Adaptive Information Extraction

JORDI TURMO, ALICIA AGENO, NEUS CATALÀ
TALP Research Center, Universitat Politècnica de Catalunya, Spain

Abstract

The growing availability of on-line textual sources and the potential number of applications of knowledge acquisition from textual data has lead to an increase in Information Extraction (IE) research. Some examples of these applications are the generation of data bases from documents, as well as the acquisition of knowledge useful for emerging technologies like question answering, information integration, and others related to text mining. However, one of the main drawbacks of the application of IE refers to its intrinsic domain dependence. For the sake of reducing the high cost of manually adapting IE applications to new domains, experiments with different Machine Learning (ML) techniques have been carried out by the research community. This survey describes and compares the main approaches to IE and the different ML techniques used to achieve Adaptive IE technology.

1 Introduction

Traditionally, information involved in Knowledge-Based systems has been manually acquired in collaboration with domain experts. However, both the high cost of such a process and the existence of textual sources containing the required information have led to the use of automatic acquisition approaches. In the early eighties, Text-Based Intelligent (TBI) systems began to manipulate text so as to automatically obtain relevant information in a fast, effective and helpful manner Jacobs [1992]. Texts are usually highly structured when produced to be used by a computer, and the process of extracting information from them can be carried out in a straightforward manner. However, texts produced to be used by a person lack an explicit structure. Generally, they consist of unrestricted Natural Language (NL) text, and the task of extracting information involves a great deal of linguistic knowledge. Between these ends falls semi-structured text, such as on-line documents, where both chunks of NL text and structured pieces of information (e.g., meta-data) appear together.

Roughly speaking, two major areas of TBI can be distinguished: Information Retrieval (IR) and Information Extraction (IE). IR techniques are used to select those documents from a collection that most closely conform to the restrictions of a query, commonly a list of keywords. As a consequence, IR techniques allow recovering relevant documents in response to the query. The role of Natural Language Processing (NLP) techniques in IR tasks is controversial and generally considered marginal. The reader may find more detailed account of IR techniques (c.f., Grefenstette [1998]; Strzalkowski [1999]; Baeza-Yates and Ribeiro-Neto [1999]).

IE technology involves a more in-depth understanding task. While in IR the answer to a query is simply a list of potentially relevant documents, in IE the relevant content of such documents has to be located and extracted from the text. This relevant content, represented in a specific format, can be integrated within knowledge-based systems, as well as used within IR in order to obtain more accurate responses. Some emerging technologies, such as Question Answering and Summarization, attempt to derive benefit from both IR and IE techniques (c.f., Pasca [2003]; Radev [2004]).

In order to deal with the difficulty of IE, NLP is no longer limited to splitting text into terms, as it generally occurs in IR, but is more intensively used throughout the extraction process, depending on the document style to be dealt with. Statistical methods, although present in many of the NL components of IE systems, are not sufficient to approach many of the tasks involved, and have to be combined with knowledge-based approaches. In addition, one of the requirements of IE is that the type of content to be extracted must be defined *a priori*. This implies domain dependence of the IE technology, which leads to portability drawbacks that are present in most IE systems. When dealing with new domains, new specific knowledge is needed and has to be acquired by such systems. In order to address these problems of portability and knowledge acquisition, Adaptive IE technology focuses on the use of empirical methods in NLP to aid the development of IE systems.

This paper is organized as follows. Section 2 briefly describes the IE problem. Section 3 describes the historical framework in which IE systems have been developed. Within this framework, the general architecture of IE systems is described in Section 4. The complexity of IE systems and their intrinsic domain dependence make it difficult for them to be accurately applied to any situation (i.e., different domains, author styles, document structures, etc.). Thus, Section 5 is devoted to the use of Machine Learning (ML) techniques for Adaptive Information Extraction. A classification of different state-of-the-art IE systems is presented from two different perspectives in Section 6, together with a more thorough description of three of these systems. Finally, Section 7 presents the conclusions of this survey.

2 The goal of Information Extraction

The objective of IE is to extract certain pieces of information from text that are related to a prescribed set of related concepts, namely, an *extraction scenario*. As an example, let us consider the scenario of extraction related to the domain of Management Succession[1]:

> This scenario concerns events that are related to changes in company management. An article may describe one or more management succession events. The target information for each succession event is the person moving into a new position (**PersonIn**), the person leaving the position (**PersonOut**), the title of the position (**Post**) and the corporation name (**Org**). The other facts appearing in the article must be ignored.

The following is an excerpt of a document from the Management Succession domain:

[1]The concepts to be dealt with are written in bold.

A. C. Nielsen Co. said George Garrick, 40 years old, president of Information Resources Inc.'s London-based European Information Services operation, will become president of Nielsen Marketing Research USA, a unit of Dun & Bradstreet Corp. He succeeds John I. Costello, who resigned in March.

An IE system should be able to recognize the following chunks, among others, as relevant information for the previous succession event: *A. C. Nielsen Co.*, *George Garrick*, *president of Information Resources Inc.*, *Nielsen Marketing Research*, *succeeds John I. Costello.* Moreover, the system should recognize the fact that all this information is related to the same event. The output of the extraction process would be the template like that shown in Figure 1. Other succession events may involve merging information across sentences and detecting pronominal coreference links.

```
<Succession_event_1> =
  PersonIn:       George Garrick
  PersonOut:      John I. Costello
  Post:           president
  Org:            Nielsen Marketing Research
```

Figure 1: Example of an output template extracted by an IE system.

The above example was extracted from free text but there are other text styles to which IE can be applied, namely, structured and semi-structured text. Structured text is easily seen on Web pages where information is expressed by using a rigid format; for example, CNN weather forecast pages. Semi-structured text often present fragments of sentences and the information in them is expressed following some order. An example of semi-structured text is found in the collection of electronic seminar announcements (Seminar Announcement domain), where information about starting time (**stime**), ending time (**etime**), speaker (**speaker**) and location (**location**) must be located and annotated. Figure 2 shows a sample of formatting styles used in the seminar announcement domain. Note that not all information expressed is target information and not all target information is expressed. The output of an IE system for these two seminar announcements is shown in Figure 3.

