CRITERIA FOR EVALUATING INFORMATION EXTRACTION SYSTEMS

Chia-Hui Chang[1], Mohammed Kayed[2], Moheb Ramzy Girgis[3] and Khaled Shaalan[4]

[1] National Central University, Taiwan, email: chia@csie.ncu.edu.tw
[2] Cairo University, Egypt, email: mskayed@yahoo.com
[3] Minia University, Egypt, email: mrgirgis@minia.edu.eg
[4] British University in Dubai (BUiD), Emirates, e-mail: khaled.shaalan@buid.ac.ae

Abstract
The Internet presents a huge amount of useful information which is usually formatted for its users, which makes it difficult to extract relevant data from various sources. Therefore, the availability of robust, flexible Information Extraction (IE) systems that transform the Web pages into program-friendly structures will become a great necessity. Although many approaches for data extraction from Web pages have been developed, there has been limited effort to compare such tools. In addition to briefly surveying the major data extraction approaches described in the literature, the paper also mainly presenting three classes of criteria for qualitatively analyzing these approaches. The criteria of the first class are concerned with the difficulties of an IE task, so these criteria are capable of determining why an IE system fails to handle some Web sites of particular structures. The criteria of the second class are concerned with the effort made by the user in the training process, so these criteria are capable of measuring the degree of automation for IE systems. The criteria of the third class are concerned with the techniques used in IE tasks, so these criteria are capable of measuring the performance of IE systems.

Keywords: Information Extraction, Wrapper, Wrapper Induction, Data Mining, Machine Learning.

1. Introduction

The explosive growth and popularity of the world-wide web has resulted in a huge amount of information sources on the Internet. However, due to the heterogeneity and the lack of structure of Web information sources, access to this huge collection of information has been limited to browsing and searching. Sophisticated Web mining applications, such as comparison shopping, require expensive maintenance to deal with different data formats. The problem of translating the contents of input documents into structured data is called information extraction (IE). Unlike information retrieval (IR), which concerns how to identify relevant documents from a document collection, IE produces structured data ready for post-processing, which is crucial to many applications of Web mining and searching tools.

IE is the task of converting text material such as newswire articles or Web pages into structured data objects suitable for automatic processing. Previously approaches in this field use techniques borrowed from areas such as machine learning (ML), natural language processing (NLP), languages and grammars, databases and ontology. The research work on IE has focused on two distinct sub-problems: the first and the conventional problem is the task of filling template slots from natural language text, and the second problem is the task of learning extraction procedures for highly structured text such as Web pages produced by CGI scripts. The first sub-problem is applied to free text documents and it uses extraction rules that are based on syntactic and semantic information such as semantic classes, thematic roles and case frames. The second sub-problem is primarily used for semi-structured information sources, and their extraction rules rely heavily on regularities in the structure of the documents.

Formally, an IE task is defined by its input and its extraction target. The input can be unstructured documents like free text that are written in natural language or the semi-structured documents that are pervasive on the Web, such as tables or itemized and enumerated lists. The extraction target of an IE task can be a relation of k-tuple (where k is the number of attributes in a record) or it can be a complex object with hierarchically organized data. For some IE tasks, an attribute may have zero (missing) or multiple instantiations in a record. The difficulty of an IE task can be further complicated when various permutations of attributes or typographical errors occur in the input documents.

Programs that perform the task of IE are referred to as extractors or wrappers. A wrapper was originally defined as a component in an information integration system which aims at providing a single uniform query interface to access multiple information sources. In an information integration system, a wrapper is generally a program that "wraps" an information source (such as a database server, or a Web server) such that the information integration system can access that information source without changing its core query answering mechanism. In the case where the information source is a Web server, a wrapper must perform information extraction to extract the contents in HTML documents. Since many IE systems reviewed in this article are aimed at Web information integration applications, we use the terms IE systems and wrappers interchangeably[1].

[1] Rigorously, a Web wrapper includes page fetching and data extraction, however, in recent literature the term wrapper induction is used to denote the generation of extractors.

Wrapper induction (WI) systems are software tools that are designed to generate wrappers. A wrapper usually performs a pattern matching procedure (e.g., a form of finite-state machines) which relies on a set of extraction rules. Tailoring a WI system to a new requirement is a task that varies in scale depending on the text type, domain, and scenario. To maximize reusability and minimize maintenance cost, designing a trainable WI system has been an important topic in the research fields of message understanding, machine learning, pattern mining, etc. The task of Web IE differs largely from traditional IE tasks in that traditional IE aims at extracting data from totally unstructured free texts that are written in natural language. Web IE, in contrast, processes online documents that are semi-structured and usually generated automatically by a server-side application program. As a result, traditional IE usually takes advantage of NLP techniques such as lexicons and grammars, whereas Web IE usually applies machine learning and pattern mining techniques to exploit the syntactical patterns or layout structures of the template-based documents.

In this paper, we will compare different IE systems using features from three dimensions which we regard as criteria for evaluating IE systems. Section 2 introduces related work on IE system taxonomy, which we can summarize into three dimensions of evaluating IE systems. Section 3 suggests the criteria for each dimension. We make a survey of contemporary IE tools in Section 4 and present a comparative analysis of the surveyed IE tools from the three dimensions in Section 5. Our conclusions are made in Section 6.

2. Related Work

In the past few years, many approaches to WI systems, including machine learning and pattern mining techniques, have been proposed, with various degrees of automation. In this section we survey the previously proposed taxonomies for IE tools developed by the main researchers.

The Message Understanding Conferences (MUCs) have inspired the early work in IE. There are five main tasks defined for text IE, including named entity recognition, coreference resolution, template element construction, template relation construction and scenario template production. The significance of the MUCs in the field of IE motivates some researchers to classify IE approaches into two different classes: MUC Approaches (e.g., AutoSolg [Riloff, 1993], LIEP [Huffman, 1996], PALKA [Kim, 1995], HASTEN [Krupka, 1995], and CRYSTAL [Soderland, 1995]) and Post-MUC Approaches (e.g., WHISK [Soderland, 1999], RAPIER [califf, 1998], SRV [Freitag, 1998], WIEN [Kushmerick, 1997], SoftMealy [Hsu, 1998] and STALKER [Muslea, 1999]).

Hsu and Dung [Hsu, 1998] classified wrappers into 4 distinct categories, including hand-crafted wrappers using general programming languages, specially designed programming languages or tools, heuristic-based wrappers, and WI approaches. Chang [Chang, 2003] followed this taxonomy and compared IE systems from the user point of view and discriminated IE tools based on the degree of automation. They classified IE tools into four distinct categories, including systems that need programmers, systems that need annotation examples, annotation-free systems and semi-supervised systems.

Muslea, who maintains the RISE (Repository of Online Information Sources Used in Information Extraction Tasks) Web site, classified IE tools into 3 different classes according to the type of input documents and the structure/constraints of the extraction patterns [Muslea, 1999]. The first class includes tools that process IE from free text using extraction patterns that are mainly based on syntactic/semantic constraints. The second class is called Wrapper induction systems which rely on the use of delimiter-based rules since the IE task processes online documents such as HTML pages. Finally, the third class also processes IE from online documents; however the patterns of these tools are based on both delimiters and syntactic/semantic constraints.

