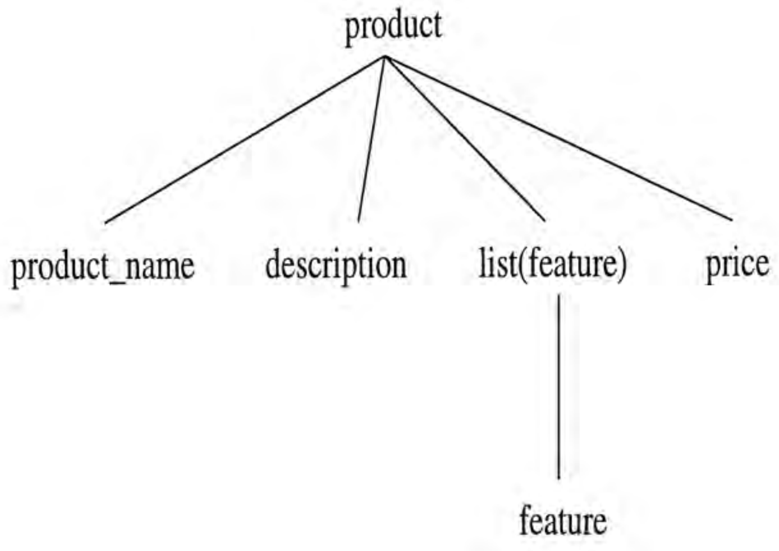


Figure 4.3: Resulting hierarchical record structure *S* for integrating *S1* and *S2*.

following main advantages. First of all, the extraction based on the hierarchical record structure allows HISER to wrap information sources that have arbitrary levels of embedded data. Besides, as each node is extracted independent of its siblings, this approach does not rely on a fixed ordering of the attribute items, and it can easily handle missing attribute items as well. Consequently, this approach turns a complex problem into several simpler tasks. Rather than constructing a single extraction rule that takes into accounts all possible attribute item orderings and arbitrary levels of embedded data, simpler extraction rules are constructed to deal with the tasks of extracting each attribute item from its parent in the hierarchical record structure.

Chapter 5

Extraction Rule Induction

This chapter discusses the second module of our HISER system, which is used to induce extraction rules for an information source. Extraction rules will be induced based on the hierarchical record structure described in Chapter 4. The representation and learning algorithm of the extraction rule are presented in this chapter.

5.1 Rule Representation

Based on the hierarchical record structure, extraction rules are induced for each node in a depth-first manner. A Web page is first tokenized to a sequence of tokens. For each node in the structure, our wrapper needs a set of rules for extracting that particular node from its antecedent. In fact, the extraction rules of the root node in the structure are used to extract individual records from semi-structured documents. These documents can contain single or multiple records. For a list node, the extraction rules will be first applied iteratively to extract the list elements. After that, the extraction rules of the children will be applied to the list elements to extract individual attribute items. Each attribute item is extracted independently among its siblings in the structure. As a final step, we use the hierarchical record structure to group together the individual attribute items to assemble multi-slot records. This approach can handle missing attribute items, multi-valued attribute items, and different orderings of the attribute items in the records.

First of all, two key concepts will be introduced to define extraction rules. They are landmarks and landmark automata. In the extraction rules described in Chapter 3, the argument of each token scanning instruction, *ScanUntil()*, is a landmark. A landmark is a token in the document or a wild-card. A group of *ScanUntil()* instructions that must be applied in a pre-established order represents a landmark automaton. A wild-card is a class of tokens. In our framework, both domain independent and domain specific wildcards are considered. We have six domain independent wild-cards. They include *Numeric*, *Alphabetic*, *AlphaNumeric Punctuation*, *HTML tag*, and *Control Character*. To further minimize the user effort involved, HISER can optionally allow domain specific semantic classes. These semantic classes are used to recognize the semantic contents tailor-made for a particular domain. For online product catalogs, users can choose to provide domain specific classes like *price* and *product feature.* For example, the lexicon for the product price stores the keywords, thesaurus and symbols such as "price", "sell", "$" and "buy".

Suppose we are dealing with semi-structured Web documents. Most of the contents are enclosed with HTML tags. HTML annotations define how the content is to be displayed for human browsing, but provides virtually no insight into their semantic meaning. For other semi-structured documents, the contents appear as free text with no clear separators or mark-up tags like HTML tags. In order to extract content accurately from semi-structured documents, in additional to syntactic tokens, the learning procedure can optionally consider the semantic contents both located in the contexts surrounding the target attribute item as well as the content of the target attribute itself. The syntactic tokens are those containing syntactic meaning like HTML tag in the Web documents and punctuation in the free text. Semantic contents are those containing semantic meaning like number and word, or the domain specific semantic class defined in the semantic lexicons like *price* and *product feature.* HISER is able to learn expressive extraction rules that make use of these limited syntactic and semantic information.

An extraction rule consists of three distinct parts: a left pattern component, the target component of the attribute item, and a right pattern component. The left pattern component and right pattern component play the roles of left and right delimiters of the target attribute item, while the target component of the attribute item describes the information to be extracted.

A left pattern component consists of a sequence of token scanning instructions that apply to the prefix of the parent. The token scanning instruction, *ScanUntil()*, in the left pattern component instructs the wrapper to scan and consume the prefix of the parent content until a particular token is matched. Similarly, a right pattern component is a sequence of token scanning instructions that apply to the suffix of the parent. The token scanning instruction in the right pattern component instructs the wrapper to scan and consume the suffix of the parent content until a particular token is matched. The argument of a token scanning instruction can be a token or a wild-card. Both left pattern component and right pattern component allow empty or multiple token scanning instructions. The left pattern component together with the right pattern component identify the boundaries of the attribute item to be extracted.

The target component consists of a sequence of token containing instructions. A token containing instruction, *Contain()*, in the target component instructs the wrapper to test if the text segment contains the semantic content in the token containing instruction. The argument of a token containing instruction can be a token or a wild-card. The target component allows empty or multiple token containing instructions.

In order to satisfy a rule, exact matching of all the token scanning instructions in the left pattern component and right pattern component must be fulfilled. Then the text segment between the boundaries identified by the left and right pattern components can be extracted only if it satisfies the requirements of the target component. If an extraction rule is applied successfully to an instance, the instance is considered to be covered by the rule.

5.2 Extraction Rule Induction Algorithm

Our rule induction algorithm generates extraction rules that identify the start and end of an attribute item within its parent, and describe the semantic contents of the attribute item. Finding the start and end of the attribute item involves the consumption of the prefix and suffix of the parent with respect to the attribute item respectively. Identifying the semantic contents in the attribute item involves the matching of the attribute instances with the domain independent knowledge and domain specific lexicons.

The extraction rule learning algorithm is shown in Figure 5.1. It is a supervised learning algorithm, sets of extraction rules will be induced from hand-tagged training examples. The extraction rule learning is a sequential covering algorithm: as long as there are some uncovered positive examples, it tries to learn a disjunct rule that covers as many positive examples as possible by invoking the functions *Grow_RuleSet* and *Refine_Rule*. The disjunct rule will be inserted into the rule set if the disjunct is perfect. A perfect disjunct is a rule that only covers positive examples. Once all positive examples in the training set have been covered, the rule set containing disjunctive rules is returned. The learning process will be terminated when rule sets have been induced for all the nodes in the hierarchical record structure.

In each iteration, our algorithm tries to randomly select an example that is not covered by the current rule set. Then an initial rule will be generated. After that, we try to generate a set of candidate rules and then select the rule that cover the largest number of uncovered positive training examples for further consideration and refinement. Instead of considering the tokens from the beginning and the end of the parent, candidate rules are generated by considering N tokens before and after the labelled attribute items within its parent, where N is a pre-defined value. The reason is that the nearby tokens surrounding the attribute items are the most crucial to the determine the left and right delimiters. Besides, it can reduce computation time without distorting the rule accuracy by considering less candidate rules.

HISER learns expressive extraction rules that make use of limited syntac-

```
# Function Extraction Rule Learning
1  {RuleSets} = empty.
2  Struct = Hierarchical Record Structure.
3  foreach node n in Struct
4      {Inst_n} = user-labelled training instances for node n.
5      RuleSet_n = rule set of node n in Struct = empty.
6      while there exists an instance p in {Inst_n} not covered by RuleSet_n
7          Grow_RuleSet(RuleSet_n,{Inst_n}).
8          remove all instances in {Inst_n} that are covered by RuleSet_n.
9      add RuleSet_n to {RuleSets}.
10 return {RuleSets}.

# Function Grow_RuleSet(RuleSet_n,{Inst_n})
1  randomly select an instance i from {Inst_n} not covered by RuleSet_n.
2  initial rule r = empty.
3  while (r is not a perfect disjunct)
4      Refine_Rule(r,i).
5  add rule r into RuleSet_n.

# Function Refine_Rule(r,i)
1  generate a set of semantic contents {SC} of instance i.
2  call Topology_Refinement(r,{SC},i) and Token_Refinement(r,{SC}),
   the elements in {SC} will be added to the target component
   during rule refinements, return a set of candidate rules {Cand_Rule}.
3  evaluate the rules in {Cand_Rule}.
4  return the best_rule in {Cand_Rule}.
```

Figure 5.1: The extraction rule induction algorithm.

tic and semantic information. We incorporate both syntactic generalization and semantic generalization so that highly accurate and more expressive rules can be learned. In the process of generating a perfect disjunct rule, we adopt two types of rule refinements: token refinement and topology refinement. Token refinement tries to obtain better disjuncts by generalizing the argument in the token scanning instruction in the left pattern component or right pattern component. It involves generalizing the token syntactically by dropping specific features and semantically by matching with the semantic lexicons.

Based on the background knowledge on HTML, syntactic tokens like "<IMG SRC="./a.html">" and "<IMG SRC="./b.html">" can be converted to a more generalized form, "<IMG>", by dropping the specific image files such as "a.html" and "b.html". Similarly, "<FONT SIZE="+1">" can be generalized to "<FONT>" by dropping the specific font size, while "<IMG>" and other HTML tags, say "<BR>", can be further generalized to "*HTML TAG*", which represents all HTML tags. Syntactic contents like "," and "." can be generalized to *Punctuation*. Domain independent semantic contents like "259.95" and "computer" can be generalized to *Numeric* and *Alphabetic* respectively. Domain specific semantic contents such as "IBM" and "China" can be generalized to *company name* and *country* respectively by matching with the semantic lexicons.

Topology refinement tries to obtain better disjuncts by adding a new token scanning instruction in the left pattern component or right pattern component, and leave the existing token scanning instructions unchanged. Consider the product catalog example in Figure 3.1, assume that the rule for the product price at a particular stage is as follows:

left pattern component:

ScanUntil("$").

right pattern component:

ScanUntil("</FONT>").

Here shows some examples of the right pattern component after topology refinement:

ScanUntil("</FONT>"), *ScanUntil*("</TD>"). or

ScanUntil("</FONT>"), *ScanUntil*("
"). or

ScanUntil("</FONT>"), *ScanUntil*("</TR>"). or

ScanUntil("</FONT>"), *ScanUntil*("</TABLE>").

The consideration of semantic contents in the target component tries to describe the content to be extracted. The content of the training example is matched against the domain independent knowledge and domain specific lexicons. The semantic content will be added to the target component of the rule during rule refinements. Refer to the previous example, if the value of the attribute item "price" is "259.95", by adopting the domain independent knowledge, it indicates that its semantic content is *Numeric*. As a result, *Contain*(*Numeric*) is added to the target component of the rule.

The rules are evaluated by matching against the training instances. The rule that cover the largest number of the uncovered positive training examples will be returned as the best rule for further consideration. In a case where several candidate rules have the same coverage among the uncovered positive training examples, more generalized token is preferred. For example, we prefer the semantic class of a word over literals. The reason is to prefer the least restrictive rule that fits the training data so as to avoid overfitting. After all the positive training examples for a particular node have been covered by the rule set, the rule set is returned. The learning process will be terminated when rule sets have been induced for all the nodes in the hierarchical record structure.