3 Historical Framework of Information Extraction

The development of IE technology is closely bound to Message Understanding Conferences (MUC[2]), which took place from 1987 until 1998. The MUC efforts, among others, have consolidated IE as a useful technology for TBI systems. MUC conferences were started in 1987 by the US Navy (the Naval Ocean Systems Center, San Diego) and were subsequently sponsored by the United States Advanced Research Projects Agency (DARPA[3]). In 1990, DARPA launched

[2]http://www.itl.nist.gov/iaui/894.02/related_projects/muc/
[3]http://www.darpa.mil/

```
Leslie Lamport
Digital Equipment Corporation
Systems Research Center

Tuesday, April 28
3:30 pm
Wean Hall 4623
```

```
Professor John Skvoretz, U. of South Carolina, Columbia, will present
a seminar entitled ''Embedded Commitment,'' on Thursday, May 4th from
4-5:30 in PH 223D.
```

Figure 2: Two examples of seminar announcements.

```
<speaker>Leslie Lamport</speaker>
Digital Equipment Corporation
Systems Research Center

Tuesday, April 28
<stime>3:30 pm</stime>
<location>Wean Hall 4623</location>
```

```
<speaker>Professor John Skvoretz</speaker>, U. of South Carolina,
Columbia, will present a seminar entitled ''Embedded Commitment,''
on Thursday, May 4th from <stime>4</stime>-<etime>5:30</etime> in
<location>PH 223D</location>.
```

Figure 3: Output from the two seminar announcements.

the TIPSTER Text program[4] to fund the research efforts of several of the MUC participants.

The general goal of the MUC conferences was to evaluate IE systems developed by different research groups to extract information from restricted-domain free-style texts. A different domain was selected for each conference. In order to evaluate the systems, and previous to providing the set of evaluation documents to be dealt with, both a set of training documents and the scenario of extraction were provided to the participants by the MUC organization.

MUC-1 (1987). The first MUC was basically exploratory. In this first competition, neither the extraction tasks nor the evaluation criteria had been defined by organizers, although *Naval Tactical Operations* was the selected domain of the documents. Each group designed its own format to record the extracted information.

[4]http://www.fas.org/irp/program/process/tipster.html

MUC-2 (1989). For MUC-2, the same domain as for MUC-1 was used. However, on this occasion, organizers defined a task: template filling. A description of naval sightings and engagements, consisting of 10 slots (type of event, agent, time and place, effect, etc.) was given to the participants. For every event of each type, a template with the relevant information had to be filled. The evaluation of each system was done by the participants themselves. As a consequence, consistent comparisons among the competing systems were not achieved.

MUC-3 (1991). The domain of the documents was changed to *Latin American Terrorism* events. The template consisted of 18 slots (type of incident, date, location, perpetrator, target, instrument, etc.). The evaluation was significantly broader in scope than in previous MUCs. A training set of 1300 texts was given to the participants, while over 300 texts were set aside as test data. Four measures were defined over correct extracted slots (COR), incorrect extracted slots (INC), spurious extracted slots (SPUR), missing slots (MISS) and partially extracted ones (PAR). The two most relevant measures were *recall* (R) and *precision* (P), which measure the coverage and accuracy of the system, respectively. They were defined as follows:

$$R = \frac{COR + (0.5 * PAR)}{COR + PAR + INC + MISS}$$
$$P = \frac{COR + (0.5 * PAR)}{COR + PAR + INC + SPUR}$$

However, it was concluded that a single overall measure was needed for the evaluation to achieve a better global comparison among systems.

MUC-4 (1992). For MUC-4, the same task as for MUC-3 was used. However, the MUC-3 template was slightly modified and increased to 24 slots. The evaluation criteria were revised to allow global comparisons among the different competing systems. The F measure was used to identify the harmonic mean between both recall and precision:

$$F = \frac{(\beta^2 + 1.0) \cdot P \cdot R}{\beta^2 \cdot P + R} \qquad 0 < \beta \leq 1 \tag{1}$$

MUC-5 (1993). In the previous conferences, competing IE systems were only applied to extract information from documents in English. In the MUC-5 conference, documents in Japanese were also provided. Moreover, two different domains were proposed, *Joint Ventures* (JV) and *Microelectronics* (ME), which consisted of financial news and advances on microelectronics products, respectively. The central focus was on the template design, which crucially affects the success when capturing information from texts. Two different sets of object-oriented templates were defined (11 templates for JV, and 9 for ME). The evaluation was done using the same *recall-precision-based metrics* as for MUC-4. In addition, *error-based metrics* were included as well in order to classify systems by their error rates.

MUC-6 (1995). For MUC-6, the *Financial* domain was used. There were three main goals. The first goal was to identify domain independent functions on the component technologies being developed for IE. In order to meet this goal,

the organizers proposed a named entity (NE) subtask which dealt with names of persons and organizations, locations and dates, among others. The second goal was to focus on the portability of IE methodologies in order to deal with different types of events. Here, a *template element* (TE) task was proposed in order to standardize the lowest-level concepts (people, organizations, etc.), since they were involved in many different types of events. Like the NE subtask, this was also seen as a potential demonstration of the ability of systems to perform a useful, relatively domain independent task with near-term extraction technology (it required merging information from several places in the text). The old-style MUC IE task, based on a description of a particular class of event, was called *scenario template* (ST). Finally, the last goal was to encourage participants to build up the mechanisms needed for deeper understanding. Three new subtasks were proposed: *coreference resolution* (CO), *word-sense-disambiguation* and *predicate-argument syntactic structuring*. However, only the first one was evaluated. In all the evaluations partial credit was given to partially extracted slots. As a consequence, recall and precision metrics were formulated as follows:

$$R = \frac{COR}{COR + INC + MISS}$$
$$P = \frac{COR}{COR + INC + SPUR}$$

MUC-7 (1998). For MUC-7 the *Airline Crashes* domain was proposed. The difference between this and later competitions were not substantial. The NE subtask was carried out in Chinese, Japanese and English. Moreover, a new task was evaluated, which focused on the extraction of relations between TEs, as *location-of*, *employee-of* and *product-of* in the *Financial* domain. This new task was named *Template Relation* (TR).