Kushmerick classified many of the IE tools into two distinct categories finite-state and relational learning tools [Kushmerick, 2003]. The extraction rules in finite-state tools are formally equivalent to regular grammars or automata, e.g WIEN, SoftMealy and STALKER, while the extraction rules in relational learning tools are essentially in the form of Prolog-like logic programs, e.g. SRV, Crystal, WebFoot [Soderland, 1997], Rapier and Pinocchio [Ciravegna, 2000].

Laender proposed a taxonomy for data extraction tools based on the main technique used by each tool to generate a wrapper [Laender, 2002a]. These include languages for wrapper development (e.g., Minerva [Crescenzi, 1998], TSIMMIS [Hammar, 1997] and WebOQL [Arocena, 1998]), HTML-aware tools (e.g., W4F [Saiiuguet, 2001], XWrap [Liu, 2000] and RoadRunner [Crescenzi, 2001]), NLP-based tools (e.g., WHISK, RAPIER and SRV), Wrapper induction tools (e.g., WIEN, SoftMealy and STALKER), Modeling-based tools (e.g., NoDoSE [Adelberg, 1998] and DEByE [Laender, 2002b; Ribeiro-Neto, 1999]), and Ontology-based tools (e.g., BYU [Embley, 1999]). Laender compared among the tools by using the following 7 features: degree of automation, support for complex objects, page contents, availability of a GUI, XML output, support for non-HTML sources, resilience and adaptiveness.

Sarawagi classified HTML wrappers into 3 categories according to the kind of extraction tasks [Sarawagi, 2002]. The first category, record-level wrappers, exploits regularities to discover record boundaries and then extract elements of a single list of homogeneous records from a page. The second category, page-level wrappers, extracts elements of multiple kinds of records. Finally, the site-level wrappers populate a database from pages of a Web site.

Kuhlins and Tredwell classified the toolkits for generating wrappers into two basic categories, based on commercial and non-commercial availability [Kuhlins, 2002]. They also contrasted the toolkits by using some features such as output methods, interface type, web crawling capability and GUI support.

This survey shows three main dimensions for evaluating IE systems. First, the distinction of free text IE and online documents made by Muslea, the three-level of extraction tasks proposed by Sarawagi, and the capabilities of handling non-HTML sources, together suggest the first dimension, which concerns the difficulty or the task domain that an IE task refers to. Second, the categorizations of programmer-involved, learning-based or annotation-free approaches, imply the second dimension which concerns the degree of automation. Finally, the taxonomy of regular expression rules or Prolog-like logic rules, and that of deterministic finite-state transducer or probabilistic hidden Markov models, prompts the third dimension which relates the underlying techniques used in IE systems. These three dimensions are discussed in the next section.

3. Three Dimensions for Comparing IE Systems

Continuing our survey of various taxonomies, there are three dimensions to be used in the comparison. The first dimension evaluates the difficulty of an IE task, which can be used to answer the question "why an IE system fails to handle some Web sites with particular structures?" The second dimension evaluates both the effort made by the user for the training process and the necessity to port an IE system across different domains. The third dimension compares the techniques used in different IE systems. From the user's point of view, the last dimension is less important. However, researchers might get an overview of which machine-learning or data mining technologies have been used for WI through the comparison. In this section we describe each of these dimensions, and for each one we include a set of features that can be used as criteria for evaluating IE systems from the dimension prospective.

3.1 Task difficulties

The input file of an IE task may be structured, semi-structured or free-text. Extracting relevant data from structured Web pages (e.g. XML pages) is comparably simpler, whereas extracting relevant data from free-text Web pages requires syntactic and semantic information, such as part-of-speech and lexical semantic tagging. However, extracting data from structured and free-text Web sites is out of the scope of this paper. Semi-structured inputs are the documents of a fairly regular structure and data in them may be presented in HTML or non-HTML format. Most of the semi-structured HTML pages in a Web site are generated using a common template or layout. For example, the Amazon Web site lays out the author, title, price, comments, etc. in a similar way for all its book pages. Other sites may contain semi-structured HTML pages generated by hand, where each page has its own layout. Therefore, a semi-structured Web page may be singleton or template-based. A singleton page is one where no other pages have the same layout. A site home page may be an example of a singleton page. A template-based Web page is one that is automatically generated from a back-end DBMS using scripts.

Depending on the extraction target, an IE task domain can be classified into the three categories: record-level, page-level and site-level IE tasks. Record-level wrappers discover record boundaries and then divide them into separate attributes; page-level wrappers extract elements of multiple kinds of records, while site-level wrappers populate a database from pages of a Web site. In recent years, academic researchers have devoted much effort to develop record-level and page-level data extraction, whereas industrial researchers have more interest in complete suites which support site-level data extraction.

Assume an extraction target is a data-object consisting of a set of attributes (A_1, A_2, …, A_n), $n \geq 1$; where each attribute represents a relevant datum to be extracted for the user. A data-object may be of a plain/nested structure. A plain text data-object is a tuple with a fixed number of attributes and values, while a nested data-object is a tuple with a variable number of attributes and values. The set of attributes that form a data-object is subject to the following variations:

- An attribute may have zero or more values in a data-object. If the attribute has zero value, it is called a *missing* attribute, if it has more than one value, it is called a *multi-valued* attribute. The name of a book's author may be an example of multi-valued attribute, whereas a special offer, which is available only for certain books, is an example of missing attribute.
- The set of attributes (A_1, A_2, …, A_n) may have multiple ordering, i.e., an attribute Ai may have variant positions in different instances of a data-object; and we call this attribute a *multi-ordering* attribute. For example, a movie site might list the release date before the title for movies prior to 1999, but after the title for recent movies.
- An attribute may have fixed/variant formats along with different instances of a data-object. If the format of an attribute is fixed/variant, we call it a *fixed/variant-format* attribute. For example, an e-commerce site might list prices in bold face, except for sale prices which are in red. So, price would be an example of a variant-format attribute in this site.
- Different attributes in a data-object may have the same format. If an attribute A_i has the same display format with another attribute A_j ($i \neq j$), we call each one an *undistinguishable* attribute.
- Depending on the tokenization methods used, sometimes an attribute can not be decomposed into different tokens. Such an attribute is called a *untokenized* attribute. For example, in a college course

catalogue the department code has no delimiter separated it from the course number in strings such as "COMP4016" or "GEOL2001".

- For patterns based on string comparison, some extraction rules consist of non-contiguous delimiters, which we call *sequential-pattern* attributes.

This description for the structure of Web sites motivates us to suggest some features for evaluating IE systems from the task domain aspect. These features are *page type*, *non-HTML support*, *singleton/template-based pages*, *record/page/site level extraction*, *nested data-object*, *missing attributes*, *multi-valued attributes*, *multi-ordering attributes*, *fixed/variant-format attributes*, *undistinguishable attributes*, *untokenized attributes* and *sequential-pattern attributes*. These features represent most of the difficulties that may occur in a Web site. We claim that, analyzing how an IE system supports these features can be used to evaluate this system.