In order to extract the attribute items of interest from semi-structured documents, our wrapper exploits the hierarchical structure and sets of extraction rules. For each node in the structure, it is associated with a rule that extracts that particular node from its antecedent. When we extract the content from an unseen document, the page is first tokenized to a sequence of tokens. The extraction rules for each node in the hierarchical record structure are applied to the sequence of tokens until all tokens have been consumed. Each attribute item is extracted independently among its siblings in the structure. As a final step, we use the hierarchical record structure to group together the individual attribute items to assemble multi-slot records. Our approach can handle missing attribute items, multi-valued attribute items, and different orderings of the attribute items in the records.

Chapter 6

Experimental Results of Wrapper Learning

This chapter presents wrapper learning results of HISER for semi-structured documents in three experimental domains. They are online electronic appliance catalogs, online book catalogs and seminar announcements. Experimental results show that HISER offers encouraging performance for automatic wrapper construction.

6.1 Experimental Methodology

In order to demonstrate the effectiveness of wrapper learning of HISER, we have conducted an experiment to extract attribute items from semi-structured documents including online electronic appliance catalogs, online book catalogs, and seminar announcements. Electronic appliance catalogs and book catalogs are semi-structured Web documents in HTML format, while seminar announcements are semi-structured documents obtained from emails or electronic bulletin board postings.

For each information source, a few training sample records are provided via a graphical user interface. The underlying organization of the textual document is not visible to the users and thus they do not know about the underlying HTML content and tag organization for Web documents. The remaining records in the documents are reserved for evaluation purpose. Based on the user-annotated training examples, a hierarchical record structure and extraction rules are induced for each information source. In order to measure the extraction performance, we manually extract all correct attribute instances from the information sources. These correct answers will be compared with the system extracted answers to evaluate the system performance. If the extracted text segment exactly matches with the true answer associated with the instance, it is considered to be a correct extraction, otherwise it is regarded as an error.

We use two common metrics, namely, *recall* and *precision*, which are widely used in information retrieval tasks, to evaluate the extraction performance. For each attribute item of interest, we compute recall and precision separately. Recall is the number of attribute instances for which the system correctly extracts, divided by the total number of actual attribute instances. Precision is the number of attribute instances for which the system correctly extracts, divided by the total number of attribute instances it extracts.

$$recall = \frac{\text{number of attribute instances correctly extracted}}{\text{total number of actual attribute instances}}$$

$$precision = \frac{\text{number of attribute instances correctly extracted}}{\text{total number of extracted attribute instances}}$$

6.2 Results on Electronic Appliance Catalogs

The information of the online electronic appliance catalog sources is shown in Table 6.1, the second and third column show the name and the Web address of the information sources respectively. The fourth column shows the total number of records collected from that information source[1]. The purpose of every document in this collection is to display the information of electronic appliances by online vendors. The attribute items of interest in this domain are the name, price, and description of the products. The motivation of choosing this domain is that integrating information from Web

[1]The URLs are accurate at the time of writing the thesis.

vendors allows online intelligent comparison-shopping agents to find the best bargain for the users. A sample HTML page in this experimental domain is shown in Figure A.1 in Appendix A.

	Web Site	URL	Total number of Records
S1	1 Cache	http://www.1cache.com	295
S2	Tek Gallery	http://www.tekgallery.com	255
S3	Best Buy Digital	http://www.bestbuydigital.com	274
S4	800.com	http://www.800.com	235
S5	soundcity.com	http://www.soundcity.com	351
S6	Turboprice	http://www.turboprice.com	184
S7	ROXY.com	http://www.roxy1.com:80	287
S8	Zones Portal	http://www.zones.com	319
S9	CDeals	http://www.cdeals.com	175
S10	AbtElectronics.com	http://www.abtelectronics.com	358

Table 6.1: Information on electronic appliance sources

Table 6.2 shows the performance of HISER on the information sources containing electronic appliance catalogs. It shows the recall and precision of each attribute item of interest. Information sources such as 1 Cache, Tek Gallery, Best Buy Digital, Zone Portal, CDeals and AbtElectronics.com contain records with attribute items displayed in a fixed order, but the attribute items of interest may be missing in some records. Although none of these Web sources can be wrapped perfectly, the recall on the target attribute items in most of these information source exceeds 0.85 and the precision is very close to 1.0. It shows that our approach can extract contents accurately from those semi-structured Web information sources with attribute items organized in a linear fashion.

Sites such as 800.com, soundcity.com, Turboprice, and Roxy.com contain records with missing attribute items and multi-valued attribute items. In addition to the selling price, these Web sources display similar information like the original price or discount as well. Still, HISER performs very well on extracting contents from these Web sources with recall always exceeding 0.8 and precision close to 1.0. It shows that our approach can extract contents accurately from those semi-structured Web information sources with attribute items organized in a hierarchical fashion as well.

In semi-structured Web documents, most of the target contents are enclosed with clear separators like HTML tags. As a result, the left pattern component together with the right pattern component in our extraction rules can accurately identify the boundaries of the target contents. Experimental results illustrate that HISER offers very nice extraction performance on a wide range of semi-structured Web documents containing electronic appliance catalogs.

		Product Name		*Description*		*Price*	
	Web Site	recall (%)	prec. (%)	recall (%)	prec. (%)	recall (%)	prec. (%)
S1	1 Cache	98.3	100.0	97.9	99.3	98.2	99.0
S2	Tek Gallery	95.7	99.6	98.8	98.8	98.8	98.8
S3	Best Buy Digital	99.6	100.0	98.9	99.6	100.0	100.0
S4	800.com	79.0	100.0	94.7	98.9	99.0	100.0
S5	soundcity.com	99.4	100.0	98.3	99.4	99.7	100.0
S6	Turboprice	100.0	100.0	76.0	100.0	99.5	100.0
S7	ROXY.com	99.3	100.0	77.4	99.1	99.3	100.0
S8	Zones Portal	100.0	100.0	100.0	100.0	84.3	100.0
S9	CDeals	85.7	98.0	86.3	99.3	86.9	99.4
S10	AbtElectronics.com	87.2	99.7	96.4	100.0	95.0	98.0

Table 6.2: Performance of HISER on the experimental Web sites containing electronic appliance catalogs. (prec. refers to precision)

6.3 Results on Book Catalogs

Table 6.3 shows the information of the online book catalog sources[2]. The purpose of every document in this collection is to display the book information by online bookstores. The attribute items of interest in this domain are the name, author, and price of the books. The motivation of choosing this domain is that integrating information from online bookstores allows online intelligent comparison-shopping agents to find the best bargain for the users.

A sample HTML page in this experimental domain is shown in Figure A.2 in Appendix A. Some of the Web sites in this domain contain records with attribute items organized in a hierarchical fashion. Most of them contain records with missing attribute items, multi-valued attribute items, or different orderings of attribute items. An example of the hierarchical record structure in this domain is shown in Figure 6.1. It shows the hierarchical record structure induced for information source S14.

Table 6.4 shows the performance of HISER on the information sources containing book catalogs. It shows the recall and precision of each attribute item of interest. The performance on the book catalogs domain is similar to the electronic appliance domain. Although most of the sites contain records with missing attribute items, multi-valued attribute items and at-

[2]the URLs are accurate at the time of writing the thesis

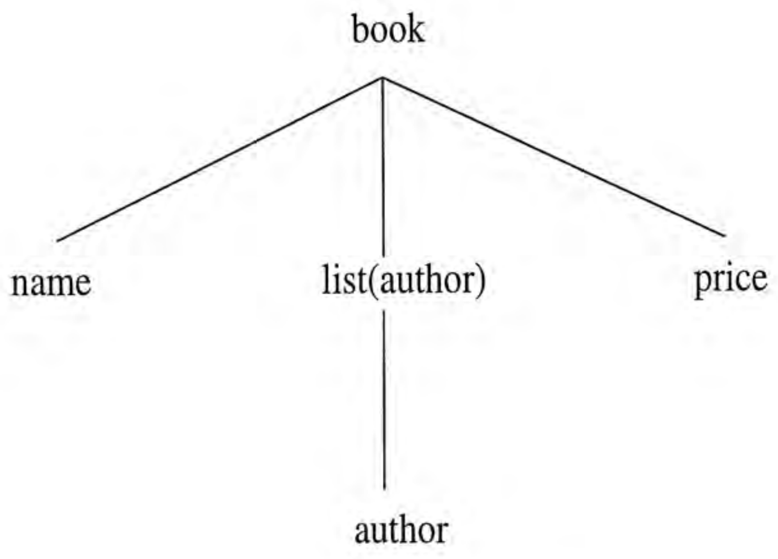


Figure 6.1: Hierarchical record structure for source S14.

	Web Site	URL	Total number of Records
S11	Powell's Book	http://www.powells.com	263
S12	1Bookstreet.com	http://www.1bookstreet.com	189
S13	Borders	http://www.borders.com	200
S14	buy.com	http://www.buy.com	200
S15	BookCloseOuts	http://www.bookcloseouts.com	237
S16	Barnes & Noble.com	http://www.barnesandnoble.com	168
S17	Full Circle Book Store	http://www.fullcirclebooks.com	193
S18	Amazon.com	http://www.amazon.com/books/	250

Table 6.3: Information on book catalog sources.

tribute items displayed in different orderings, the extraction performance is still good. The recall on most of these information source always exceed 0.8. In most of the cases, HISER offers very high precision, which is over 0.9 and is very close to 1.0.

This encouraging results show that HISER is capable of extracting content from a wide range of semi-structured Web documents with attribute items organized in a linear or hierarchical structure. Similar to the electronic appliance catalogs, most of the target contents in book catalogs are enclosed with clear separators like HTML tags. As a result, the left pattern component together with the right pattern component in our extraction rules can accurately identify the boundaries of the target contents.

6.4 Results on Seminar Announcements

Seminar announcements are semi-structured documents that do not contain full grammatical sentences or any mark-up tags. Most of the seminar announcements have missing attribute items, or different orderings of attribute items. The seminar announcement domain contain two collections of university seminar announcements collected at the Chinese University of Hong Kong (CUHK) and the Carnegie Mellon University (CMU). For the first set of this experiment, we collected a total of 103 emails from CUHK containing

		Name		*Author*		*Price*	
	Web Site	recall (%)	prec. (%)	recall (%)	prec. (%)	recall (%)	prec. (%)
S11	Powell's Books	96.2	100.0	96.6	100.0	96.6	100.0
S12	1Bookstreet	98.4	99.5	89.4	89.9	98.9	99.5
S13	Borders	97.0	100.0	93.0	99.5	100.0	100.0
S14	buy.com	94.0	96.9	95.8	97.4	94.0	96.9
S15	BookCloseOuts	97.1	99.6	97.9	97.9	100.0	100.0
S16	Barnes & Nobles.com	85.7	100.0	83.9	100.0	84.5	100.0
S17	Full Circle Book Store	99.5	100.0	100.0	100.0	100.0	100.0
S18	Amazon.com	96.0	100.0	98.7	98.7	100.0	100.0

Table 6.4: Performance of HISER on the experimental Web sites containing book catalogs. (prec. refers to precision)

seminar announcements. Sample seminar announcements from CUHK are shown in Figure A.3 and Figure A.4 in Appendix A. In the second set of this experiment, we randomly select 200 pieces of electronic bulletin board postings containing seminar announcements from CMU which is available at the RISE repository [34]. Sample seminar announcements from CMU are shown in Figure A.5, Figure A.6 and Figure A.7 in Appendix A. The purpose of every document in these two collections is to announce an upcoming seminar or meeting. The attribute items of interest are the topic, speaker, venue, date, start time and end time of the seminar. The motivation of choosing this domain is that extracting information from electronic board postings and email allows intelligent agents to summarize upcoming seminars and alert interested parties.