The number of systems participating increased over the course of the MUCs but only a few of them participated in all the competitions and tasks defined. In general, the methodologies of these systems evolved into a more flexible, simple and standard approach, as will be described in Section 4, and their evaluations demonstrated the advantages of the new approaches.

The results reported by the organization for every IE task and subtask defined in each MUC show two main conclusions. On the one hand, independent of the domain, the NE subtask and TE and TR tasks achieved acceptable results compared with the results of the other goals. The best F-scores[5] to for NE, TE and TR were higher than 90%, 80% and 70% respectively. As a consequence, it seems that a more difficult set of these kinds of tasks and subtasks might be evaluated. On the other hand, the best results for CO subtasks and ST tasks were lower than 65% and 60% respectively, and consequently, it looks as if more effort should be devoted to them.

These conclusions could have conditioned the strategies of future IE evaluation frameworks, such as Automatic Content Extraction (ACE) described below.

In parallel with the MUC conferences, the European Comission funded under the LRE (Linguistic Research and Engineering) program[6] a number of projects

[5]Hereafter, we will consider the F measure defined in formula 1 with β set to 1.
[6]http://www2.echo.lu/langeng/en/lehome.html

devoted to developing tools and components for IE (also for IR), such as automatically or semi-automatically acquiring and tuning lexicons from corpora, extracting entities, parsing in a flexible and robust way, and others. These projects included ECRAN[7], SPARKLE[8], FACILE[9], and AVENTINUS[10].

Beyond the MUCs, research on IE technology has been included in the TIDES (Translingual Information Detection, Extraction and Summarization) program[11] funded by DARPA in 1999. It is an initiative on fast machine translation and information access, including translingual IE technology. Some of the sponsored projects are RIPTIDES[12], PROTEUS[13], CREST[14], Coreference.com[15], and the UMass system[16].

The primary information extraction evaluation used by the TIDES program was the ACE[17] (Automatic Content Extraction) evaluation. The ACE program was started in 1999 with the aim of developing automatic content extraction technology to extract information from human language in textual form. The reference corpus includes stories from three different sources: newswire (text), broadcast news (speech - ASR transcribed) and newspaper (image - OCR transcribed). The pilot study process dealt with Entity Detection and Tracking (EDT). This task entails the detection of mentions of entities and chaining them together by identifying their coreference. Therefore the task gets more complex with respect to MUC's by adding hierarchies of types, subtypes and classes of entities, as well as levels of mentions (names, nominal expressions or pronouns).

The ACE Phase 2 (2001 and 2002) added an evaluation on Relation Detection and Characterization (RDC) between an identified set of entities. These relations are also more complex than MUC's, with more variety of types and subtypes of relations, and the fact that they can be either explicit or implicit.

The 2003 evaluation included for the first time two other languages in addition to English, Chinese and Arabic (though the latter only for the EDT task). In 2004, the program expanded to include both Relation annotation for Arabic and a separate TIMEX2[18] task for recognizing and normalizing date and time expressions (the latter only for English and Chinese). More importantly, an Event Detection and Recognition task (VDR) was introduced. The concept of an ACE event is simpler than MUC's, being an event involving zero or more ACE entities, values and time expresions. Five different types of events were defined. However, in the end the VDR task was postponed until ACE-05 due to problems with the definition of the events. Finally, for the 2005 edition, the tasks have been the same although new corpora have been added for all three languages (conversational telephone speech and Usenet newsgroups and discussion forums for English, weblogs for all three).

[7] http://www.dcs.shef.ac.uk/intranet/research/networks/Ecran/
[8] http://www.informatics.susx.ac.uk/research/nlp/sparkle/sparkle.html
[9] http://tcc.itc.it/research/textec/projects/facile/facile.html
[10] http://www.dcs.shef.ac.uk/nlp/funded//aventinus.html
[11] http://www.darpa.mil/ipto/programs/tides/
[12] http://www.cs.cornell.edu/home/_cardie/tides/
[13] http://nlp.cs.nyu.edu/
[14] http://crl.nmsu.edu/Research/Projects/Crest/
[15] http://www.coreference.com/lingpipe/
[16] http://ciir.cs.umass.edu/research/tides.html
[17] http://www.nist.gov/speech/tests/ace/
[18] TIDES Standard for the Annotation of Temporal Expressions (http://timex2.mitre.org)

The evaluation is similar to MUC's, relative to a reference model of the application value of system output. Unfortunately, unlike the MUCs, the results and architectures of the systems presented to the ACE Program have not been as disseminated since the evaluation workshops do not publish proceedings and the rules of the program prevent the publications of global results.

IE technology has been applied to other domains in free text, different to those of MUC or ACE. Soderland Soderland et al. [1995] extracts diagnosis and signs or symptoms in the medical domain from hospital discharge reports. Holowczak and Adam Holowczak and Adam [1997] extract useful information to classify legal documents. Glasgow Glasgow et al. [1998] extracts relevant information related to the life insurance domain to support underwriters.