3.2 Automation Degree

WI systems have many phases to be accomplished: collecting training pages, labeling training examples, generalizing extraction rules, extracting the relevant data, and outputting the result in an appropriate format. The first phase can be done manually by the user or automatically by using a crawler or robot that navigates the target site and collects a set of training pages. The second phase specifies the output of an extraction task. However, some IE systems do not require the collected training examples to be labeled before the learning stage, instead, the labeling or annotation of the extracted data can be done after the generation of extraction rules. In the third phase, IE systems use the training pages (labeled or unlabeled) to generate the extraction rules that will be used to extract relevant data from the documents in the fourth phase. Finally, the system will output relevant data in an appropriate format (such as general text, back-end DBMS or XML format).

Therefore, we suggest the following features which we regard as criteria for evaluating WI systems from the automation degree prospective, including *GUI support* for labeling and debugging, *crawler support* for collecting training pages, *training examples* required for rule generalization, *output support* and *API support* for integration.

3.3 The Techniques Used

To induce a wrapper, many techniques have been applied. The extractor architecture may require single or multiple passes over the pages. The type of extraction rules may be expressed using regular grammars or logic rules. Some of the WI systems use DOM-tree path addressing as the features in extraction rules; some use syntactic or semantic constraints, such as POS-tags and WordNet semantic class; while others use delimiter-based constraints, such as HTML tags or literal words, in the extraction rules. There are various granularities for input string tokenization, including tag-level and word-level encoding. The former encoding translates each HTML tag as a token and translates any text string between two tags as a special token, while the later, word-level, treats each word in a document as a token. Extraction rules can be induced by top-down or bottom-up generalization, pattern mining, or logic programming.

Therefore, we suggest some features which we regard as criteria for evaluating IE systems from the perspective of techniques used. These features are *scan pass*, *extraction rule type*, *features used*, *tokenization/encoding* schemes and *learning algorithm*.

4. Survey for Contemporary IE Systems

WI is a task of automatically learning a wrapper used for an information resource. To construct a wrapper, we need both a semantic model of the source that describes the attributes available on that type of page and a syntactic model, or grammar, that describes the page format, so the attributes can be extracted.

Let us consider the two main tasks: rule generation and user interaction in WI systems. In the first task, although many researchers use different rule representation and different techniques for wrapper generation, all of them exploit the regularities in the structure of the semi-structured Web pages to generate their wrappers. The main goal of this task is to generate an accurate wrapper that can handle most of the Web site's problems which are mentioned in Section 3.1. In the second task, the role of the user in the process of wrapper generation represents an important dimension. Thus, the degree of automation is a good metric that can be used to classify WI systems into the four classes *manually-constructed IE Systems*, *supervised IE Systems*, *semi-supervised IE Systems* and *unsupervised IE Systems*.

4.1 Manually-constructed IE systems

As shown on the right of Figure 1, in manually-constructed IE systems, users program a wrapper for each Web site by hand by using some programming languages such as Perl or by using special languages. These tools require the user to have substantial computer and programming backgrounds, so it becomes expensive. Such systems include TSIMMIS, Minerva, Web-OQL, W4F and XWRAP.

Minerva: A process for writing wrappers over semi-structured data sources [Crescenzi, 1998]. It attempts to combine the advantages of a declarative grammar-based

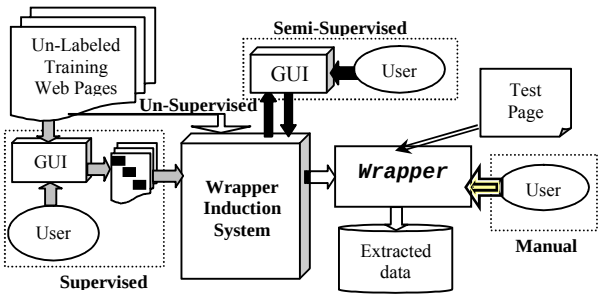

Fig 1: A General View for WI Systems

approach with the flexibility of procedural programming in handling heterogeneities and exceptions. This is done by incorporating an explicit exception-handling mechanism inside a regular grammar. Whenever the parsing of that production fails, an exception is raised and the corresponding exception handler is executed. The language used by Minerva to handle exceptions is called EDITOR.

TSIMMIS: A system that provides integrated access to a wide variety of heterogeneous data sources [Hammar, 1997]. The main component of this project is a wrapper that takes as input a specification that declaratively states where the data of interest is located on the pages and how the data should be "packaged" into objects. TSIMMIS outputs data in OEM (Object Exchange Model) that contains the desired data together with information about the structure and the contents of the result. It provides two important operators: *split* and *case*. The split operator is used to divide the input list element into individual elements. The case operator allows the user to handle the irregularities in the structure of the HTML pages.

WebOQL: A declarative query language that synthesizes ideas from different research areas such as Web queries, semi-structured data and website restructuring [Arocena, 1998]. It used a generic wrapper that parses HTML pages and produces an abstract syntax tree called *hypertree* for them. The language provided by WebOQL is powerful enough to navigate, query and restructure these trees. WebOQL is capable of querying pages with irregular structure, and pages whose structure is not fully known since hypertrees support collections, nesting and ordering.

W4F (World-Wide Web Wrapper Factory): This is a Java toolkit to generate Web wrappers [Saiiuguet, 2001]. The wrapper development process consists of three independent layers: *retrieval*, *extraction* and *mapping* layers. In the retrieval layer, a to-be-processed document is retrieved, cleaned and then fed to an HTML parser that constructs a parse tree following the Document Object Model (DOM). In the extraction layer, extraction rules are applied on the parse tree to extract information and then store them into the W4F internal format called Nested String List (NSL). In the mapping layer, the NSL structures are exported to the upper-level application according to mapping rules. Extraction rules are expressed using the HTML Extraction Language (HEL) that permits a declarative specification of IE. The language offers conditions, regular expressions and some constructs to build complex structures.

XWrap: A system that exploits formatting information in Web pages to hypothesize the underlying semantic structure of a page, and then encode the hypothetical structure and the IE knowledge of the web pages in a rule-based declarative language designed specifically for XWrap [Liu, 2000]. The wrapper generation process includes 3 phases: *structure analysis*, *source-specific XML generation* and *generic XML-based content filter*. In the first phase, XWrap fetches, cleans up, and generates a tree-like structure for the page. The system then identifies regions, semantic tokens of interest and useful hierarchical structures of sections of the page. In the second phase, the system generates a source-specific XML template file based on the content tokens and the nesting hierarchy specification, and then constructs a source-specific XML generator. In the last phase, after wrapping a Web source in an XML format, the generic XML-based content filter is capable of handling complex conjunctive or disjunctive queries handed over by various mediator applications.

4.2 Supervised IE systems

As shown in the left-bottom of Figure 1, supervised IE systems take a set of web pages labeled with examples of the data to be extracted and output a wrapper. The user provides an initial set of labeled examples and the system (with a GUI) may suggest additional pages for the user to label it. Such systems are SRV, RAPIER, WHISK, WIEN, STALKER, SoftMealy, NoDoSE and DEByE.

SRV: A top-down relational algorithm that generates single-slot extraction rules [Freitag, 1998]. It regards IE as a kind of text classification. The rules generated by SRV rely on a set of easy to implement token-oriented features. These features have two basic varieties: simple and relational. A simple feature is a function maps a token into some discrete value. A relational feature maps a token to another token. SRV makes no assumptions about the document structure or the kinds of information available for use in learning extraction rules. SRV has the ability to take advantage of orthographic, syntactic, and semantic information if such information is provided in the user-defined features.