In this experiment, we label training examples in 6 seminar announcements from CUHK and 10 seminar announcements from CMU, the remaining documents in the two collections are reserved for evaluation purposes. Table 6.5 shows the performance of HISER on the domain of seminar announcements. The second and third column show the extraction performance of HISER on the seminar announcements from CUHK, while the fourth and fifth column show the extraction performance of HISER on the seminar announcements from CMU.

Although there are no clear HTML delimiters in seminar announcements,

	CUHK		CMU	
	recall (%)	precision (%)	recall (%)	precision (%)
Topic	82.5	87.0	65.5	77.1
Speaker	86.6	91.3	74.4	72.6
Venue	86.3	89.1	56.5	92.9
Date	83.5	91.0	97.0	98.9
Start time	86.3	89.1	85.9	93.4
End time	89.9	100.0	86.1	98.8

Table 6.5: Performance of HISER on the seminar announcements from CUHK and CMU.

experimental results show that our extraction rules are expressive enough to handle these semi-structured documents containing seminar announcements from CUHK. The recall and precision for all the attribute items of interest always exceed 0.82 and 0.87 respectively. All of the seminar announcements from CUHK have a linear record structure, which means all attribute items are single-valued. But, most of them have missing attribute items, or different orderings of attribute items. In addition, all of the target attribute items, the topic, speaker, venue, date, start time, and end time, appear in the heading of the documents. Refer to the sample seminar announcements from CUHK shown in Figure A.3 and Figure A.4, all of the target contents appear before the abstract of the seminar. This important feature makes the extraction tasks less difficult. Instead of considering both semi-structured contexts as in the heading and free text as in the abstract and biography, HISER can learn accurate extraction rules which capture the common delimiters of the target contents in the semi-structured contexts only.

For semi-structured Web documents such as HTML pages, most of the content are enclosed with mark-up tags. As a result, the left pattern component together with the right pattern component are expressive enough to locate the clear HTML delimiters and determine the content to be extracted. For other semi-structured documents such as seminar announcements, there are no clear delimiters to separate the contents to be extracted. To extract content accurately from this kind of documents, in additional to the left and right pattern components, the target component plays a significant role to restrict the type of content to be extracted.

For the seminar announcements from CMU, the extraction performance on the date, start time and end time is very good. The recall and precision for these attribute items exceed 0.85 and 0.93 respectively. The performance on the topic, speaker and venue is fair when comparing with other attribute items and the results of CUHK domain. Unlike the seminar announcements from CUHK, some of the target attribute items appear in the semi-structured contexts of the documents, some of them appear as free text in the collections of CMU seminar. The date, start time and end time of the seminar always present in the semi-structured heading of the document, while other target attribute items may appear as free text in full, grammatical sentences. Refer to the sample seminar announcements from CMU shown in Figure A.5 and Figure A.6, all of the target contents appeared in semi-structured format. In Figure A.7, the speaker and venue appeared as free text with no clear separator.

Since HISER does not make use of any linguistic processing, extracting the attribute items speaker, topic and venue are more difficult as some of them appear as free text in the document collection. That is why the extraction performance on these three attribute items is lower. Still, HISER is able to extract content accurately from the semi-structured contexts. The extraction performance on the date, start time and end time is very good. In addition to the left and right pattern components, the target component of the extraction rule plays a significant role to extract contents accurately from semi-structured non-Web documents. Since some of the attribute items appear in the semi-structured context and some of them appear in free text context, the left pattern component together with the right pattern component are not expressive enough to capture the surrounding patterns of the target attribute items. The target component is useful to describe the type of the information to be extracted, which can improve the extraction accuracy.

Experimental results illustrate that HISER offers very nice extraction performance on a range of semi-structured documents including online Web documents with HTML tags and non-Web documents without clear mark-up tags. In addition, HISER is able to handle missing attribute items, multi-valued attribute items, and different orderings of attribute items.

Chapter 7

Adapting Wrappers to Unseen Information Sources

This chapter discusses the third module of our HISER system, which is used for wrapper adaption. HISER is able to adapt a learned wrapper of a particular information source to unseen information sources in the same domain. Our approach is to label training examples automatically by employing a machine learning technique.

7.1 Motivation

Semi-structured information sources such as Internet sites often change their layout and formatting regularities. Thus, maintaining wrapper and adapting wrappers across information sources is difficult. Users are required to provide training examples for each information source when the first time the wrapper is constructed. Likewise, manual training examples are needed for an information source when the formatting regularities are altered. Very few existing works address the problem of wrapper adaptation. To cope with these problems and further reduce the user effort involved, HISER tries to adapt a learned wrapper to unseen information sources by using a machine learning approach. Novel feature selection and candidate seeking techniques are developed to automate the process of manual annotation of training examples. This approach can also solve the wrapper maintenance problem when the layout and formatting regularities of an information source changes.

It appears that the wrapper learning method in HISER can be used for the task of automatic labelling of training examples. However, directly applying this learning method to unseen information sources may not yield satisfactory performance. While the layout and formatting regularities of semi-structured Web documents differ among different information sources and the extraction rules of a particular information source usually are not applicable to unseen information sources, the attribute items of interest often share some common features and characteristics across different information sources. We observe that the semantic content of the attribute instances of interest remains unchanged. For example, the price of a product is typically in *Numeric* type, the description of a product usually contains terms related to *product feature*. Besides, the characteristics of the content of the attribute instances are often similar across different information sources. For example, the length, word count and proportion of punctuation symbols are roughly similar.

In addition to the content, the surrounding tokens of the attribute instances are sometimes similar across different information sources. For example, in the domain of electronic appliance catalogs, the product name of an electronic appliance is usually a hyperlink for that particular product. Thus it is usually surrounded by HTML tags "<A HREF=...>" and "</A>". Beside, the product name is usually displayed in special formats, say in bold type or special font type. Thus it is usually surrounded by HTML tags like "<FONT ...>", "</FONT>", "<B>" and "</B>". These regularities enable HISER to tackle the problem of wrapper maintenance and adaptation because those previously-labelled training examples can be an approximation to the training examples that a user may provide for new unseen information sources.

Our HISER approach tackles the wrapper adaptation problem by first automatic labelling of training examples. In addition to the left pattern component and right pattern component, our extraction rules contain the target component, which captures the semantic contents of the target attribute item. To trigger an extraction rule, exact matching of the token scanning instructions and token containing instructions are required. While the extraction rules of an information source usually can not be applicable to unseen information sources with dissimilar layout or format, the target component of the extraction rules can be treated as "sample-seeking" rules to seek potential training samples from unseen information sources automatically. By making use of Support Vector Machines (SVM) [16, 63], a machine learning technique, to capture the common characteristics of the attribute instances of interest, the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources can be predicted. Furthermore, the automatic annotated examples can be examined by human to refine the annotation quality before they are used for subsequent processing. Once the training examples are annotated, a suitable wrapper can be induced for the unseen information sources.

7.2 Support Vector Machines

Systems for document retrieval, routing, categorization and other information retrieval tasks rely heavily on automated classifiers [43]. Support Vector Machines (SVM) [16, 63], a kind of machine learning technique, is a relatively new learning approach for solving two-class pattern recognition problems.

They have strong theoretical foundations and proven empirical successes.

The SVM method is defined over a vector space where the problem is to find a decision surface that best separates the data points into two classes. For simplicity, Figure 7.1 shows an example in a two-dimensional space with linear separable data. The solid line shows a possible decision surface that correctly separates the two groups of data. The dashed lines parallel to the solid line show how much the decision surface can move without causing misclassification of the data. The distance between the two dashed lines is the margin. The SVM problem is to find the decision surface that maximizes the margin between the data points in a training set. The idea can also be applied to high dimensional space and to data points that are not linearly separable. A decision surface in a linearly separable space is a hyperplane.

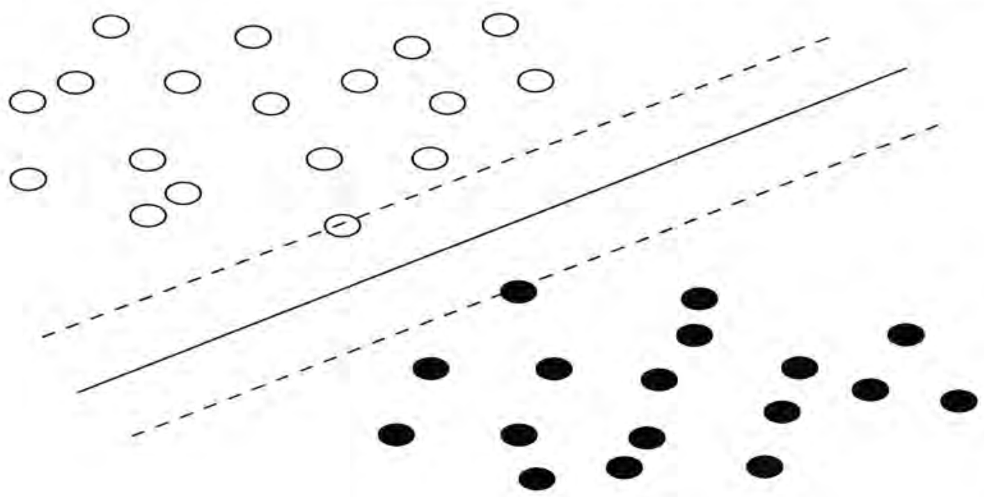


Figure 7.1: The decision line (solid) with a margin which is the distance between the two parallel dashed lines

Let $y_i \in \{\pm 1\}$ be the classification for $\vec{x_i}$, and there are n user-labelled training examples: $(\vec{x_1}, y_1), \cdots, (\vec{x_n}, y_n)$, $\vec{x_i} \in R^d$ where d is the dimensionality of the vector. The problem is to find an optimal weight vector $\vec{w^*}$ and a constant b from the training set of data such that $||\vec{w^*}||$ is minimum, and the following constraints are satisfied:

$$\vec{w^*} \cdot \vec{x_i} - b \geq +1 \qquad \text{if } y_i = +1$$

$$\vec{w^*} \cdot \vec{x_i} - b \leq -1 \qquad \text{if } y_i = -1$$

The decision surface for linearly separable space is a hyperplane which can be written as:

$$\vec{w^*} \cdot \vec{x_i} - b = 0$$

Training examples that satisfy the equality are termed support vectors. The support vectors define two hyperplanes, one that goes through the support vectors of one class and one goes through the support vectors of the other class. The distance between the two hyperplanes defines a margin and this margin is maximized when the norm of the weight vector $||\vec{w^*}||$ is minimized.

SVM has been proven to perform well on a wide range of applications, including object recognition and text classification [19, 36, 70]. The advantage of SVM is that the execution speed is very fast. Another advantage is that SVM is remarkably intolerant of the relative sizes of the number of training examples of the two classes. In most learning algorithms, if there are more examples of one class than another, the algorithm will tend to correctly classify the class with the larger number of examples, thereby minimizing the error rate. Since SVM tries to separate the patterns in high dimensional space, but not directly minimize the error rate, it is relatively insensitive to the number of training examples of each class.

SVM^{light} is an implementation of Vapnik's Support Vector Machines [63]. The source code is free for scientific use and available at [35]. There are two executable programs, "svm_learn" and "svm_classify". "svm_learn" is the learning module used to learn a SVM model for a set of training examples, "svm_classify" is the classification module used for classifying potential candidates.