Other efforts focus on the application of IE technology to semi-structured documents. Within such a framework, traditional IE systems do not fit. This is because such documents are a mixture of sentences and sentences fragments, sometimes telegraphic, and include meta-tags (e.g., HTML). Examples can be found related to the apartment rental advertising domain Soderland [1999], to the biomedical domain Craven [1999], and to web pages Freitag [1998b]; Califf [1998].

Techniques very close to those of IE have been applied to highly structured documents, such as the use of specific procedures, named *wrappers*, to extract relevant information from HTML/XML tagged documents available on the Web Kushmerick [1997]; Craven et al. [1998]; Thomas [1999]; Kushmerick [2000]; Muslea et al. [2001]; Chidlovskii [2000]; Freitag and Kushmerick [2000]; Ciravegna [2001].

More recently and in parallel with the TIDES program, the European Comission has funded the Pascal Network of Excellence[19]. This organization, together with the Dot.Kom European project[20] has launched the Challenge on Evaluation of Machine Learning for Information Extraction from Documents, which involves three tasks (namely a full scenario task, an active learning task and an enriched scenario task) in the domain of a Workshop Call for Papers semi-structured texts. The first edition of this challenge started in June 2004, and the formal evaluation took place in November 2004. The main difference with respect to the MUC competitions is that the framework of comparison is machine learning (ML) oriented instead of IE oriented. In fact, the main promoters of the challenge have signed a paper (Lavelli et al. [2004]) in which they critically review the previous methodology adopted for evaluation in IE and draw the lines for a more detailed and reliable one.

The experience and results drawn from the first edition of the challenge (with 11 participants and 23 systems) acknowledge the difficult task of manually annotating the corpus. The first task (learning textual patterns to extract the annotated information) might be seen as a subset of ACE's EDT (without coreference), although with a stricter evaluation. The second task examined the effect of selecting which documents to add to the training data, and the third task focused on the effects of adding to the training data a set of 500 new

[19]Pascal is a Network of Excellence on "Pattern Analysis, Statistical Modelling and Computational Learning" funded by the European Comission as a part of Framework 6 (http://www.pascal-network.org/)

[20]Dot.Kom is a project on "Designing Adaptative Information Extraction from Text for Knowledge Management" funded by the European Comission as a part of Framework 5 (http://www.dot-kom.org)

annotated documents and using the Web as resource. The systems presented span a wide variety of learning algorithms. The best F-scores obtained for the first task (on the test data) are higher than 70%, lower than ACE 2004's on EDT which were around 80%.

4 Architecture of Information Extraction Systems

Within the historical framework, the general architecture of an IE system was defined by Hobbs Hobbs [1993] in MUC-5 as "*a cascade of transducers or modules that, at each step, add structure to the documents and, sometimes, filter relevant information, by means of applying rules*" (such as the architecture depicted in Figure 4). Most current systems follow this general architecture, although specific systems are characterized by their own set of modules, and most of the architectures are currently being developed. In general, the combination of such modules allows some of the following functionalities to a greater or lesser degree:

- Document preprocessing
- Syntactic parsing, full or partial.
- Semantic interpretation, to generate either a logical form or a partial template from a parsed sentence.
- Discourse analysis, to link related semantic interpretations among sentences. This is done using anaphora resolution, and other kinds of semantic inferences.
- Output template generation, to translate the final interpretations into the desired format.

A brief description of each functionality is presented below.

4.1 Document Preprocessing

Preprocessing the documents can be achieved by a variety of modules, such as: *text zoners* (turning a text into a set of text zones), *segmenters*, also named *splitters* (segmenting zones into appropriate units, usually sentences), *filters* (selecting the relevant segments), *tokenizers* (to obtain lexical units), *language guessers* (making guesses on the language in which the text is written), *lexical analyzers* (including morphological analysis and NE recognition and classification), engines dealing with unknown words, *disambiguators* (POS taggers, semantic taggers, etc.), *stemmers* and *lemmatizers*, etc.

Most systems take advantage of available resources and general purpose (domain independent) tools for the preprocessing step. Highly relevant for IE are the NE recognition modules. The process of NE recognition may be quite straightforwardly performed by using *finite-state transducers* and dictionary look up (domain specific dictionaries, terminological databases, etc.). In spite of this, results depend heavily on the information sources involved. Grishman Grishman [1995], for instance, in his participation in MUC-6, used the

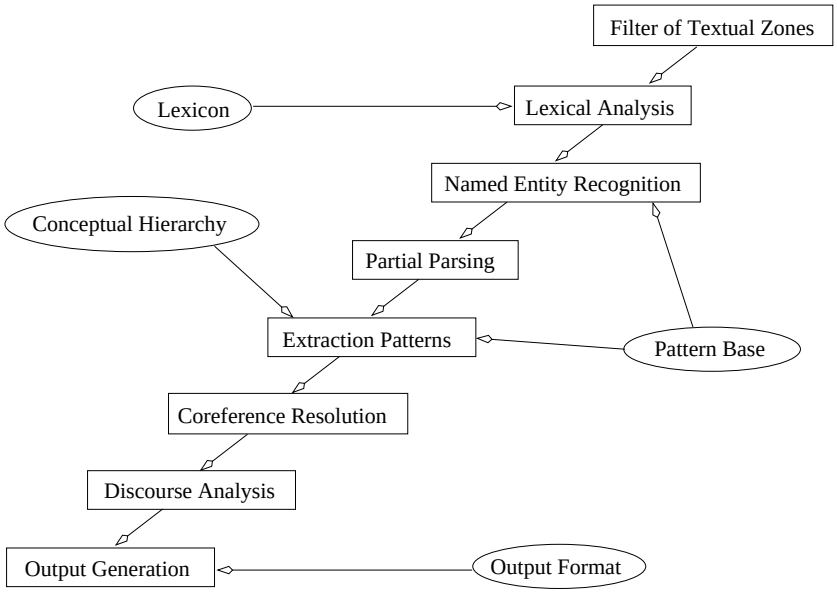

Figure 4: Usual architecture of an IE system.

following specific sources: a small gazetteer, containing the names of all countries and most major cities; a company dictionary derived from the Fortune 500; a Government Agency dictionary; a dictionary of common first names; and a small dictionary of scenario specific terms.