RAPIER: A compression-based relational learning algorithm [Califf, 1998]. It learns single slot extraction patterns that make use of limited syntactic and semantic information, using freely available, robust knowledge sources such as a part-of-speech tagger or a lexicon (WordNet). The extraction rules consist of three distinct patterns. The first one is the pre-filler pattern that matches text immediately preceding the filler, the second one is the pattern that match the actual slot filler, finally the last one is the post-filler pattern that match the text immediately following the filler. RAPIER begins with the most specific rules and then replacing them with more general rules.

WHISK: A covering learning system that generates extraction rules for a wide variety of documents ranging from structured to free text [Soderland, 1999]. WHISK does not require prior syntactic analysis when it is applied to structured or semi-structured text. For free text, WHISK works best with input that has been annotated by a syntactic analyzer and a semantic tagger. Whisk uses a pattern-matching representation, which is a restricted form of regular

expressions that has two components. The first component describes the *context* that makes a phrase relevant, and the second component is one that specifies the exact *delimiters* of the phrase to be extracted. WHISK's extraction rules can extract either single or multiple slots.

WIEN: Kushmerick identified a family of six wrapper classes, LR, HLRT, OCLR, HOCLRT, N-LR and N-HLRT [Kushmerick, 1997]. The first four wrappers are used for semi-structured documents, while the remaining two wrappers are used for hierarchically nested documents. The LR wrapper is a vector of 2*K* delimiters for a site containing *K* attributes. The HLRT class uses two additional delimiters to skip over potentially-confusing text in either the head or tail of the page. The OCLR class uses two additional delimiters to identify an entire tuple in the document, and then uses the LR strategy to extract each attribute in turn. The HOCLRT wrapper combines the two classes OCLR and HLRT. The two wrappers N-LR and N-HLRT are limited to deal with irregular semi-structured data.

STALKER: A WI system that performs hierarchical data extraction [Muslea, 1999]. The system turns the difficult problem of extracting data from an arbitrary complex document into a series of easier extraction tasks. It uses an inductive learning algorithm, STALKER, that is based on the idea of using an embedded catalog (EC) formalism which can describe the structure of a wide range of semi-structured documents. The EC description of a page is a tree-like structure in which the leaves are the attributes to be extracted and the internal nodes are lists of tuples. In order to extract the items of interest, a wrapper uses the EC description of the document and a set of extraction rules.

SoftMealy: A Web WI approach that is based on a finite-state transducer (FST) and contextual rules [Hsu, 1998]. The wrapper is represented as a FST where each distinct attribute permutation in the page can be encoded as a successful path and the state transitions are determined by matching contextual rules that describe the context delimiting two adjacent attributes. Contextual rules consist of individual separators that represent invisible borderlines between adjacent tokens. An FST consists of two different parts: the *body transducer*, which extract the sub-string of the page that contains the tuples, and the *tuple transducer* which iteratively extracts the tuples from the body. The tuple transducer accepts a tuple and returns its attributes.

NoDoSE (Northwestern Document Structure Extractor)**:** An interactive tool for extracting data from semi-structured documents (plain text or HTML pages) [Adelberg, 1998]. This is accomplished by two main tasks. For the first task, the system attempts to infer the format of the input documents from the information the user has input through a GUI. Using this interface, the user hierarchically decomposes the document, outlining its interesting regions and then describing its semantics. This task is expedited by two heuristic-based mining components: one that mines text files and the other one parses the HTML code. The result of this task is a tree that describes the structure of the document. For the second task, once the format of a document has been determined, its data can be extracted.

DEByE (Data Extraction By Example)**:** An interactive data extraction system [Laender, 2002b; Ribeiro-Neto, 1999]. The user inputs to the system a small set of example data objects to describe, from his viewpoint, what to extract. The system consists of two main modules called *GUI* and *Extractor*. The user uses the GUI module to assemble (in a hierarchical structure) a small set of example objects to be extracted, which are then used to generate object extraction patterns (OEPs). These OEPs exploit a combination of structural and textual information to extract new objects from new Web pages. The extractor module applies a bottom-up extraction algorithm that, given a set of Web pages as input, recognizes objects matching the generated OEPs and extracts these objects as an output.

4.3 Semi-Supervised IE systems

The systems that we categorize as semi-supervised IE systems include IEPAD and OLERA. OLERA requires a rough (instead of a complete and exact) example from users for extraction rule generation, therefore it is semi-supervised. IEPAD, although requires no labeled training pages, post-effort from the user is required to choose the target pattern and indicate the data to be extracted. Both systems are targeted for record-level extraction tasks. Since no extraction targets are specified for such systems, a GUI is required for users to specify the extraction targets after the learning phase. Thus, users' supervision is still involved.

IEPAD: The first IE system that generalizes extraction patterns from unlabeled Web pages [Chang, 2001]. This method exploits the fact that relevant data are often generated based on some predefined templates (patterns), and applied a repeat mining technique to discover such patterns. Therefore, learning wrappers can be solved by discovering repetitive patterns. Since several repetitive patterns can be discovered in a Web page, a GUI is designed to let the user select the target pattern and decide what data values to be extracted.

OLERA: A semi-supervised IE system that acquires a rough example from the user for extraction rule generation [Chang, 2004]. OLERA can learn extraction rules for pages containing single-records, a situation where IEPAD fails. OLERA consists of 3 main operations. (1) *Enclosing an information block of interest*: where the user marks an information block containing a record to be extracted, then OLERA discovers other similar blocks (using *approximate matching* technique) and generalizes them to an extraction pattern (using *multiple string alignment* technique). (2) *Drilling-down/rolling-up an information slot*: drilling-down allows the user to navigate from a text fragment to more

detailed components, whereas rolling-up combines several slots to form a meaningful information unit. (3) *Designating relevant information slots*: the result of the above operations is presented by a spreadsheet with multiple slots decomposed from the enclosed information block.

4.4 Un-Supervised IE systems

As shown at the left-top of Figure 1, unsupervised IE systems do not use any labeled training examples and have no user interactions to generate a wrapper. Unsupervised IE systems, RoadRunner and EXALG, are designed to solve page-level extraction task. In contrast to supervised IE systems where the extraction targets are specified by the users, the extraction target is defined as the data that is used to generate the page. In some cases, several schemas may comply with the training pages due to the presence of nullable data attributes, leading to ambiguity [Yang, 2003]. If not all data is needed, post-processing may be required for the user to select relevant data and give each piece of data a proper name.

DeLa: A system that automatically assign labels to the extracted data from a semi-structured Web sites [Wang, 2002; 2003], as an extension of IEPAD. The main component of this system is a *Wrapper generator*, which has three main tasks: a *data-rich section extractor* to extract data-rich sections from the Web pages (using the DSE algorithm), a *pattern extractor* to build token suffix-trees for the token sequences and extract continuous repeated (C-repeated) patterns with nested structure in the suffix trees, and a *wrapper generator* to generalize a final wrapper for the Web pages using a string alignment algorithm.