SVM can be applied to HISER for addressing the wrapper adaptation problem. The idea is to annotate training examples automatically from unseen information sources. In our framework, for each attribute item of interest, there are n user-labelled training examples: $(\vec{x_1}, y_1), \cdots, (\vec{x_n}, y_n)$, $\vec{x_i} \in R^d$ where d is the dimensionality of the vector representing the number of features considered, $y_i \in \{\pm 1\}$ is the classification for the candidate $\vec{x_i}$, +1 represents positive example and -1 represents negative example. The problem is to learn an optimal weight vector $\vec{w^*}$ and a constant b from the training set of data such that a decision surface is found to best separate the data points into two classes. For each attribute item of interest, a SVM model is built to predict the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources.

7.3 Feature Selection

SVM is parameterized by a set of features, denoted by the feature vector $\vec{x_i}$. In text classification, a feature can be a term. A term can be a word or a phrase. The feature value can be defined as the term frequency (TF) or the product of term frequency and the inverse document frequency (TF-IDF) [57]. Term frequency is the number of times the term appears in the document. Document frequency (DF) is the number of times the term occurs in a document collection. Such selection is based on the fact that terms can be good representation of the document category.

In our approach, in order to capture the characteristics of the content of the attribute instances as well as the contexts surrounding them, three sets of features are defined. The first set is related to the content information. We observe that the semantic content and the content characteristics of the attribute instances are often quite similar across different information sources. The following nine domain independent features are considered:

1. **length**: number of characters in the content

2. **token count**: number of tokens in the content

3. **mean token length**: average number of characters of each token in the content

4. **digit density**: proportion of digits in the content

5. **letter density**: proportion of alphabetic characters in the content

6. **uppercase density**: proportion of uppercase characters in the content

7. **lowercase density**: proportion of lowercase characters in the content

8. **punctuation density**: proportion of punctuation symbols in the content

9. **HTML density**: proportion of HTML tags in the content

For example, consider the book name "C: How to Program". Each feature will take on a value as follows; **length** = 17, **token count** = 8, **mean token length** = 2.125, **digit density** = 0.0, **letter density** = 0.765, **uppercase density** = 0.176, **lowercase density** = 0.588, **punctuation density** = 0.059, and **HTML density** = 0.0.

In addition to the content, the surrounding contexts of the attribute instances are sometimes similar across different information sources. The second and third set of domain independent features are related to the left and right surrounding contexts respectively. The features represent different categories of tokens as follows.

1. **HTML_table**: HTML tags related to tables

   (e.g.: <TABLE>, <TD>, </TR>)

2. **HTML_link**: HTML tags related to hyperlink

   (e.g.: <A HREF=...>, </A>)

3. **HTML_img**: HTML tags related to displaying image

   (e.g.: <IMG SRC=...>, </IMG>)

4. **HTML_list**: HTML tags related to displaying list

   (e.g.: <OL>, <LI>, </UL>)

5. **HTML_font**: HTML tags related to font type or size

   (e.g. <FONT SIZE=...>, <B>, </FONT>)

6. **HTML_layout**: HTML tags related to layout such as paragraphing or newline (e.g.: <P>, <BR>)

7. **WORD**: tokens that contain alphabetic characters only

   (e.g.: NEW, sell, Price)

8. **NUMBER**: tokens that are numeric strings

(e.g.: 2000, 199.95)

9. **PUNCTUATION**: tokens that are punctuation symbols

   (e.g.: !, ;)

The feature value is defined as the number of matched tokens in the surrounding contexts for the category divided by the total number of tokens for the category in the document. It represents the proportion of the matched tokens for a category in the document.

$$feature_j = \frac{\text{number of matched tokens for category } j \text{ in the sorrounded contexts}}{\text{total number of tokens for category } j \text{ in the document}}$$

where $feature_j$ is the value of the j^{th} feature for a particular candidate.

For example, if **HTML_table** is a category for all HTML tags related to table such as <TABLE>, </TABLE>, <TH>, </TH>, <TR>, </TR>, <TD> and </TD>. In the left surrounding context of a particular attribute instance, 2 tokens matched with the elements in this category and there are totally 50 tokens in the documents matched this category, the feature value for **HTML_table** will be 0.04.

Although all features listed above are domain independent, domain specific semantic classes such as *price* and *product feature* can be easily adopted to different domains without modifying the algorithm.

We have three sets of features in total. The first set of feature is related to the content information. The second and third set of features are related to the left and right surrounding contexts respectively. The value of these three sets of features may fall in different orders of magnitude. Refering to the previous example, the value of the feature **length** may be in the order of 10^1 and the value of the feature **punctuation density** may be in the order of 10^{-2}. It is not effective if different features have values in very much different orders of magnitude. As a result, we normalize all features to values between 0 and 1, by using the cosine normalization [56]:

$$feature'_{ij} = \frac{feature_{ij}}{\sqrt{\sum_i feature_{ij}^2}}$$

where $feature_{ij}$ is the value of the j^{th} feature for the i^{th} training example and $feature'_{ij}$ is the normalized feature value. The normalized feature values will be used to learn SVM models and classify potential candidates.

7.4 Automatic Annotation of Training Examples

Basically, our method for automatic annotation of training examples from unseen information sources can be treated as a kind of classification. Once the potential candidates for the training example of a target attribute item are identified in unseen information sources, they are further considered and their degree of confidence of being "good" positive training examples can be predicted. We employ a classifier generated by SVM to capture the common characteristics of the attribute instances of interest. Those potential candidates can then be classified as training examples or not.

7.4.1 Building SVM Models

We observe that the semantic content and the content characteristics of the attribute instances of interest remain largely unchanged no matter how they are displayed in different information sources. In addition to the content, the surrounding contexts of the attribute instances are often similar across different information sources. These regularities enable HISER to tackle the problem of wrapper adaptation because those previously-labelled training examples can be an approximation to the training examples that a user may provide for new unseen information sources.

We employ inductive SVM to capture the common characteristics of the attribute instances of interest. For a particular information source, users are required to provide positive training examples as well as negative training examples for each target attribute item. Based on the three sets of features described in Section 7.3, the feature values of the training examples are identified. After that, the feature values are normalized to the range between 0 and 1 by using cosine normalization. Eventually, a SVM model is built for each attribute item of interest. These SVM models are used to estimate the degree of confidence of the potential candidates being "good" positive training examples.

7.4.2 Seeking Potential Training Example Candidates

Each textual document contains a sequence of fields or tokens. A field or token can be a word, number, punctuation, date, HTML tag, specific ASCII character or domain specific contents like *price* and *product feature.* We define an *object* to be a list of continuous tokens in a Web page. The sequence of tokens that constitutes the object may contain any kind of the tokens described above, but the first and last elements cannot be HTML tags. We define a *segment* to be a sequence of continuous tokens in a Web page, but it does not contain any HTML tags.

The first and last tokens of each training example cannot be HTML tags, so each training example can be represented as an object. We assume that the attribute items of interest are the same across different sources of the same domain. For each attribute item of interest, our method will first try to locate potential candidates for the training examples from unseen information sources using the target component of the learned extraction rules. The initial candidate is the segment containing the contents of the target component. Next, we try to produce a set of potential object candidates by extending this segment forward and backward. A parameter T is used to control the degree of extension.

We define *Pre-Candidate* and *After-Candidate* to identify the position of the beginning and the ending of a potential object candidate. For an object candidate c, *Pre-Candidate* is defined as follows:

> *Pre-Candidate* is the position of the first element of
>
> the segment immediately before the segment of the candidate c.

A set of $\{$*Pre-Candidate*$\}$ is generated by extending the initial candidate c_0 backward for a degree of extension T.

Similarly, for an object candidate c, *After-Candidate* is defined as follows:

> *After-Candidate* is the position of the last element of
>
> the segment immediately after the segment of the candidate c.

A set of $\{$*After-Candidate*$\}$ is generated by extending the initial candidate c_0 forward for a degree of extension T.

After the positions are identified, all the combinations of $\{$*Pre-Candidate*$\}$ and $\{$*After-Candidate*$\}$ constitute the potential object candidates for the attribute item of interest. For example, if $\{$*Pre-Candidate*$\}$ $= \{s_1, s_2, s_3\}$ and $\{$*After-Candidate*$\}$ $= \{e_1, e_2, e_3\}$, a set of potential object candidates is generated as follows:

$$\{(s_1, e_1), (s_1, e_2), (s_1, e_3), (s_2, e_1), (s_2, e_2), (s_2, e_3), (s_3, e_1), (s_3, e_2), (s_3, e_3)\}.$$

where (s_i, e_j) denotes an object candidate with starting position at s_i and ending position at e_j.

7.4.3 Classifying Potential Training Examples

Once the potential object candidates for each attribute item of interest are identified in unseen information sources, the three sets of features defined in Section 7.3 will be used to determine the content characteristics of the potential candidates as well as the contexts surrounding them.

For each attribute item of interest, the feature values of the potential candidates are determined and then normalized by cosine normalization. Then we apply the previously learned classifiers to predict the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources. After classification, those candidates with high degree of confidence are treated as positive training examples for unseen information sources.

Once the training examples are annotated, a hierarchical record structure and the extraction rules can be induced by using the wrapper learning component in HISER. As a result, HISER can extract information from unseen information sources in the same domain by requiring manual training examples from one single information source only. This technique can also be adopted to wrapper maintenance problem when a wrapper is no longer suitable for a particular information source due to modifications in layout presentation.

Chapter 8

Experimental Results of Wrapper Adaptation

This chapter presents wrapper adaptation results of HISER for semi-structured Web documents in two experimental domains. They are online electronic appliance catalogs and book catalogs. Experimental results show that HISER is capable of adapting a learned wrapper of one information source to unseen information sources.

8.1 Experimental Methodology

In order to demonstrate the capability of HISER to adapt a learned wrapper to unseen information sources by automatic annotation of training examples, we have conducted experiments using the same Web information sources as in the previous experiments. There are two different domains, namely, online electronic appliance catalogs and book catalogs.

We wish to evaluate the information extraction performance on unseen information sources when we adapt a previously learned wrapper to them. For each experimental domain, we have conducted two sets of experiments. The first set of experiment is to simply apply the learned wrapper of one particular source without adaptation to all other sources for extraction. The results of this experiment can be treated as a baseline. The second set of experiment is to adapt the learned wrapper of one particular source to other unseen information sources by using the wrapper adaptation technique. Once training examples are annotated automatically, unseen information sources can learn their own wrapper for extraction. In all experiments, users are only required to provide a few training examples for one particular information source only.

Table B.1 and Table B.3 in Appendix B show the detailed results of the first set of experiment for the domain of online electronic appliance catalogs and online book catalogs respectively. The i^{th} row in these two tables represents an experiment of extracting contents from all information sources by using the learned wrapper of the information source i. Each cell in these tables is divided into two sub-columns and three sub-rows. The three sub-rows represent the extraction performance on the name, description, and price of the products respectively for the domain of electronic appliance catalogs. The three sub-rows represent the extraction performance on the title, author, and price of the books respectively for the domain of book catalogs. The two sub-columns represent the recall and precision on the attribute items of interest respectively. The diagonal cells of Table B.1 and Table B.3 show the capability of HISER to extract information from semi-structured Web documents originating from the same source. This result is, in fact, covered in Chapter 6. It shows that HISER presents a very nice extraction capability on semi-structured Web documents when training examples are available. Other cells in these tables represent the results of applying the learned wrapper of a particular information source to other unseen information sources for extraction. The results of this experiment can be treated as a baseline when comparing with the results of adapting a wrapper to unseen information sources.

Table B.2 and Table B.4 in Appendix B show the detailed results of the second set of experiment for the domain of online electronic appliance catalogs and online book catalogs respectively. They show the experimental results of adapting a learned wrapper of an information source to the remaining unseen information sources by automatic annotation of training examples. The i^{th} row in these tables represents an experiment of adapting the learned wrapper of information source i to all other information sources. Once again, each cell in these tables is divided into two sub-columns and three sub-rows. The three sub-rows represent the extraction performance after adaptation on the name, description, and price of the products respectively for the domain of electronic appliance catalogs. The three sub-rows represent the extraction performance after adaptation on the title, author, and price of the books respectively for the domain of book catalogs. The two sub-columns represent the recall and precision on the attribute items of interest respectively.