4.2 Syntactic Parsing and Semantic Interpretation

A more important controversy arises from parsing. At the beginning of the MUC conferences, traditional NL understanding architectures were adopted to IE. Such approaches were based on full parsing, followed by a semantic interpretation of the resulting in-depth syntactic structure, and discourse analysis. In MUC-3, however, the best scores were achieved by a simpler approach presented by Lehnert Lehnert et al. [1991], named *selective concept extraction*. Such a new approach was based on the assumption that only those concepts being within the scenario of extraction need to be detected in the documents. Consequently, syntactic and semantic analysis should be simplified by means of a more restricted, deterministic and collaborative process. Their strategy was to replace the traditional parsing, interpretation and discourse modules with a simple phrasal parser, to find local phrases, an event pattern matcher, and a template merging procedure, respectively. In MUC-4, Hobbs' group recast such an approach in terms of a more flexible model, which was based on finite-state transducers (FST) Appelt et al. [1992].

The simplification of the understanding process, presented by both Lehnert and Hobbs, has been widely adopted by the IE community. In general, this fact is due to different drawbacks on the use of full parsing Grishman [1995]:

- Full parsing involves a large and relatively unconstrained search space, and is consequently expensive.

- Full parsing is not a robust process because a global parse tree is not always achieved. In order to correct such incompleteness, the parse covering the largest substring of the sentence is attempted. Sometimes, however, this global goal leads to incorrect local choices of analyses.

- Full parsing may produce ambiguous results. More than one syntactic interpretation is usually achieved. In this situation, the most correct interpretation must be selected.

- Broad-coverage grammars, needed for full parsing, are difficult to be consistently tuned. Dealing with new domains, new specialized syntactic constructs could occur in texts and be unrecognized by broad-coverage grammars. Adding new grammar rules could produce an inconsistent final grammar.

- A full parsing system cannot manage off-vocabulary situations.

Nowadays, most existing IE systems are based on partial parsing, in which non-overlapping parse fragments, i.e. phrasal constituents, are generated. Generally, the process of finding constituents consists in using a cascade of one or more parsing steps against fragments. The resulting constituents are tagged as noun, verb, or prepositional phrases, among others. Sometimes, these components are represented as chunks of words, and the parsing process is named *chunking*[21]. However, they can also be represented as parse subtrees.

Once constituents have been parsed, systems resolve domain-specific dependencies among them, generally by using the semantic restrictions imposed by the scenario of extraction. Two different approaches are usually followed to resolve such dependencies:

- *Pattern matching.* This approach is followed by most IE systems. Syntax simplification allows reducing semantic processing to simple pattern matching, where scenario-specific patterns, also named *extraction patterns* or *IE rules*, are used to identify both modifier and argument dependencies between constituents. In fact, such IE-rules are sets of ambiguity resolution decisions to be applied during the parsing process. They can be seen as sets of syntactico-semantic expectations from the different extraction tasks. On the one hand, some IE rules allow the identification of properties of entities and relations between such entities (TE and TR extraction tasks). In general, this is done by using local syntactico-semantic information about nouns and modifiers. On the other hand, IE rules using predicate-argument relations (object, subject, modifiers) allow the identification of events among entities (ST extraction task). The representation of these IE rules greatly differs among different IE systems.

- *Grammatical relations.* Generally, the pattern-matching strategy requires a proliferation of task-specific IE rules, with explicit variants for each verbal form, explicit variants for different lexical heads, etc. Instead of using IE rules, the second method for resolving dependencies entails a more flexible syntactic model originally proposed by Vilain Vilain [1999]

[21] In fact, the formal definition of chunk is a bit more complex (c.f., Abney [1996]). A chunk is usually considered as a simple non-recursive phrasal constituent.

similar to Carroll's Carroll et al. [1998]. It consists in defining a set of grammatical relations, as general relations (subject, object and modifier), some specialized modifier relations (temporal and location) and relations that are mediated by prepositional phrases, among others. In a similar way to dependency grammars, a graph is built by following general rules of interpretation for the grammatical relations. Previously detected chunks are nodes within such a graph, while relations among them are labeled arcs.

4.3 Discourse Analysis

IE systems generally proceed by representing the information extracted from a sentence either as partially filled templates or as logical forms. Such information can be incomplete due to the occurrence of ellipsis, and sometimes, it can refer to the same entities in the presence of coreference. The main goal of the discourse analysis phase is the resolution of these aspects. Systems working with partial templates make use of some merging procedure for such a task. However, working with logical forms allows IE systems to use traditional semantic interpretation procedures in this phase.

4.4 Output Template Generation

Finally, the output template generation phase mainly aims at mapping the extracted pieces of information onto the desired output format. However, some inferences can occur in this phase due to domain-specific restrictions in the output structure, like in the following cases:

- Output slots that take values from a predefined set.
- Output slots that are forced to be instantiated.
- Extracted information that generate multiple output templates. For instance, in the MUC-6 financial domain, when a succession event is found that involves a person leaving and another person taking up the same job in an organization, two different output templates have to be generated: one for the person leaving and another for the person starting.
- Output slots that have to be normalized. For instance, dates, products that have to be normalized with a code from a standard list, etc.

5 Machine Learning for Information Extraction

It is well known that each module involved in the extraction process achieves results with certain accuracies. This fact leads to the *error propagation* problem, meaning that a small error could produce a greater one as the extraction process advances.

On the other hand, IE systems need to be as portable as possible to new situations. Dealing with new domains and/or author styles implies that different kinds and amounts of new specific knowledge will be needed to achieve good results. Moreover, new extraction scenarios could imply new concepts to be dealt with, which are beyond IE system's capabilities.