RoadRunner: An automatic wrapper generation system [Crescenzi, 2001]. It considers the site generation process as an encoding of the original database content into strings of HTML code; as a consequence, data extraction is considered as a decoding process. Therefore, generating a wrapper for a set of HTML pages corresponds to inferring a grammar for the HTML code. The system uses the *ACME matching technique* to compare HTML pages of the same class and generate a wrapper based on their similarities and differences. Data extraction process was accomplished by 3 modules. The first module, *Classifier*, analyzes pages and collects them into clusters with a homogeneous structure. The second module, *Expander*, infers a wrapper for the classes of pages. The final module, *Labeler*, discovers attribute names.

EXALG: Arasu and Molina presented an effective formulation for the problem of data extraction from Web pages [Arasu, 2003]. The input of EXALG is a set of pages created from the unknown template T and the values to be extracted. EXALG deduces the template T and uses it to extract the set of values from the encoded pages as an output. This method defines a template T as a set of patterns, where each pattern represents an ordered set of strings which include words and HTML tags. EXALG detects the unknown template by using the two techniques *equivalence classes* and *differentiating roles*, which are based both on the number of occurrences of each token in the input pages and the different roles of the occurrences of a token in different contexts.

Although unsupervised approaches use annotation-free training page for extraction rule generation or schema inference. However, there are not fully-automatic.

5. A Comparative Analysis of IE Tools

Although many researchers have developed various tools for data extraction from Web pages, there has been only a limited amount of effort to compare such tools. Unfortunately, in only a few cases can results generated by distinct tools be directly comparable. From our viewpoint, even in these few cases, the main goal of the comparison is for a survey. Therefore, in this section, we use the criteria of the 3 dimensions suggested in Section 3 to compare the surveyed IE tools.

5.1 Task Domain-based comparison

To determine why an IE system fails to handle some Web sites with a particular structure, we have suggested several features in section 3.1 which represent the difficulties that may exist in a Web site. In this section, we contrast among the capabilities of the surveyed IE systems to support these features as shown in Table 1. The results are discussed as follows.

Page Type (PT): As discussed above, Web pages may be *structured*, *semi-structured* or *free-text* Web pages. The extraction rules for the three systems SRV, WHISK and RAPIER use a combination of both delimiter and syntactic/semantic constraints, so they can handle free-text Web pages. All other systems in Table 1 are developed to handle semi-structured Web pages.

Non-HTML Support (NHS): Some wrappers, e.g. WebOQL, W4F, XWrap, RoadRunner and DeLa, rely purely on the use of HTML tags, so they cannot support non-HTML documents. Others, e.g. supervised systems, can support non-HTML documents since their extraction rules also include other kinds of marks and information. The extraction rules in Minerva and TSIMMIS are written by hand and can be adapted by the wrapper developer to handle non-HTML documents. Finally, IEPAD and OLERA can be easily adapted to handle non-HTML documents by adding new encoding schemes.

Singleton Pages (SP): Singleton pages are those whose layout is different from all other pages in the site. Extracting of relevant data from singleton Web pages requires a specific type of wrapper which may be different from that used to extract relevant data from template-based Web pages. All of the manually-constructed and supervised systems can extract data from singleton pages. IEPAD fails to handle singleton pages that contain a single record. Unsupervised systems can not handle singleton pages because they are based on template to generate their wrappers, except for RoadRunner which contains a special wrapper "*Expander*" for handling singleton pages.

Extraction Level (EL): IE tasks can be classified into the four categories: field-level, record-level, page-level and site-level. Rapier and SRV are designed to extract single-slot records, or equivalently field-level extractions. Wrappers in EXALG and RoadRunner extract attributes of multiple kinds of records, so wrappers in these systems are page-level. The other remaining systems in Table 1 discover record boundaries and then divide them into separate items, so they are examples of record-level IE tasks. The bottom-up extraction strategy in DEByE extracts a set of attributes and then assembles them to form a record, so we consider it a record-level task.

Nested Data-Object (Nested): Many Web pages are hierarchically organized with multiple nesting levels. Typically, this complex structure is loose, presenting degrees of variations on semi-structured data. A good IE system is one that can handle data-objects of high complexity and nested structure. As shown in Table 1, programming-based IE systems, unsupervised IE systems and some supervised IE systems, such as DEByE, NoDoSE and Stalker, all support the handling of data objects with nested structure. Programming-based IE systems support the feature with the design of exception handlers, for example, the *fork* operators in W4F support the feature. SoftMealy handles objects of nested structures through multi-pass FST. Semi-supervised IE systems, IEPAD and OLERA, support nested data object extraction through multiple encoding schemes. Finally, single-slot extraction tasks do not deal with the relationships of attributes in the data objects (either simple or nested).

Missing Attributes (MA): Missing attributes may cause the exception of execution for multi-slot extraction task. Therefore, it requires special care, such as the exception handler in Minerva and the "case" operator in TSIMMIS. For single-slot extraction tasks, SRV and Rapier, handling missing attributes does not present specific difficulties. Most contemporary IE tools support missing attributes except for WIEN. The systems IEPAD, OLERA, RoadRunner and DeLa make use of alignment technique to handle missing valued attributes. SoftMealy handles missing attributes through skipped states of the FST.

Multi-Valued Attributes (MVA): Another difficulty of execution for multi-slot extraction tasks is the extraction of multi-valued attributes. The exception handler in Minerva, the "split" operator in TSIMMIS, the matching technique in RoadRunner and the alignment technique in IEPAD, OLERA and DeLa, all provide different ways of handling multi-valued attributes. SoftMealy handles multi-valued attributes through the cycles between states. Most contemporary IE tools support multiple valued attributes, except for WIEN. Multi-valued attributes also cause little difficulty (usually a loop) for single-slot extractions tools.

Multi-Ordering Attributes (MOA): Attributes of multiple ordering may cause the biggest exception of execution for multi-slot extraction task. Hsu was a pioneer who attempted to overcome the problem of multiple ordering attributes. However, from our viewpoint the situations he handled were instances of missing attributes. So, we consider that SoftMealy is limited to handle multi-ordering attributes using single-pass FST. Also, the alignment technique in IEPAD, OLERA and DeLa is limited to handle multi-ordering attributes. Stalker can handle multi-ordering attributes by multi-pass scans over the input data. The single-slot extraction tasks can handle attributes of multiple ordering since they search for the fragments (no constraints on the position) to fill in the field slot. Also, the exception handler in Minerva, the HEL language in W4F, the query language in WebOQL, the combination of mining technique in NoDoSE and the bottom-up strategy (where the set of attributes are recognized, extracted and stored in a set variable prior to the object itself) in DEByE gives these systems the ability to handle multi-ordering attributes. Finally, TSIMMIS, XWrap, WIEN, RoadRunner and EXALG can not handle multi-ordering attributes because their extraction rules depend on the location of the fields within a record.

Fixed/Variant-Format Attributes (FVF): Variant-format attributes usually require disjunctive rule supports, and WHISK, RAPIER and SRV can handle disjunctive attributes because each one can impose a set of constraints. The extraction rules in STALKER, SoftMealy and EXALG also support disjunctive attributes, while WIEN, W4F, XWrap, NoDoSE, DEByE and RoadRunner do not support disjunctive rules.