8.2 Results on Electronic Appliance Catalogs

Table 8.1 summarizes the experimental results for the domain of electronic appliance catalogs. It shows the average extraction performance on the product name, description, and price respectively for the cases of without adaptation and with adaptation when training examples of one particular information source are provided. The detailed experimental results are shown in Table B.1 and Table B.2 in Appendix B.

In Table 8.1, the first column shows the information sources where training examples are provided manually. The columns labelled with "Without Adaptation" show the average recall and precision on the attribute items of interest for the first set of experiment. The learned wrapper of the information source specified in column one are applied for extraction from all the remaining information sources. The columns labelled with "With Adaptation" show the average recall and precision on the attribute items of interest for the second of experiment. The learned wrapper of the information source specified in column one is adapted to all remaining information sources. The last row shows the overall average recall and precision on the attribute items for all information sources in the cases of without adaptation and with adaptation.

For the first set of experiment, since the layout and formatting regularities of the first three information sources are exactly the same, the wrapper of any one of these three information sources can be applied to the other two without adaptation. But in general, it is not true for most of the information sources. The wrapper of a particular information source cannot be applied to other information sources for extraction. Thus, in most of the cases of without adaptation, the average extraction performance is 0.

In the case of adapting a learned wrapper of a particular information source to other information sources, the extraction performance on the two important pieces of information, the product name and price are good. The extraction performance on the product description is fair. The overall average recall and precision for the product name are 0.63 and 0.79 respectively. The

	product name				*description*				*price*			
	Without Adaptation		**With Adaptation**		**Without Adaptation**		**With Adaptation**		**Without Adaptation**		**With Adaptation**	
Source	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.
S1	21.7	22.2	70.4	83.8	22.0	22.0	34.0	43.7	22.1	22.1	89.2	93.8
S2	22.0	22.2	72.6	86.2	21.9	22.1	34.1	46.3	22.0	22.1	90.2	94.9
S3	21.6	22.2	72.1	84.8	21.9	22.0	34.4	45.6	21.9	22.0	89.8	95.0
S4	0.0	0.0	63.5	79.2	0.0	0.0	13.8	24.4	0.0	0.0	81.6	96.5
S5	0.0	0.0	61.4	77.8	0.0	0.0	10.0	14.4	0.0	0.0	87.8	94.5
S6	0.0	0.0	62.6	80.9	0.0	0.0	19.4	27.3	0.0	0.0	78.1	95.6
S7	0.0	0.0	73.6	90.0	0.0	0.0	10.8	13.8	0.0	0.0	86.1	96.0
S8	0.0	0.0	59.7	69.8	0.0	0.0	5.1	10.2	0.0	0.0	81.1	90.0
S9	0.0	0.0	44.8	64.8	0.0	0.0	11.1	18.2	0.0	0.0	78.0	96.9
S10	0.0	0.0	52.7	67.6	0.0	0.0	22.9	34.6	0.0	0.0	88.7	93.2
average	6.5	6.6	63.3	78.5	6.6	6.6	19.6	27.9	6.6	6.6	85.1	94.6

Table 8.1: Average extraction performance on *product name*, *description*, and *price* in the electronic appliance catalog domain for the cases of without adaptation and with adaptation. (ave. refers to average and prec. refers to precision in terms of percentage)

overall average recall and precision for the price are 0.85 and 0.95 respectively. The overall average recall and precision for the description are 0.20 and 0.28 respectively. It means that the training examples for the name and price of the products can be annotated accurately in most cases.

The reason is that the semantic content of the product name and price share some regularities across different information sources. The price of a product is typically in *Numeric* type, while the name of a product usually contains a sequence of words and cardinal numbers. In addition, the surrounding contexts of the product name and price also bear some regularities across different information sources. The product name is usually a hyperlink for that particular product and thus it is usually surrounded by HTML tags like "<A HREF=...>" and "</A>". Besides, the product name is usually displayed in special formats or font size and thus it is usually surrounded by HTML tags like "<FONT SIZE=...>" and "</FONT>". Similarly, the left surrounding contexts of the price usually contains some tokens related to price such as "$", "Price" and "buy". Besides, the price is usually displayed in special formats or font size and thus it is usually surrounded by HTML tags like "<FONT SIZE=...>", "</FONT>" and "<B>".

For the description of the products, the extraction performance is fair. It means that the training examples for the product description cannot be annotated accurately. The reason is that the information sources display the product description in many different formats. Besides, the content of the product description contains some HTML tags in many cases. So it is rather difficult to locate the true boundaries for the training examples of the product description. Although extracting the product description fails in many cases, the extracted description, in fact, contains part of the expected description or covers more than the expected description. Sometimes, the extracted content for the product description contains the product name as well. As a result, they still contains some useful information or clues on the products.

8.3 Results on Book Catalogs

Table 8.2 summarizes the experimental results for the domain of book catalogs. It shows the average extraction performance on the book title, author, and price respectively for the cases of without adaptation and with adaptation when training examples of one particular information source are provided. The detailed experimental results are shown in Table B.3 and Table B.4 in Appendix B.

In Table 8.2, the first column shows the information sources where training examples are provided manually. The columns labelled with "Without Adaptation" show the average recall and precision on the attribute items of interest for the first set of experiment. The columns labelled with "With Adaptation" show the average recall and precision on the attribute items of interest for the second set of experiment. The last row shows the overall average recall and precision on the attribute items for all information sources in the cases of without adaptation and with adaptation.

For the first set of experiment, since different information sources use their own layout and formatting regularities, the wrapper of a particular information source cannot be applied to other sources for extraction. As a result, in most of the cases of without adaptation, the average extraction performance is 0.

In the case of adapting a learned wrapper of a particular information source to other information sources, the extraction performance on the two important pieces of information, the book title and price are good. The extraction performance on the author is fair. The overall average recall and precision for the book title are 0.54 and 0.71 respectively. The overall average recall and precision for the price are 0.84 and 0.94 respectively. The overall average recall and precision for the author are 0.17 and 0.25 respectively. It means that the training examples for the title and price of the books can be annotated accurately in most cases.

Similar to electronic appliance catalogs, the title and price of the books can be annotated accurately. The reason is that the surrounding contexts of

	book title				*author*				*price*			
	Without Adaptation		**With Adaptation**		**Without Adaptation**		**With Adaptation**		**Without Adaptation**		**With Adaptation**	
Source	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.	ave. recall	ave. prec.
S1	0.0	0.0	59.9	70.1	0.0	0.0	21.2	30.7	0.0	0.0	88.0	95.9
S2	0.0	0.0	60.5	69.3	0.0	0.0	19.1	32.5	0.0	0.0	83.1	94.0
S3	0.0	0.0	55.9	69.1	0.0	0.0	17.5	30.4	0.0	0.0	87.1	95.7
S4	0.0	0.0	47.5	65.6	0.0	0.0	10.4	14.0	0.0	0.0	83.2	94.8
S5	0.0	0.0	56.2	73.8	0.0	0.0	21.3	31.8	0.0	0.0	89.7	93.9
S6	0.0	0.0	63.2	83.1	0.0	0.0	23.0	29.7	0.0	0.0	86.5	95.6
S7	0.0	0.0	34.8	57.6	0.0	0.0	0.0	0.0	0.0	0.0	77.0	92.8
S8	0.0	0.0	55.3	75.0	0.0	0.0	22.0	33.1	0.0	0.0	79.8	92.6
average	0.0	0.0	54.2	70.5	0.0	0.0	16.8	25.3	0.0	0.0	84.3	94.4

Table 8.2: Average extraction performance on *book title*, *author*, and *price* in the book catalog domain for the cases of without adaptation and with adaptation (ave. refers to average and prec. refers to precision in terms of percentage)

the title and price bear some regularities across different information sources. For example, the book title is usually a hyperlink to that particular book and is is usually displayed in special formats or font size. The price of a book is typically in *numeric* type. Besides, the price is usually displayed in special formats or font size, and its left surrounding contexts usually contains some tokens related to price such as "$", "Price" and "buy",

In many cases, the performance on the author is fair. The major reason is that there are no clear separators for displaying book author. There are many various ways to display this attribute item and so it is difficult to capture the common surrounding contexts for this attribute item across different information sources. In some information sources, book authors are enclosed with HTML tags. But in other sources, they are displayed in free text format together with other contents as well. Since our technique makes use of the HTML tags to identify potential candidates for the training examples of the target attribute items, the performance on this piece of information is rather limited.

To conclude, HISER is capable of adapting a learned wrapper of one information source to unseen information sources by annotating training examples automatically. Experimental results show that our method offers encouraging performance for automatic wrapper adaptation.

Chapter 9

Conclusions and Future Work

9.1 Conclusions

We have developed an information extraction system, known as HISER, to handle semi-structured documents. Based on a few user-labelled training examples, HISER can automatically learn a hierarchical record structure and extraction rules tailored to an information source. To further reduce the user effort of providing training examples, we have proposed a technique to adapt a learned wrapper to unseen information sources by annotating training examples automatically using SVM.

For automatic wrapper construction, HISER employs a two-stage learning task, namely, hierarchical record structure learning and extraction rule induction. In hierarchical record structure learning, we try to automatically generate a representation of the relationship among the attribute items for the records appeared in an information source. Extraction rules are then induced for each node in the hierarchical record structure to handle the extraction task. Instead of constructing a single extraction rule that takes into accounts all possible attribute item orderings and arbitrary levels of embedded data, novel extraction rules are constructed to deal with the tasks of extracting each attribute item from its parent in the record structure. Our design can handle missing attribute items, multi-valued attribute items, and attribute items appeared in different orders. In the extraction rule induction, HISER learns expressive extraction rules that incorporate both syntactic generalization and semantic generalization. Experimental results show that our HISER system is able to accurately extract information from semi-structured documents with high degree of structure.

New standards such as XML will simplify the extraction of structured information from heterogenous information sources. However, XML requires Web information sources to accept such standards. Still, large amount of data available electronically does not follow such standards. So, XML will make wrapper construction simpler but will not eliminate the need of wrappers.

The performance of HISER on extracting more loosely structured information is fair. We assume that most of the records in an information source are formatted with similar structure. If the format of the records in an information source varies a lot, the performance will decline. Another example is to extracting information from free texts. To extract attribute items from free texts, sophisticated linquistic processing techniques are needed. Techniques such as part-of-speech tagging, parsing, lexical semantic tagging and syntactic analysis can be used to capture the contextual patterns appearing just before and just after the target attribute items in free texts.

We also investigate the problem of adapting a learned wrapper to unseen information sources. To achieve this goal, we develop novel feature selection and candidate seeking techniques to automate the process of annotating training instances for unseen information sources. Support Vector Machines (SVM), a machine learning technique, is employed to capture the characteristics of the surrounding contexts and the content of the previously-labelled training examples. Eventually, the degree of confidence of the potential candidates being "good" positive training examples for unseen information sources can be predicted. Experimental results show that this technique can accurately annotate the training examples of the attribute items of interest for unseen information sources.

9.2 Future Work

The performance of our HISER approach for automatic wrapper construction and wrapper adaptation for semi-structured documents has been shown to be useful. Further research can be done to improve the system. Some directions for future work are summarized as follows.

One possibility is to optionally allow HISER to make use of sophisticated linguistic processing to handle free texts. These techniques include part-of-speech tagging, parsing, lexical semantic tagging and syntactic analysis. This feature may enhance HISER to handle textual documents ranging from highly structured format to free text format.