These two problems, then, are what cause the difficulty of exploiting IE technology.

In order to handle such difficulties, Adaptive IE technology has benefited from the improvements achieved in all the involved tasks over the last two decades. Most of these improvements are based on the use of empirical methods in NLP. Given the kinds of knowledge needed by empirical approaches to NLP, machine learning techniques have been widely used for its acquisition: Sekine et al. [1998]; Borthwick et al. [1998]; Baluja et al. [1999]; Borthwick [1999]; Takeuchi and Collier [2002]; Yarowsky [2003] for NE recognition, Cardie et al. [2000] for chunking, and McCarthy and Lehnert [1995]; Aone and Bennet [1996]; Cardie and Wagstaff [1999]; Mitkov [1998]; Ng and Cardie [2003] for coreference resolution. Detailed thorough surveys on the use of ML techniques for NLP tasks can be also found Young and Bloothooft [1997]; Manning and Schütze [1999]; Mooney and Cardie [1999].

Most of the research effort in this Adaptive IE has been devoted to applying symbolic inductive learning methods to the acquisition of domain-dependent knowledge that is useful for extraction tasks. Most of the approaches focus on acquiring IE rules from a set of training documents. These rules can be classified either as *single-slot rules* or *multi-slot rules*, given that a concept can be represented as a template (e.g., a template element, a template relation or a scenario template in MUC terms). A single-slot rule is able to extract document fragments related to one slot within a template, while a multi-slot rule extracts tuples of document fragments related to the set of slots within a template. The representation of extraction rules depends heavily on the document style, from which rules have to be learned. In general, the less structured the documents, the greater the variety of linguistic constraints for a given rule. Some surveys presenting different kinds of rules used for IE can be found Muslea [1999]; Glickman and Jones [1999].

Since MUC, typical IE-rule learners focus on learning rules from free text to deal with the extraction of events (i.e., ST tasks). With some exceptions, these rules are useful for extracting document fragments containing slot-filler values and postprocessing is needed in order to select the exact values from the fragments. However, the large amount of online documents available on the Internet has recently increased the interest in algorithms that can automatically process and mine these documents by extracting relevant exact values. Within this framework, some systems have been developed and applied to learning single-slot rules. Few efforts have focused on learning other types of knowledge useful for such a task (e.g., Hidden Markov Models (HMMs), hyperplane separators).

Table 1 shows a classification of different state-of-the-art ML approaches for the acquisition of such domain-dependent knowledge. In this table, different versions of an initial approach have been grouped in the first column. The second column (KL) refers to the type of knowledge learned by the approaches. The third column (Paradigm) relates to the learning paradigm of the approach, either propositional (i.e., based on representing the examples of a concept in terms of zero order logic or attribute-value logic) or relational (that is, based on representing the examples of a concept in terms of first order logic). In the fourth column (Strategy) the learning strategy of the approach is shown. The fifth column (EX) shows whether the approach learn knowledge that is useful to extract exact slot-filler fragments or not. The final column (DOC) refers to the type of documents the approach uses for the learning (free text -f-,

APPROACH	KL	PARADIGM	STRATEGY	EX	DOC
AutoSlog Riloff [1993] AutoSlog-TS Riloff [1996]	rules	propositional learning	heuristic driven specialization	no	f
Harabagiu and Maiorano [2000]					
PALKA Kim and Moldovan [1995]			candidate elimination		
Chai and Biermann [1997] TIMES Chai et al. [1999]			brute force		
Basili et al. [2000]			heuristic driven generalization		
CRYSTAL Soderland et al. [1995] Soderland [1997] WAVE Aseltine [1999]			bottom-up covering		s
ExDISCO Yangarber [2000] Yangarber [2003]					f
ESSENCE Català [2003]					
DIPRE Brin [1998] Snowball Agichtein and Gravano [2000]				yes	
LIEP Huffman [1995]		relational learning		no	
WHISK Soderland [1999]			top-down covering	yes	st/s/f
EVIUS Turmo and Rodríguez [2002]					s/f
SRV Freitag [1998a]					s
RAPIER Califf [1998]			bottom-up compression		
Seymore et al. [1999]	statistical models	propositional learning	MLE		
Freitag and McCallum [1999]			shrinkage		
Freitag and McCallum [2000]			shrinkage + stochastic optimization		
McCallum et al. [2000]			GIS		
Peshkin and Pfeffer [2003]			EM		
Ray and Craven [2001] Skounakis et al. [2003]			MLE + discriminative training	no	f
Miller et al. [1998] Miller et al. [2000]			MLE + interpolated estimation	yes	
Chieu and Ng [2002]			ME classification		s/f
Cox et al. [2005]					s
Kambhatla [2004]			Quasi-Newton optimization		f
Sun et al. [2003]			maximum margin separator	no	f
Alice-SVM Chieu et al. [2003]					
Zelenko et al. [2003]				yes	
Zhao and Grishman [2005]					
ELIE Finn and Kushmerick [2004]					s
SNoW-IE Roth and Yih [2001]			mistake-driven		

Table 1: A classification of ML approaches for IE specific knowledge acquisition.

semi-structured documents -s-, and structured ones -st-). More details on the different approaches are provided below. Here, the approaches are first classified according to the kind of knowledge learned: the first section is devoted to learning rules, while the second one deals with learning statistical models. Section 5.3 describes multi-strategy approaches, and Section 5.4 presents the state-of-the-art in wrapper generation. Finally, Section 5.5 compares the performance of some of those approaches that have used a common evaluation domain.