UnTokenized Attributes (UTA): Most IE systems that use delimiters as extraction patterns do not support the extraction of untokenized attributes, except for the following. Untokenized attributes can be handled by the exception handler in Minerva, the specification file in TSIMMIS, the lower level of encoding scheme in IEPAD and OLERA. STALKER can handle untokenized attributes if it was defined by the EC formalism that describes the structure of a page. SoftMealy can handle Web pages with untokenized attributes because it uses contextual rules instead of delimiters. Finally, the constraints in the extraction rules for WHISK, RAPIER, and SRV can be tuned to handle untokenized attributes.

Sequential-Pattern Attributes (SPA): Some IE systems deal with the delimiter of an attribute as a single structure, others deal with the delimiter as it consists of a set of sequential delimiters. The advantages of dealing with the delimiter as sequential delimiters arise in the process of rule generalization, where these sequence of delimiters are interruptible in-between to avoid situations when no direct rules can be generalized. The systems TSIMMIS, WebOQL, RAPIER, WHISK, SRV, WIEN, NoDoSE and DEByE all consider the delimiter as if it consists of a single structure, whereas the other systems in Table 1 consider it as a set of sequential delimiters.

UnDistinguishable Attributes (UDA): Sometimes, the different attributes in a data-object that may have the same display format. We believe that new features relating to the extracted values themselves such as **digit density** (fraction of numeric characters); **upper-case density** (fraction of upper-case letters); **length** (number of characters) can be used to handle undistinguishable-pattern attributes. As shown in Table 1, the only systems that may handle such attributes are SRV, WHISK and RAPIER, all of which use semantic features for data extraction from free text documents.

Table 1: Analysis based on the task domains

Tools		PT	NHS	SP	EL	Nested	MA	MVA	MOA	FVF	UTA	SPA	UDA
Manual	Minerva	Semi-S	Yes	Yes	Record Level	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	TSIMMIS	Semi-S	Yes	Yes	Record Level	Yes	Yes	Yes	No	Yes	Yes	No	No
	WebOQL	Semi-S	No	Yes	Record Level	Yes	Yes	Yes	Yes	Yes	No	No	No
	W4F	Semi-S	No	Yes	Record Level	Yes	Yes	Yes	Yes	No	No	Yes	No
	XWRAP	Semi-S	No	Yes	Record Level	Yes	Yes	Yes	No	No	No	Yes	No
Supervised	RAPIER	Free	Yes	Yes	Field Level	No	Yes	Yes	Yes	Yes	Yes	No	Yes
	SRV	Free	Yes	Yes	Field Level	No	Yes	Yes	Yes	Yes	Yes	No	Yes
	WHISK	Free	Yes	Yes	Record Level	No	Yes	Yes	Yes	Yes	Yes	No	Yes
	NoDoSE	Semi-S	Yes	Yes	Record Level	Yes	Yes	Yes	Yes	No	No	No	No
	DEByE	Semi-S	Yes	Yes	Record Level	Yes	Yes	Yes	Yes	No	No	No	No
	WIEN	Semi-S	Yes	Yes	Record Level	No	No	No	No	No	No	No	No
	STALKER	Semi-S	Yes	Yes	Record Level	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	SoftMealy	Semi-S	Yes	Yes	Record Level	Multi Pass	Yes	Yes	Limited	Yes	Yes	Yes	No
Semi-Supervised	IEPAD	Semi-S	No	Limited	Record Level	Limited	Yes	Yes	Limited	Yes	Yes	Yes	No
	OLERA	Semi-S	No	Limited	Record Level	Limited	Yes	Yes	Limited	Yes	Yes	Yes	No
Un-Supervised	RoadRunner	Semi-S	No	Yes	Page Level	Yes	Yes	Yes	No	No	No	Yes	No
	EXALG	Semi-S	Yes	No	Page Level	Yes	Yes	Yes	No	Yes	No	Yes	No
	DeLa	Semi-S	No	No	Record Level	Yes	Yes	Yes	Limited	Yes	No	Yes	No

5.2 Automation degree-based comparison

In this section, we use the features suggested in Section 3.2 to compare and evaluate IE systems from the automation degree prospective. The results are shown in Table 2 and discussed below.

GUI Support: Minerva, TSIMMIS and WebOQL do not use any GUI for data extraction, and the overall process was accomplished manually. W4F and XWrap provide user-friendly interfaces that assist the user during the various stages of wrapper construction. For example, the user can use the extraction wizard component of W4F to write the extraction rules. The supervised systems WIEN, Stalker, SoftMealy, Rapier, Whisk and SRV use a GUI throughout the labeling process to prepare the training examples. DEByE and NoDoSE use a GUI to aid the user to decompose the input training document into a hierarchy that describes its structure. IEPAD provides a GUI called *pattern viewer*, for the user to select the target patterns from the discovered patterns. OLERA provides a GUI with three operations: enclosing, drill-down/roll-up and attribute naming for the user to replace the process of labeling the input data. The three unsupervised systems do not need to use any GUI to induce wrappers, but EXALG only proposes to build a GUI to aid in data extraction post-processing.

Crawler Support: W4F has a component called *RetrieveAgent* that is used to retrieve a Web source by inputting its URL. Also, the *syntactical normalizer* component of XWrap accepts an URL entered by the user, issues an HTTP request to the remote server identified by the URL and fetches the corresponding Web page. The supervised and the semi-supervised IE systems as well as EXALG and RoadRunner use a set of pages that are manually downloaded as training examples. DeLa uses the existing Hidden Web crawler, *HiWe*, to automatically collect the labels of the elements from Web sites and send queries to the Web site; it does not automatically locate a Web site from which users want to extract data objects.

Output Support: Outputting the extracted relevant data in a general text format is comparably simple, so most IE systems support it. On the other hand, supporting the output in XML or SQL database format needs special care. The systems Minerva, W4F, XWrap, NoDoSE, DEByE, SoftMealy, OLERA and RoadRunner output the extracted data in a XML format. Also, NoDoSE supports other formats, such as OEM, and DEByE supports SQL database output format.

Training Examples: Wrappers in Minerva, TSIMMIS and WebOQL were written by hand and do not need any training examples, but the wrapper developers have to carefully study the structure of the Web sites that the wrappers were developed for. The supervised systems and the two systems W4F and XWrap use a set of labeling training examples (labeled by the wrapper developers) to generate their wrappers. Also, Kushmerick tried to make the labeling process automatically in his system WIEN, although its heuristic-based algorithm is impractical [Kushmerick, 1997]. Finally, the training examples input into semi-supervised and unsupervised systems are unlabeled.

API Support: Programming-based IE systems have API supports, while others do not specifically mention this in their papers.

Table 2: Analysis Based on Automation Degree.