Another possibility is to improve the quality of the automatic annotated training examples from unseen information sources via relevance feedback. Users can judge if the automatic annotated training examples are relevant via a graphical user interface. Hopefully, more accurate extraction rules can be induced to improve the extraction performance.

Appendix A

Sample Experimental Pages

This chapter shows some sample experimental pages. Figure A.1 and Figure A.2 show the excerpt of sample HTML pages from the domain of electronic appliance catalogs and book catalogs respectively. Figure A.3 and Figure A.4 show sample seminar announcements from CUHK. Figure A.5, Figure A.6 and Figure A.7 show sample seminar announcements from CMU.

```
......

<TR>
<TD ALIGN=right VALIGN=top><IMG SRC=/Img/trans_1x1.gif BORDER=0 WIDTH=1 HEIGHT=5><BR></TD>
<TD COLSPAN=2><FONT SIZE=3><A HREF="http://store.yahoo.com/1cache/brotmffaxcol1.html">
<b>Brother MFC-7160 Fax, Color Scanner, Color PC, Printer, Copier...</b></A></TD>



<TD ALIGN=right><FONT SIZE=2 ><B>&nbsp;&nbsp;$699.95</TD></TR>
<TR><TD></TD><TD><A HREF="http://store.yahoo.com/1cache/brotmffaxcol1.html">
<IMG SRC="http://store1.yimg.com/I/1cache_1550_77361246" WIDTH=70 HEIGHT=61 BORDER=0
HSPACE=0 VSPACE=0></A></TD>
<TD><FONT SIZE=2>Brother MFC-7160 Fax, Color <font color=red>Scanner</font>, Color PC,
Printer, Copier, Video Capture Color Printer Up to 1440  ... </FONT></TD><TD>
</TD></TR>

<TR><TD></TD><TD></TD><TD></TD><TD></TD></TR>
<TR>
<TD ALIGN=right VALIGN=top><IMG SRC=/Img/trans_1x1.gif BORDER=0 WIDTH=1
HEIGHT=5><BR></TD>
<TD COLSPAN=2><FONT SIZE=3><A HREF="http://store.yahoo.com/1cache/canl6plainpa.html">
<b>Canon L-6000 Plain Paper Laser Fax, Copier, Scanner, Printer</b></A></TD>



<TD ALIGN=right><FONT SIZE=2 ><B>&nbsp;&nbsp;$685.95</TD></TR>
<TR><TD></TD><TD><A HREF="http://store.yahoo.com/1cache/canl6plainpa.html">
<IMG SRC="http://store1.yimg.com/I/1cache_1550_77381156" WIDTH=70 HEIGHT=62 BORDER=0
HSPACE=0 VSPACE=0></A></TD>
<TD><FONT SIZE=2>Enjoy Great Looking Documents While You Save Time. These days there's
never enough time to do a job right, unless you have a MultiPASS L6000. It delivers crisp
laser printing  ... </FONT></TD><TD>
</TD></TR>

......
```

Figure A.1: Excerpt of the HTML page in Source 1 (1 Cache) containing electronic appliance catalogs.

```
......

<span class="body"><div class="searchBuffer">
<span class="body"><b>&#149;</b></span>
<a href="product.asp?sku=30275052" class="productName"><b>C++ Primer</b></A>
<br>
        &nbsp;&nbsp;<span class="author">Authors:</span>&nbsp;
<a href="searchresults.asp?qutype=2&qu=Lippman%2C+Stanley+B%2E&search_store=3"
class="productDescription">
<B>Lippman, Stanley B.</B></A>
&
<a href="searchresults.asp?qutype=2&qu=Lajoie%2C+Jose&search_store=3"
class="productDescription"><B>Lajoie, Jose</B></A>
        &nbsp;&nbsp;<span class="publisher">Publisher:</span>&nbsp;
<a href="searchresults.asp?qutype=5&qu=Addison+Wesley+Publishing+Company&search_store=3"
class="productDescription"><b>Addison Wesley Publishing Company</b></A>
&nbsp;&nbsp;<span class="date">Publish Date:&nbsp;<b>4/1/98</b></span>
&nbsp;&nbsp;<span class="">Format:</span>&nbsp;<span class="productDescription">
<b>Paperback</b></span>
<br>
&nbsp;&nbsp;<span class=""><b>Our Price:&nbsp;</b></span>
<span class="price"><b>$47.47</b><p></span>
</span>
<br>
<br>&nbsp;<br></div>


<span class="body"><div class="searchBuffer">
<span class="body"><b>&#149;</b></span>
<a href="product.asp?sku=30075808" class="productName"><b>Practical Object-Oriented
Development in C++ and Java</b></A>
<br>
        &nbsp;&nbsp;<span class="author">Author:</span>&nbsp;
<a href="searchresults.asp?qutype=2&qu=Horstmann%2C+Cay+S%2E&search_store=3"
class="productDescription"><B>Horstmann, Cay S.</B></A>
        &nbsp;&nbsp;<span class="publisher">Publisher:</span>&nbsp;
<a href="searchresults.asp?qutype=5&qu=John+Wiley+%26+Sons&search_store=3"
class="productDescription"><b>John Wiley & Sons</b></A>
&nbsp;&nbsp;<span class="date">Publish Date:&nbsp;<b>4/1/97</b></span>
&nbsp;&nbsp;<span class="">Format:</span>&nbsp;<span class="productDescription">
<b>Paperback</b></span>
<br>
&nbsp;&nbsp;<span class=""><b>Our Price:&nbsp;</b></span>
<span class="price"><b>$36.89</b><p></span>
</span>
&nbsp;&nbsp;&nbsp;<span class="body">You Save:</span> <span class="price">$8.10</span>

......
```

Figure A.2: Excerpt of the HTML page in Source 14 (buy.com) containing book catalogs.

```
                 *   *   * Seminar *   *   *

          Dept. of Systems Engineering & Engineering Management
                  Chinese University of Hong Kong

------------------------------------------------------------------------

Title    :        Fat-Btree: An Update-Conscious Parallel Directory Structure

Speaker :         Prof. Haruo Yokota
                  Information Science and Engineering in Tokyo Institute of Technology

Venue    :        Rm 122, HSH Engineering Building, CUHK

Date     :        Dec. 15, 1999, Wednesday

Time     :        4:00pm-5:00pm

Abstract  :

We propose a parallel directory structure, Fat-Btree, to improve
high-speed access for parallel database systems in shared-nothing
environments.  The Fat-Btree has a three-fold aim: to provide an
indexing mechanism for fast retrieval in each processor, to balance
the amount of data among distributed disks, and to reduce
synchronization costs between processors during update operations.
......

Biography  :

Haruo Yokota received the B.E., M.E., and D.Eng. degrees from Tokyo
Institute of Technology in 1980, 1982, and 1991, respectively.  He
joined Fujitsu Ltd. in 1982, and was a researcher at the Institute
of New Generation Computer Technology (ICOT) from 1982 to 1986, and
at Fujitsu Laboratories Ltd. from 1986 to 1992.  From 1992 to 1998,
he was an Associate Professor in Japan Advanced Institute of Science
and Technology (JAIST).
......

Enquiry : Dr. H. Yan (2609-8329)

Email : yan@se.cuhk.edu.hk
```

Figure A.3: Sample seminar announcement 1 from CUHK.

```
Organised by Department of Automation and Computer-Aided Engineering,
The Chinese University of Hong Kong
======================================================================

Title   :  Global Output Regulation of Nonlinear Systems
Speaker :  Prof. T.J. Tarn
           Department of Systems Science & Mathematics, Washington University
Date    :  May 9, 2001 (Wednesday)
Time    :  3:30p.m.-4:45p.m.
Venue   :  Room 416, Mong Man Wai Building, CUHK

Abstract :
--------
This presentation focuses on the problems of global output regulation
for MIMO nonlinear systems.  The output regulation problem is to design
a feedback control law by which the output of a nonlinear system
asymptotically tracks a trajectory generated by an exosystem, no matter
how large the initial output tracking error is.  Moreover, undesired
disturbances are simultaneously rejected.  Whenever the exosignal is
absent, this control law must also be capable of globally asymptotically
stabilizing the system.
......

Biography
---------
Professor T. J. Tarn is currently a Professor and the Director of the
Center for Robotics and Automation at Washington University.  He served
as the President of the IEEE Robotics and Automation Society, 1992-1993,
the Director of the IEEE Division X (Systems and Control), 1995-1996,
and a member of the IEEE Board of Directors, 1995-1996.  He currently
serves as the Vice President for Publications of the IEEE Neural Network
Council and the Chairman of the management committee of the IEEE/ASME
Transactions on Mechatronics.
......

              *****  ALL ARE WELCOME  *****

*  Refreshment: provided between 3:30 p.m. and 3:45 p.m.

Enquiries: Ms Kan or Prof. Michael Wang, Department of Automation and
Computer-Aided Engineering, CUHK at 2609 8343.  * ACAE Seminar Series
(2000-2001) is contained in the World-Wide Web home page at
http://www.acae.cuhk.edu.hk/seminar.htm
```

Figure A.4: Sample seminar announcement 2 from CUHK.

```
<0.21.4.92.11.15.11.lydia+@PROOF.ERGO.CS.CMU.EDU (Lydia Defilippo).0>
Type:     cmu.cs.scs
Who:      Leslie Lamport
          Digital Equipment Corporation
          Systems Research Center
Topic:    The Temporal Logic of Actions
Dates:    28-Apr-92
Time:     3:30 PM
Place:    Wean Hall 4623
PostedBy: lydia+ on 21-Apr-92 at 11:15 from PROOF.ERGO.CS.CMU.EDU (Lydia Defilippo)
Abstract:


                SPECIAL SEMINAR

 Speaker:  Leslie Lamport
           Digital Equipment Corporation
           Systems Research Center

 Date:     Tuesday, April 28
 Time:     3:30 pm
 Place:    Wean Hall 4623
 Topic:    The Temporal Logic of Actions

Traditional verification employs a programming language for writing the
program, and a logic for expressing its properties.  When verifying
concurrent programs, we reason about an abstract program that is not meant
to be directly compiled and executed, so we don't have to use a programming
language.  Contrary to what one might conclude from the literature, logic
can be simpler than programming languages.  With the right logic, it is
easy to express a program as a logical formula.  The distinction between
properties and programs then vanishes, and specifications can be written
as simple programs.  Proving that one program implements another is no
different from proving that a program satisfies a property.

The temporal logic of actions is the right logic.  It provides a tool that
can help us cope with the complexity of verifying real concurrent algorithms.

No previous fondess for logic is assumed.
```

Figure A.5: Sample seminar announcement 1 from CMU.

```
<0.26.2.92.10.33.14.stankus+@STANKUS.ADM.CS.CMU.EDU (Terri Stankus).0>
Type:      cmu.cs.scs
Topic:     Distinguished Lecture--Today
Dates:     26-Feb-92
Time:      3:30 PM
PostedBy:  stankus+ on 26-Feb-92 at 10:33 from STANKUS.ADM.CS.CMU.EDU (Terri Stankus)
Abstract:


            GRAND CHALLENGES FOR MACHINE LEARNING

                  Jaime Carbonell
               School of Computer Science
               Carnegie Mellon University

                     3:30 pm
                  7500 Wean Hall

Machine learning has evolved from obscurity in the 1970s into a
vibrant and popular discipline in artificial intelligence during the
1980s and 1990s.  As a result of its success and growth, machine
learning is evolving into a collection of related disciplines:
inductive concept acquisition, analytic learning in problem
solving (e.g. analogy, explanation-based learning), learning
......
```

Figure A.6: Sample seminar announcement 2 from CMU.

```
<0.05.12.91.08.50.06.valdes+@CS.CMU.EDU (Raul Valdes-Perez).0>
Type:     cmu.cs.scs
Topic:    Re: Seminar notice (STC)
Dates:    6-Dec-91
Time:     4:00 PM
PostedBy: valdes+ on 05-Dec-91 at 08:50 from CS.CMU.EDU (Raul Valdes-Perez)
Abstract:
Ok, I guess one needs to know the time/place also.

The talk by Jonathan Minden is at 4pm this Friday, Dec. 6
in 448 Mellon Institute.  The Mellon Institute is the massive
building with the tall columns on 5th Avenue;  it houses
Biology, Chemistry, Pgh Supercomputing Center, etc.
Raul
```

Figure A.7: Sample seminar announcement 3 from CMU.