5.1 Learning Rules

Since the MUC-5 conference, some IE systems have focused on the application of approaches to automatically acquire IE rules. Such approaches can be classified from different points of view: the degree of supervision, the kind of rules learned, the document style of the training set (i.e., the kind of document style to which rules learned can be applied), the learning paradigm, or the learning strategy used, among others. In this section, some state-of-the-art IE-rule learning systems are classified according to these criteria. For the sake of simplicity, the degree of supervision they need has been taken as the starting point for the comparison.

5.1.1 Supervised Approaches

Under the symbolic inductive learning paradigm, supervised approaches are the most common ones for learning IE rules. In general, a learning method is supervised if some intervention is required from the user for the learning process. Some approaches require the user to provide training examples. This supervision can be carried out as a preprocess (i.e., appropriately tagging the examples occurring in the training set) or as an online process (i.e., dynamically tagging the examples as needed during the learning process). Some of these learning approaches focus on propositional learning, while others focus on relational learning.

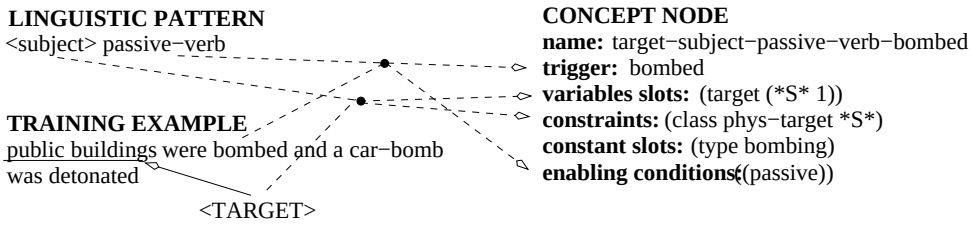

Figure 5: A concept node induced by AutoSlog.

Propositional Learning As mentioned, propositional learning is based on representing the examples of a concept in terms of either zero order logic or attribute-value logic, which have equivalent expressiveness in a strict mathematical sense. Within this learning paradigm, some IE research groups have developed learning systems that learn IE rules from positive examples. In general, examples of a concept are represented as sets of attributes (i.e. slots) whose values (i.e., slot fillers) are heads of syntactic phrases occurring within

the training documents. Rules learned by these approaches are useful extracting information from parsed free text. They identify the syntactic phrases that contain the heads that fill the slots. Often, however, partial matches occur between these phrases and the exact slot filler. This is sufficient when the aim is to extract an approximation of the relevant information. When exact values are required, a postprocess is mandatory in order to zero in on the desired content.

One of the earliest approaches was AutoSlog Riloff [1993]. AutoSlog generates single-slot rules, named *concept nodes*, by applying a heuristic-driven specialization strategy. A concept node is defined as a *trigger word* that will activate the concept node when performing the extraction, and a set of restrictions involving the trigger and the slot-filler values. Each training example is a chunk within a sentence annotated as a slot-filler value. The generation of concept nodes is based on the specialization of a set of predefined heuristics in the form of general linguistic patterns. The generation process is carried out by examining each example in an anotated corpus only once. Such linguistic patterns contain syntactic constraints which generate a concept node when a specialization occurs. For instance, in Figure 5, the linguistic pattern <**subject**> **passive-verb** is specialized into <**target**>**bombed** when examining the training example *public buildings were bombed and a car bomb was detonated* for the bombing event (*public buildings* was previously annotated as slot-filler value). As a consequence, a concept node for the <**target**> slot of a bombing template is generated with **bombed** as trigger and constraining the slot-filler value to be subject (i.e., *S*) within the training example. The resulting set of concept nodes is proposed to the user, in order to be reviewed. This is due to the fact that AutoSlog makes no attempt to generalize examples, and, consequently, generates very specific slot-filler definitions (i.e., rules with low coverage).

Other approaches learn rules by generalizing examples. Some of them can learn single-slot rules and multi-slot rules, such as PALKA Kim and Moldovan [1995], CRYSTAL Soderland et al. [1995], WAVE Aseltine [1999] and Chai and Biermann Chai and Biermann [1997], while some others learn single-slot rules such as TIMES Chai et al. [1999]. With the exception of PALKA, these systems translate the training examples into specific rules, to which a generalization process is applied.

PALKA is based on a candidate-elimination algorithm. The user must provide specific rules (*Frame-phrasal Patterns Structures* -FP- structures). These specific rules are similar to AutoSlog's. However, chunks that are slot-filler values are represented as the semantic class of their heads. PALKA generalizes and specializes such semantic classes by using an ad-hoc semantic hierarchy until the resulting FP-structures cover all the initial specific ones. The use of the semantic hierarchy allows PALKA to learn rules that are more general than AutoSlog's. However, no generalizations are made on trigger words.

Specific rules used by CRYSTAL are also in the form of concept nodes, but they consist of a set of features related to the slot-fillers and the trigger. Values for these features can be terms, heads, semantic classes, syntactic relations (subject, direct or indirect object), verbal modes, etc. CRYSTAL uses a bottom-up covering algorithm in order to relax such features in the initial specific rules. This relaxation is achieved by means of both dropping out irrelevant features, and generalizing semantic constraints by using an ad-hoc semantic hierarchy. For instance, the concept node for the succession event shown in Figure

6 was learned by CRYSTAL to be used in MUC6 competition. Filler values for slots *Person-In*, *Position* and *Organization* are extracted when their constraints match a parsed sentence.

CONCEPT NODE
concept type: Succession Event
constraints:
SUBJ::
classes include: <Person>
extract: Person_In
VERB::
terms include: NAMED
mode: passive
OBJ::
terms include: OF
classes include: <Corporate Post>, <Organization>
extract: Position, Organization

Figure 6: A concept node learned by CRYSTAL.

In addition, Webfoot Soderland [1997] was presented as a preprocess module for CRYSTAL in order to learn IE rules from more structured documents, such as web pages. Webfoot allows partitioning web pages into HTML-tagged text segments, which are used by CRYSTAL as if they were sentences. WAVE is similar to CRYSTAL, but it consists in an incremental learning approach in which a reusable hierarchy of partially learned rules is maintained during learning process.