Tools	GUI support	Crawler support	Output Support	Training Examples	API. Support
Minerva	No	No	XML	No	Yes
TSIMMIS	No	No	Text	No	Yes
WebOQL	No	No	Text	No	Yes
W4F	Yes	Yes	XML	Labeled	Yes
XWRAP	Yes	Yes	XML	Labeled	Yes
RAPIER	No	No	Text	Labeled	No
SRV	No	No	Text	Labeled	No
WHISK	No	No	Text	Labeled	No
NoDoSE	Yes	No	XML, OEM	Labeled	Yes
DEByE	Yes	Yes	XML, SQL DB	Labeled	Yes
WIEN	Yes	No	Text	Labeled	Yes
STALKER	Yes	No	Text	Labeled	Yes
SoftMealy	Yes	Yes	XML, SQL DB	Labeled	Yes
IEPAD	Yes	No	Text	Unlabeled	No
OLERA	Yes	No	XML	Unlabeled	No
RoadRunner	No	Yes	XML	Unlabeled	Yes
EXALG	No	No	Text	Unlabeled	No
DeLa	No	Yes	Text	Unlabeled	Yes

5.3 Technique-based comparison

In this section, we use the criteria suggested in Section 3.3 to compare and evaluate IE systems from the perspective of the underlying techniques used. The results are shown in Table 3 and discussed below.

Scan Pass: This comparison refers to the number of scan passes required for the generated extractors. Most WI systems design the extractor to scan the input document once, referred to as single-pass extractor, while others (e.g. STALKER and multi-pass SoftMealy) scan the input document several times to complete the extraction. Generally speaking, single-pass wrappers are more efficient than multi-pass wrappers. However, multi-pass wrappers are more effective at handling data objects with unrestricted attribute permutations or complex object extraction. SRV and Rapier can only generate single slot rules, so the extractor needs to make multiple passes over the input page to extract relevant data.

Extraction Rule Type: Most WI systems use extraction rules that are represented as regular grammars to identify the beginning and end of the relevant data, whereas Rapier and SRV use extraction rules expressed using first order logic.

Features Used: Earlier IE systems are designed to handle non-template based Web pages, say computer science department Web pages from various universities. Therefore, they have used HTML tags and literal words as delimiter-based constraints. For template-based Web pages, it is possible to use DOM tree paths to denote a specific piece of information in a Web page. For example, W4F, XWrap and other commercial products use DOM tree paths to address a Web page. Since the data to be extracted are often co-located in the same path of the DOM tree, this makes the rule learning process much easier. For free text information extraction, natural language processing techniques such as part-of-speech tagger and Word-Net semantic classes are used as additional features. SRV also uses orthographic features, token's length, and link grammars. Finally, EXALG exploits statistical information of the tokens in Web pages to generate their wrappers.

Learning Algorithm: Wrappers in programming-based WI systems are written by hand and take as input a specification that is declaratively stated where the data of interest is located in the HTML pages and how the data is packaged into objects. Thus, no learning algorithms are used in these systems. Rapier is a bottom-up relational learning system inspired by ILP methods, while SRV is a top-down relational algorithm. Whisk is a top-down covering learning system. Its patterns have two components that specify the *context* and the exact *delimiters* of the phrase to be extracted. WIEN, Stalker and SoftMealy are ML techniques which generate extraction rules derived from a given set of labeled training examples. Semi-supervised or unsupervised IE systems mainly apply data mining techniques for various pattern discoveries. IEPAD discovers regular and adjacent maximum patterns using PAT trees and string alignment techniques, while DeLa discovers continuous repeated (C-repeated) patterns. OLERA applies approximate string matching and string alignment techniques following the users' enclosing, drill-down/roll-up operations. DEByE and NoDoSE all require a large amount of support from users to model the data in the documents. EXALG exploits statistical information to generate the template and schema of Web pages by using *equivalence classes* and *differentiating roles* techniques. Finally, RoadRunner analyzes HTML structure by simultaneously comparing two HTML pages of the same class and generating a wrapper based on their similarities and differences by applying the ACME (Align, Collapse under Mismatch and Extract) technique.

Tokenization Schemes: Wrappers in Minerva and TSIMMIS are written by hand, so they do not need to tokenize the input pages. Most WI systems for Web pages support tag-level tokernization. Some systems even support word-level tokernization, e.g. supervised WI systems and EXALG. WebOQL, W4F, XWrap, RoadRunner and DeLa use a tag-level encoding scheme to translate the input training pages into tokens. Also, the input HTML page in W4F and XWrap has

been parsed to construct a parse tree that reflects its HTML tags hierarchy following the document object model (DOM). Finally, IEPAD and OLERA allow multiple levels of encodings for input training pages.

Table 3: Analysis based on the techniques used.

Tools	Scan Pass	Extraction Rule Type	Features Used	Learning Algorithm	Tokenization Schemes
Minerva	Single	Regular exp.	HTML tags/Literal words	None	Manually
TSIMMIS	Single	Regular exp.	HTML tags/Literal words	None	Manually
WebOQL	Single	Regular exp.	Hypertree	None	Manually
W4F	Single	Regular exp.	DOM tree path addressing	None	Tag Level
XWRAP	Single	Context-Free	DOM tree	None	Tag Level
RAPIER	Multiple	Logic rules	Syntactic/Semantic	ILP (bottom-up)	Word Level
SRV	Multiple	Logic rules	Syntactic/Semantic	ILP (top-down)	Word Level
WHISK	Single	Regular exp.	Syntactic/Semantic	Set covering (top-down)	Word Level
NoDoSE	Single	Regular exp.	HTML tags/Literal words	Data Modeling	Word Level
DEByE	Single	Regular exp.	HTML tags/Literal words	Data Modeling	Word Level
WIEN	Single	Regular exp.	HTML tags/Literal words	Ad-hoc (bottom-up)	Word Level
STALKER	Multiple	Regular exp.	HTML tags/Literal words	Ad-hoc (bottom-up)	Word Level
SoftMealy	Both	Regular exp.	HTML tags/Literal words	Ad-hoc (bottom-up)	Word Level
IEPAD	Single	Regular exp.	HTML tags	Pattern Mining, String Alignment	Multi-Level
OLERA	Single	Regular exp.	HTML tags	String Alignment	Multi-Level
RoadRunner	Single	Regular exp.	HTML tags	String Alignment	Tag Level
EXALG	Single	Regular exp.	HTML tags/Literal words	Equivalent Class and Role Differentiation	Word Level
DeLa	Single	Regular exp.	HTML tags	Pattern Mining	Tag Level

6. Conclusions

In this paper, we survey the major IE tools in the literature and compare them in three dimensions: the task domain, the automation degree, and the techniques used. A set of criteria are proposed for the comparison and evaluation in each dimension. The criteria of the first dimension explain why an IE system fails to handle some Web sites of particular structures. The criteria of the second dimension measure the degree of automation for IE systems. The criteria of the third dimension measure the performance of IE systems. We present our taxonomy of WI systems from the users' viewpoint and compare important features of WI systems that affect their effectiveness. We focus on semi-structured documents that are usually produced with templates by a server-side Web application. The extension to non-HTML can be achieved for some WI systems if a proper tokenization or feature set is used. The most critical issue for these state-of-the-arts WI systems is that the generated extraction rules apply to only a small collection of documents that share the same layout structure. Usually those rules only apply to documents from one Web site. Generalization across different layout structures is still an extremely difficult and challenging research problem, and it is difficult to resolve this problem since information integration and data mining to the scale of billions of Web sites is still too expensive and impractical.

References

Adelberg, B., NoDoSE: A Tool for Semi-Automatically Extracting Structured and Semi-Structured Data from Text Documents. SIGMOD Record 27(2): 283-294, 1998.