Appendix B

Detailed Experimental Results of Wrapper Adaptation of HISER

This Appendix shows the detailed experimental results of wrapper adaptation of HISER. We have conducted experiments on two semi-structured Web information sources: online electronic appliance catalogs and online book catalogs.

Table B.1 and Table B.3 show the results of the the first set of experiment on the two experimental domains respectively. They show the results of simply applying the learned wrapper of a particular information source to other unseen information sources for extraction without adaptation. The i^{th} row in Table B.1 represents an experiment of extracting contents from other information sources by using the learned wrapper of information source i.

Table B.2 and Table B.4 show the results of the second set of experiment for the domain of online electronic appliance catalogs and online book catalogs respectively. They show the experimental results of adapting a learned wrapper of a particular information source to the remaining unseen information sources. The i^{th} row in these tables represents an experiment of adapting the learned wrapper of information source i to all other information sources.

	S1		S2		S3		S4		S5		S6		S7		S8		S9		S10	
	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P
S1	98.3	100	95.7	99.6	99.6	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	97.9	99.3	98.8	98.8	98.9	99.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	98.2	99.0	98.8	98.8	100	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S2	98.3	100	95.7	99.6	99.6	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	97.9	99.3	98.8	98.8	98.9	99.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	98.3	99.0	98.8	98.8	100	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S3	98.3	100	95.7	99.6	99.6	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	97.9	99.3	98.8	98.8	98.9	99.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	98.3	99.0	98.8	98.8	100	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S4	0.0	0.0	0.0	0.0	0.0	0.0	79.0	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	94.7	98.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	99.9	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.4	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	98.3	99.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.7	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	76.0	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.5	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.3	100	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	77.4	99.1	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.3	100	0.0	0.0	0.0	0.0	0.0	0.0
S8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100	100	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100	100	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	84.3	100	0.0	0.0	0.0	0.0
S9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	85.7	98.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	86.3	99.3	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	86.9	99.4	0.0	0.0
S10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	87.2	99.7
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	96.4	100
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	95.0	98.0

Table B.1: Experimental results of applying a learned wrapper of an information source without adaptation to extract contents from the remaining unseen information sources in the electronic appliance catalog domain. (R and P denote recall and precision respectively in terms of percentage)

	S1		S2		S3		S4		S5		S6		S7		S8		S9		S10	
	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P
S1	-	-	92.7	98.8	95.2	96.4	60.8	78.7	73.7	94.7	62.8	94.6	92.9	97.8	71.7	90.5	50.9	67.5	28.2	35.5
	-	-	88.3	92.5	94.9	98.6	0.0	0.0	0.0	0.0	20.5	60.7	14.9	25.5	15.9	35.0	0.0	0.0	71.6	80.8
	-	-	98.6	98.6	98.6	100	34.7	52.5	97.8	98.5	85.7	99.6	98.2	98.8	95.2	99.6	96.4	98.4	97.8	98.2
S2	90.3	98.0	-	-	97.6	100	55.1	68.5	76.6	95.2	85.9	100	88.8	95.8	75.2	99.6	47.5	56.7	36.3	51.7
	88.6	85.3	-	-	86.0	96.1	0.0	0.0	0.0	0.0	21.8	80.0	34.9	55.0	25.7	41.3	0.0	0.0	50.2	58.7
	98.4	99.6	-	-	99.6	100	41.4	59.8	98.7	100	85.5	98.5	97.5	100	95.2	99.6	96.8	98.4	98.3	98.3
S3	92.5	98.8	91.8	99.3	-	-	58.5	70.7	84.4	96.6	78.8	95.7	91.5	97.8	75.7	98.5	41.7	60.2	33.6	46.0
	91.3	92.6	91.1	92.8	-	-	0.0	0.0	0.0	0.0	22.4	78.8	39.1	60.0	14.9	25.6	0.0	0.0	50.8	60.5
	99.1	99.3	98.8	98.8	-	-	38.5	63.5	97.9	100	85.7	97.8	97.5	98.7	97.2	99.6	94.9	97.8	98.8	99.5
S4	80.7	85.1	77.5	80.5	78.0	81.0	-	-	22.6	46.7	77.0	97.2	79.3	99.5	53.9	88.7	33.8	55.6	68.4	78.8
	0.0	0.0	0.0	0.0	0.0	0.0	-	-	20.5	55.3	0.0	0.0	11.0	36.7	17.5	31.5	24.2	37.2	50.8	58.5
	82.8	98.1	83.5	98.3	88.0	99.6	-	-	46.1	78.3	88.7	98.5	82.0	99.3	75.2	99.6	89.5	96.5	98.3	100
S5	73.5	96.8	71.3	95.9	78.0	92.5	29.3	55.9	-	-	75.7	93.5	86.8	88.6	0.0	0.0	68.6	88.8	69.4	88.4
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-	-	0.0	0.0	0.0	0.0	36.2	51.5	0.0	0.0	53.8	78.5
	96.9	98.7	98.8	98.8	98.0	99.2	35.3	66.5	-	-	99.5	100	88.5	98.8	75.2	89.6	99.4	100	98.3	99.2
S6	79.0	98.3	77.3	96.5	73.9	97.2	39.7	83.1	84.9	88.9	-	-	88.7	94.5	51.9	83.7	0.0	0.0	68.3	85.8
	28.5	31.0	32.5	38.7	35.7	39.9	0.0	0.0	0.0	0.0	-	-	28.8	48.5	0.0	0.0	48.8	87.8	0.0	0.0
	83.8	98.9	88.9	97.8	84.5	100	32.5	78.8	96.8	99.8	-	-	98.5	100	75.2	96.9	69.5	100	73.5	88.3
S7	88.3	99.0	87.6	98.5	89.0	97.9	39.3	75.0	85.7	98.6	76.4	88.3	-	-	79.8	92.9	56.8	94.2	59.4	65.8
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	58.6	74.3	-	-	0.0	0.0	0.0	0.0	38.3	50.2
	89.0	98.0	97.6	98.8	99.2	100	42.2	78.8	98.5	100	99.5	100	-	-	75.0	91.1	75.9	97.2	98.3	100
S8	78.0	88.5	87.3	89.5	87.3	88.4	79.3	98.7	0.0	0.0	48.5	78.8	88.6	96.2	-	-	0.0	0.0	68.7	87.9
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	23.2	55.5	0.0	0.0	0.0	0.0	-	-	22.8	36.4	0.0	0.0
	88.3	98.6	98.8	98.8	97.5	99.3	38.2	48.0	85.2	95.7	79.5	88.6	87.7	95.6	-	-	62.8	86.4	91.8	98.8
S9	55.9	84.3	59.2	89.0	63.6	90.6	39.3	66.5	82.6	96.8	0.0	0.0	54.3	86.9	0.0	0.0	-	-	48.4	69.4
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	57.0	87.9	0.0	0.0	0.0	0.0	-	-	42.6	75.8
	61.3	96.6	69.3	98.7	63.9	99.6	79.1	83.3	98.6	98.9	67.2	99.2	88.8	95.8	75.2	99.6	-	-	98.3	100
S10	18.5	45.2	15.7	38.5	19.1	46.2	59.3	75.0	88.8	97.9	78.0	86.4	68.5	81.2	83.9	77.5	42.8	60.3	-	-
	21.2	27.0	22.5	31.2	22.7	29.4	0.0	0.0	75.2	86.8	0.0	0.0	38.5	48.9	0.0	0.0	26.0	88.4	-	-
	94.7	100	98.6	99.4	97.7	100	44.4	54.0	86.5	95.1	99.5	100	82.5	91.2	95.1	99.5	99.4	100	-	-

Table B.2: Experimental results of adapting a learned wrapper of an information source to the remaining unseen information sources in the electronic appliance catalog domain. (R and P denote recall and precision respectively in terms of percentage)

	S11		S12		S13		S14		S15		S16		S17		S18	
	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P
	96.2	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S11	96.6	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	96.6	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	98.4	99.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S12	0.0	0.0	89.4	89.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	98.9	99.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	97.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S13	0.0	0.0	0.0	0.0	93.0	99.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	100.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	94.0	96.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
S14	0.0	0.0	0.0	0.0	0.0	0.0	95.8	97.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	94.0	96.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	97.1	99.6	0.0	0.0	0.0	0.0	0.0	0.0
S15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	97.9	97.9	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	85.7	100.0	0.0	0.0	0.0	0.0
S16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	83.9	100.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	84.5	100.0	0.0	0.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.5	100.0	0.0	0.0
S17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	100.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	100.0	0.0	0.0
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	96.0	100.0
S18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	98.7	98.7
	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	100.0

Table B.3: Experimental results of applying a learned wrapper of an information source without adaptation to extract contents from the remaining information sources in the book catalog domain. (R and P denote recall and precision respectively in terms of percentage)

	S11		S12		S13		S14		S15		S16		S17		S18	
	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P
	-	-	67.3	78.2	63.2	75.8	0.0	0.0	52.1	67.8	83.7	95.8	84.5	95.5	68.5	77.7
S11	-	-	31.8	48.9	23.2	45.5	0.0	0.0	0.0	0.0	58.9	68.2	0.0	0.0	34.4	52.5
	-	-	75.2	95.6	72.6	85.8	96.2	99.0	94.0	97.8	96.8	99.5	95.8	98.6	85.2	95.2
	73.2	82.0	-	-	67.5	72.5	80.1	88.7	60.7	79.5	60.5	97.5	36.3	58.5	44.9	88.5
S12	28.8	45.0	-	-	28.4	40.8	0.0	0.0	36.5	62.4	40.2	79.3	0.0	0.0	0.0	0.0
	93.5	100.0	-	-	87.8	97.8	95.8	98.5	88.7	95.0	86.8	100.0	62.5	79.3	66.5	87.3
	65.4	85.9	65.3	80.8	-	-	0.0	0.0	80.9	89.5	68.1	94.3	25.6	37.4	86.3	95.9
S13	15.7	37.5	45.2	68.8	-	-	0.0	0.0	33.9	62.0	0.0	0.0	0.0	0.0	27.6	44.7
	84.9	95.2	85.2	85.6	-	-	82.6	96.8	98.7	100.0	86.2	99.2	77.6	96.5	94.7	96.7
	25.4	38.5	67.1	82.4	0.0	0.0	-	-	72.3	91.9	70.2	86.8	30.6	77.5	66.7	82.3
S14	18.8	25.8	0.0	0.0	0.0	0.0	-	-	0.0	0.0	0.0	0.0	0.0	0.0	54.1	72.0
	68.4	100.0	87.1	89.9	94.0	96.4	-	-	92.7	98.5	95.6	99.2	68.2	86.9	76.2	92.9
	45.8	77.4	44.9	64.4	77.5	90.6	69.0	96.3	-	-	68.7	93.8	0.0	0.0	87.5	94.3
S15	0.0	0.0	34.8	58.8	45.6	74.7	0.0	0.0	-	-	0.0	0.0	0.0	0.0	68.9	88.9
	96.1	100.0	89.2	93.9	87.5	98.8	89.8	93.2	-	-	89.3	93.3	84.5	93.1	91.7	85.2
	44.3	68.8	50.8	88.8	64.2	72.3	62.7	86.5	60.7	79.5	-	-	84.5	96.8	75.4	88.7
S16	47.8	70.2	36.6	59.3	0.0	0.0	0.0	0.0	0.0	0.0	-	-	18.4	18.8	58.4	59.5
	75.4	87.8	77.1	89.9	84.9	96.6	86.6	99.3	88.7	100.0	-	-	90.5	97.2	92.5	98.3
	35.4	68.5	34.7	44.2	37.2	70.9	32.3	60.9	0.0	0.0	54.5	88.2	-	-	49.4	70.9
S17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-	-	0.0	0.0
	96.1	100.0	69.5	86.9	75.7	98.8	69.8	82.3	87.8	95.2	83.7	94.5	-	-	56.6	92.2
	65.4	87.4	54.7	75.2	87.2	98.5	40.7	81.5	48.7	60.1	56.4	58.3	34.3	63.8	-	-
S18	28.4	35.8	0.0	0.0	54.7	88.6	0.0	0.0	42.5	68.7	28.4	38.8	0.0	0.0	-	-
	91.6	97.9	61.7	90.7	97.5	98.9	88.3	97.6	78.8	86.9	82.4	85.6	58.5	90.9	-	-

Table B.4: Experimental results of adapting a learned wrapper of an information source to the remaining information sources in the book catalog domain. (R and P denote recall and precision respectively in terms of percentage)

Bibliography

[1] A. Gupta, editor. *Integration of Information Systems: Bringing Heterogeneous Databases.* IEEE Press, 1989.

[2] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The Lorel query language for semistructured data. *International Journal on Digital Libraries*, 1:66–88, 1997.

[3] B. Adelberg. NoDoSE-a tool for semi-automatically extracting structured and semistructured data from text documents. In *Proceedings of 1998 ACM SIGMOD International Conference on Management of Data*, pages 283–294, June 1998.

[4] Defense Advanced Research Projects Agency. *Proceedings of the Sixth Message Understanding Conference (MUC-6).* Morgan Kaufmann Publisher, Inc., 1995.

[5] Defense Advanced Research Projects Agency. *Proceedings of the Seventh Message Understanding Conference (MUC-7).* Morgan Kaufmann Publisher, Inc., 1998.

[6] N. Ashish and C. Knoblock. Wrapper generation for semi-structured Internet sources. *SIGMOD Record*, 26(4):8–15, December 1997.

[7] P. Atzeni and G. Mecca. Cut and paste. In *Proceedings of the Sisteenth ACM Symposium on Principle of Database Systems (PODS'97)*, pages 144–153, May 1997.

[8] C. M. Bowman, P. Danzig, U. Manber, and M. Schwartz. Scalable internet resource discovery: Research problems and approaches. *Communications of the ACM*, 37(8):98–107, August 1994.

[9] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. A query language and optimization techniques for unstructured data. In *Proceedings of 1996 ACM SIGMOD International Conference on Management of Data*, pages 505–516, 1996.

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

[11] J. Chai, A. Biermann, and C. Guinn. Two dimensional generalization in information extraction. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence(AAAI-1999)*, pages 431–438, 1999.

[12] C. H. Chang, S. C. Lui, and Y. C. Wu. Applying pattern mining to web information extraction. In *Fifth Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 4–16, 2001.

[13] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS project: integration of heterogeneous information sources. In *Proceedings of the Information Processing Society of Japan*, pages 7–18, 1994.

[14] W. Cohen. Recognizing structure in Web pages using similarity queries.

In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 59–66, July 1999.

[15] W. Cohen and W. Fan. Learning page-independent heuristics for extracting data from web pages. *Computer Networks*, 31(11-16):1641–1652, May 1999.

[16] C. Cortes and V. N. Vapnik. Support vector network. *Machine Learning*, 20:273–297, 1995.

[17] J. Cowie and W. Lehnert. Information extraction. *Communications of the ACM*, 39(1):80–91, 1996.

[18] R. B. Doorenbos, O. Etzioni, and D. S. Weld. A scalable comparison-shopping agent for the World-Wide Web. In *Proceedings of the First International Conference on Autonomous Agents*, pages 39–48, 1997.

[19] H. Drucker, D. H. Wu, and V. N. Vapnik. Support Vector Machines for spam categorization. *IEEE Transactions on Neural Network*, 10(5):1048–1054, September 1999.

[20] D. W. Embley, D. Campbell, R. D. Smith, and S. Liddle. A conceptual-modeling approach to extracting data from the Web. In *Proceedings of the 17th International Conference on Conceptual Modeling (ER'98)*, pages 78–91, November 1998.

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

[22] D. W. Embley, Y. S. Jiang, and Y. K. Ng. Record-boundary discovery in Web documents. In *Proceedings of 1999 ACM SIGMOD International Conference on Management of Data*, pages 467–478, June 1999.

[23] D. Freitag. Information extraction from HTML: Application of a general machine learning approach. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 517–523, July 1998.

[24] D. Freitag and N. Kushmerick. Boosted wrapper induction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence(AAAI-2000)*, pages 577–583, July 2000.

[25] D. Freitag and A. McCallum. Information extraction with hmms and shrinkage. In *AAAI-99 Workshop on Machine Learning for Information Extraction*, July 1999.

[26] R. Ghani, R. Jones, D. Mladenic, K. Nigam, and S. Slattery. Data mining on symbolic knowledge extracted from the Web. In *KDD-2000 Workshop on Text Mining*, August 2000.

[27] S. Grumbach and G. Mecca. In search of the lost schema. In *Proceedings of 7th International Conference on Database Theory*, pages 314–331, January 1999.

[28] J.-B. Gruser, L. Raschid, M. Vidal, and L. Bright. Wrapper generation for Web accessible data sources. In *Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems*, pages 14–23, 1998.

[29] T. Guan and K. F. Wong. KPS: a Web information mining algorithm. *Computer Networks*, 31(11-16):1495–1507, May 1999.

[30] J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha, and A. Crespo. Extracting semistructured information from the Web. In *Workshop on Management of Semistructured Data*, May 1997.

[31] C. Hsu and M. Dung. Generating finite-state transducers for semi-structured data extraction from the Web. *Journal of Information Systems, Special Issue on Semistructured Data*, 23(8):521–538, November 1998.

[32] G. Huck, P. Frankhauser, K. Aberer, and E. Neuhold. Jedi: Extracting and synthesizing information from the Web. In *Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems*, pages 32–43, 1998.

[33] S. B. Huffman. Learning information extraction patterns from examples. In *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pages 246–260, 1996.

[34] Information Sciences Institute/USC. RISE: A repository of on-line information sources used in information extraction tasks., http://www.isi.edu/muslea/rise/index.html.

[35] T. Joachims. svm^{light}, http://ais.gmd.de/ thorsten/svm_light/.

[36] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142, 1998.

[37] J. T. Kim and D. I. Moldovan. Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):713–724, October 1995.

[38] C. Knoblock, S. Minton, J. Ambite, N. Ashish, J. Modi, I. Muslea, A. Philpot, and S. Tejada. Modeling web sources for information integration. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 211–218, July 1998.

[39] N. Kushmerick. Regression testing for wrapper maintenance. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 74–79, July 1999.

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

[41] L. Lakshmanan, F. Sadri, and I. N. Subramanian. A declarative language for querying and restructing the Web. In *6th International Workshop on Research Issues in Data Engineering: Interoperability of Non-traditional Database Systems*, 1996.

[42] A. Levy, C. Knoblock, S. Minton, and W. Cohen. Trends and controversies: Information integration. *IEEE Intelligent Systems*, 13(5):12–24, 1998.

[43] D. D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka. Training algorithms for linear text classifiers. In *Proceedings of 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96)*, pages 298–306, 1996.

[44] S. J. Lim and Y. K. Ng. An automated approach for retrieving hierarchical data from HTML tables. In *Proceedings of the Eighth International Conference on Information and Knowledge Management CIKM 1999*, pages 466–474, November 1999.

[45] S. J. Lim and Y. K. Ng. WebView: A tool for retrieving internal structures and extracting information from HTML documents. In *Proceedings of the 6th International Conference on Database Systems for Advanced Applications*, pages 71–80, April 1999.

[46] W. Y. Lin and W. Lam. Learning to extract hierarchical information from semi-structured documents. In *Proceedings of the Ninth International Conference on Information and Knowledge Management CIKM 2000*, pages 250–257, November 2000.

[47] A. O. Mendelzon, G. A. Mihaila, and T. Milo. Querying the World Wide Web. *International Journal on Digital Libraries*, 1:54–67, 1997.

[48] I. Muslea. Extraction patterns for information extraction tasks: A survey. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-1999)*, 1999.

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

[50] U. Y. Nahm and R. J. Mooney. Using information extraction to aid the discovery of prediction rules from text. In *KDD-2000 Workshop on Text Mining*, August 2000.

[51] S. Nestorov, S. Abiteboul, and R. Motwani. Extracting schema from semistructured data. In *Proceedings of 1998 ACM SIGMOD International Conference on Management of Data*, pages 295–306, June 1998.

[52] J. Prager, E. Brown, A. Coden, and D. Radev. Question-Answering by predictive annotation. In *Proceedings of 23th Annual International*

ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2000), pages 184–191, 2000.

[53] B. Ribeiro-Neto, A. H. F. Laender, and A. S. da Silva. Extracting semi-structured data through examples. In *Proceedings of the Eighth International Conference on Information and Knowledge Management CIKM 1999*, pages 94–101, November 1999.

[54] E. Riloff. Automatically constructing a dictionary for information extraction tasks. In *Proceedings of the Eleventh National Conference on Artifical Intelligence*, pages 811–816, 1993.

[55] A. Sahuguet and F. Azavant. Building intelligent Web applications using lightweight wrappers. *Data and Knowledge Engineering*, 36(3):283–316, March 2001.

[56] G. Salton. *Automatic text processing, the transformation, analysis, and retrieval of information by computer.* Addison-Wesley, Reading (MA), 1989.

[57] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[58] K. Seymore, A. McCallum, and R. Rosenfeld. Learning hidden markov model structure for information extraction. In *AAAI-99 Workshop on Machine Learning for Information Extraction*, July 1999.

[59] R. D. Smith and M. Lopez. Information extraction for semi-structured documents. In *Workshop on Management of Semistructured Data*, May 1997.

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

[61] S. Soderland, D. Fisher, J. Aseltine, and W. Lehnert. CRYSTAL: Inducing a conceptual dictionary. In *Proceedings of the Fourteenth International Joint Conference on Artifical Intelligence*, pages 1314–1321, 1995.

[62] R. Srihari and W. Li. Question Answering supported by information extraction. In *The Eighth Text REtrieval Conference (TREC-8)*, pages 185–196, 1999.

[63] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

[64] E. Voorhees and D. Harman, editors. *The Eighth Text REtrieval Conference (TREC-8).* National Institute of Standards and Technology, 1999.

[65] The World-Wide Web Consortium (W3C). Extensible Markup Language (XML), http://www.w3.org/xml/.

[66] The World-Wide Web Consortium (W3C). HyperText Markup Language (HTML), http://www.w3.org/markup/.

[67] H. L. Wang, S. H. Wu, I. C. Wang, C. L. Sung, W. L. Hsu, and W. K. Shih. Semantic search on internet tabular information extraction for answering queries. In *Proceedings of the Ninth International Conference on Information and Knowledge Management CIKM 2000*, pages 243–249, November 2000.

[68] G. Wiederhold. *Intelligent Information Integration.* Kluwer, 1996.

[69] J. Y. Yang, E. S. Lee, and J. G. Choi. A shopping agent that automatically constructs wrappers for semi-structured online vendors. In *Second International Conference on Intelligent Data Engineering and Automated Learning (IDEAL2000)*, pages 368–373, December 2000.

[70] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 42–49, 1999.