Arasu, A. and Garcia-Molina, H., Extracting Structured Data from Web Pages. Proceedings of the ACM SIGMOD International Conference on Management of Data, San Diego, California, pp. 337-348, 2003.

Arocena, G. O. and Mendelzon, A. O., WebOQL: Restructuring Documents, Databases, and Webs. Proceedings of the 14th IEEE International Conference on Data Engineering (ICDE), Orlando, Florida, pp. 24-33, 1998.

Califf, M. and Mooney, R., Relational learning of pattern-match rules for information extraction. Proceedings of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing Stanford, California, March, 1998.

Chang, C.-H. and Lui, S.-C., IEPAD: Information Extraction Based on Pattern Discovery. Proceedings of the Tenth International Conference on World Wide Web (WWW), Hong-Kong, pp. 223-231, 2001.

Chang, C.-H., Hsu, C.-N., and Lui, S.-C. Automatic Information Extraction from Semi-Structured Web Pages by Pattern Discovery. Decision Support Systems Journal, 35(1): 129-147, 2003.

Chang, C.-H. and Kuo, S.-C. OLERA: A Semi-supervised Approach for Web Data Extraction with Visual Support. IEEE Intelligent Systems, 2004 (To appear).

Ciravegna, F., Learning to Tag for Information Extraction from Text. Proceedings of the ECAI-2000 Workshop on Machine Learning for Information Extraction, Berlin, August 2000.

Crescenzi, V., and Mecca, G., Grammars Have Exceptions. Information Systems, 23(8): 539-565, 1998.

Crescenzi, V., Mecca, G. and Merialdo, P., RoadRunner: Towards-Automatic Data Extraction from Large Web Sites. Proceedings of the 26th International Conference on Very Large Database Systems (VLDB), Rome, Italy, pp. 109-118, 2001.

Embley, D. W., Campbell, D. M., Jiang, Y. S., Liddle, S. W., Kai Ng, Y., Quass, D. and Smith, R. D., Conceptual-Model-Based Data Extraction from Multiple-Record Web Pages. Data and Knowledge Engineering, 31(3): 227-251, 1999.

Freitag, D., Information extraction from HTML: Application of a general learning approach. Proceedings of the Fifteenth Conference on Artificial Intelligence (AAAI-98).

Hammer, J., McHugh, J. and Garcia-Molina, Semistructured Data: The TSIMMIS Experience. In Proceedings of the 1st East-European Symposium on Advances in Databases and Information Systems (ADBIS), St. Petersburg, Rusia, pp. 1-8, 1997.

Hsu, C. and Dung, M., Generating finite-state transducers for semi-structured data extraction from the web. Journal of Information Systems 23(8): 521-538, 1998.

Huffman, S., Learning information extraction patterns from examples. Connectionist, statistical, and symbolic Approaches to Learning for Natural Language Processing, Springer-Verlag, 1996.

Kim, J. and Moldovan, D., Acquisition of linguistic patterns for knowledge-based information extraction. IEEE Transactions on Knowledge and Data Engineering 7(5): 713-724, 1995.

Kuhlins, S and Tredwell, R. Toolkits for generating wrappers, Net.ObjectDays 2002: Objects, Components, Architectures, Services and Ap-plications for a Networked World, http://www.netobjectdays.org/, LNCS 2591, 2002.

Liu, L., Pu, C. and Han, W., XWrap: An XML-enable Wrapper Construction System for Web Information Sources. Proceedings of the 16th IEEE International Conference on Data Engineering (ICDE), San Diego, California, pp. 611-621, 2000.

Knoblock, C. A., Minton, S., Ambite, J. L., Ashish, N., Muslea, I., Philpot, A. G., and Tejada, S., The ARIADNE approach to Web-based information integration. International Journal of Cooperative Information Systems (IJCIS), Special Issue on Intelligent Information Agents: Theory and Applications, 10(1/2):145-169, 2001.

Krupka, G., Description of the SRA system as used for MUC-6. Proceedings of the sixth Message Understanding Conference (MUC-6), pp. 221-235, 1995.

Kushmerick, N., Weld, D., and Doorenbos, R., Wrapper induction for information extraction. Proceedings of the Fifteenth International Conference on Artificial Intelligence (IJCAI), pp. 729-735, 1997.

Kushmerick. N., Adaptive Information Extraction: Core technologies for Information agents. In *Intelligent Information Agents R&D in Europe: An AgentLink perspective* (Klusch, Bergamaschi, Edwards & Petta, eds.). Lecture Notes in Computer Science 2586, Springer, 2003.

Laender, A. H. F., Ribeiro-Neto, B., DA Silva and Teixeira, A Brief Survey of Web Data Extraction Tools. SIGMOD Record 31(2): 84-93, 2002.

Laender, A. H. F., Ribeiro-Neto, B. and DA Silva, A., S., DEByE - Data Extraction by Example. Data and Knowledge Engineering, 40(2): 121-154, 2002.

Liu, L., Pu, C., and Han, W. XWRAP: An XML-Enabled Wrapper Construction System for Web Information Sources, Proceedings of the 16th IEEE International Conference on Data Engineering (ICDE), San Diego, California, pp. 611-621, 2000.

Muslea, I., Minton, S., and Knoblock, C., A hierarchical approach to wrapper induction. Proceedings of the Third International Conference on Autonomous Agents (AA-99), 1999.

Ribeiro-Neto, B., A., Laender, A., H., F. and DA Silva, A., S., Extracting Semi-Structured Data Through Examples. Proceedings of the Eighth ACM International Conference on Information and Knowledge Management (CIKM), Kansas City, Missouri, pp. 94-101, 1999.

Riloff, E., Automatically Constructing a Dictionary for Information Extraction Tasks. Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93), pp. 811-816, AAAI Press/The MIT Press, 1993.

Saiiuguet, A. and Azavant, F., Building Intelligent Web Applications Using Lightweight Wrappers. Data and Knowledge Engineering 36(3): 283-316, 2001.

Sarawagi, S., Automation in Information Extraction and Integration**,** Tutorial of The 28th International Conference on Very Large Data Bases (VLDB), 2002.

Soderland, S., Fisher, D., Aseltine, J., and Lehnert, W., CRYSTAL: Inducing a conceptual dictionary. Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI), 1995.

Soderland, S., Learning to extract text-based information from the world wide web. Proceedings of the third International Conference on Knowledge Discovery and Data Mining (KDD), pp. 251-254, 1997.

Soderland, S., Learning information extraction rules for semi-structured and free text. Journal of Machine Learning, 34(1-3): 233-272, 1999.

Wang, J. and Lochovsky, F. H., Wrapper Induction Based on Nested Pattern Discovery. Technical Report HKUST-CS-27-02, Dept. of Computer Science, Hong Kong, U. of Science & Technology, 2002.

Wang, J. and Lochovsky, F. H., Data Extraction and Label Assignment for Web Databases, Proceedings of the Twelfth International Conference on World Wide Web (WWW), Budapest, Hungary, pp. 187-196, 2003.

Yang, G., Ramakrishnan, I. V. and Kifer, M. On the Complexity of Schema Inference from Web Pages in the Presence of Nullable Data Attributes, Proceedings of the 12th ACM International Conference on Information and Knowledge Management (CIKM), pp. 224-231, 2003.