Rules learned by CRYSTAL are more expressive than those learned by AutoSlog or PALKA. This is due to the fact that CRYSTAL can learn rules where constraints for both the slot-filler values and the triggers can be based not on specific words but on generalizations. However, the ad-hoc semantic hierarchy has to be manually built when dealing with new domains.

A different approach is presented by Chai and Biermann Chai and Biermann [1997], in which a broad-coverage semantic hierarchy, WordNet[22]Miller et al. [1990], is used. By default, the system assigns the most frequently used synset[23] in WordNet to the head of each slot-filler within a specific rule. The user, however, can assign a more appropriate sense. In a first step, specific rules are semantically generalized by using a brute-force generalization algorithm to keep recall as high as possible. This process consists in replacing each noun synset in a specific rule by its top hypernym in WordNet. In a second step, these top synsets are specialized by tuning the rules to adjust the precision. This is done by a) applying generalized rules to the training set, b) keeping those sub-hierarchies of synsets that can specialize the top synsets within the rules according to the training set, c) requiring the user to select those from the resulting specializations which are relevant for the domain, d) computing a relevance rate for each of the resulting synsets and, finally, e) requiring a

[22] http://www.cogsci.princeton.edu/~wn

[23] A synset in WordNet groups together a set of word senses, variants, related by a loose kind of synonymy.

generalization threshold from the user in order to select the appropriate synsets for each rule.

A more recent version of this approach is followed by TIMES, in which two generalization steps (syntactic and semantic) are performed against specific rules in order to generate the maximum number of candidates for single-slot rules. This is done by 1) selecting and permuting different combinations of constituents within each annotated sentence separately for each slot, and 2) applying the brute-force algorithm explained before.

Rule sets learned by TIMES are more general than those learned by both CRYSTAL and the approach of Chai and Biermann Chai and Biermann [1997]. This is due to the use of permutations of constituents in the generalization step. However, as opposed to CRYSTAL, TIMES and the approach of Chai and Biermann Chai and Biermann [1997] also need the user's supervision during the learning process.

Relational Learning Relational learning is based on representing the examples of a concept in terms of first order logic. Within this learning paradigm, most IE-rule learning systems are supervised, representing training examples in terms of attributes and relations between textual elements (e.g., tokens, constituents).

Within this paradigm, LIEP Huffman [1995] automatically infers multi-slot rules from training examples of events occurring in free text by using a bottom-up covering algorithm. If a given example is not matched by any learned rule, LIEP attempts to further generalize a rule. If this is not possible, LIEP builds a new specific rule from the example. Specific rules used by LIEP consist of a feature set, similar to that used by CRYSTAL. However, as opposed to CRYSTAL, LIEP has no prior information about the syntactic relations between chunks. LIEP learns such relations ($subject(A, B)$, $object(A, B)$, $prep\_object(A, B)$, etc.) by using a form of explanation-based learning with an over-generated and incomplete theory. This is why LIEP works within relational learning. The generalization proceeds by creating disjunctive values, so LIEP rules cannot take into account missing values in the training corpus. As a consequence, CRYSTAL rules are more expressive than LIEP's.

As opposed to most approaches, LIEP requires the user to interactively annotate training examples within sentences until (s)he considers there is enough coverage. This is done by means of an interface. Other approaches based on acquisition interfaces are HASTEN Krupka [1995] and PET interfaces Yangarber and Grishman [1998]. However, instead of using an automatic generalization process, they help the user to manually build IE rules from free text.

These approaches can require fewer training examples than those explained before, because the user can select the most appropriate examples at each iteration for further generalization. However, supervision is required during the learning process.

Some other approaches are based on general relational learning systems, and more specifically, on Inductive Logic Programming (ILP) systems well known by the ML community (e.g., FOIL Quinlan [1990]; Quinlan and Cameron-Jones [1993], CIGOL Muggleton and Buntine [1988], GOLEM Muggleton and Feng [1992], CHILLIN Zelle and Mooney [1994] and PROGOL Muggleton [1995]). Two examples of these approaches are SRV Freitag [1998a] and RAPIER Califf

[1998] . Both systems are applied to semi-structured documents for learning single-slot rules to extract exact values.

SRV is an ILP system based on FOIL. SRV transforms the problem of learning IE rules into a classification problem: is a document fragment a possible slot value? The input of this system is a training set of documents, and a set of attributive and relational features related to tokens T (e.g., $capitalized(T)$, $next(T_1, T_2)$) that control the generalization process. Introducing domain-specific linguistics or any other information is a separate task from the central invariable algorithm, which is domain independent. SRV uses a top-down covering algorithm to learn IE rules from positive and negative examples. Slot-filler fragments within training documents are manually annotated as positive examples. Negative examples are automatically generated taking into account empirical observations related to the number of tokens of positive examples: if positive examples are sequences of between MIN to MAX tokens, negative examples are the rest of sequences of between MIN to MAX tokens in the training corpus. In Figure 7, a rule learned by SRV from semi-structured documents related to seminar announcements is depicted. This rule extracts exact values for the slot *speaker*.

```
speaker:-                                          // Fragment F is a speaker if
  some(?A, [], word, *unknown*)                    // F contains a token (A), and
  every(capitalizedp, true)                        // every token in F is capitalized, and
  length(=, 2))                                    // F contains exactly 2 tokens, and
  some(?B, [], word, *unknown*)                    // F contains another token (B), and
  some(?B, [prev_token], word, ":")                // B is preceded by a colon, and
  some(?A, [next_token], doubletonp, false)        // A is not followed by a 2-char token, and
  every(quadruple_char_p,false)                    // every token in F does not consists of
                                                   // exactly 4 alpha. characters, and
  some(?B, [prev_token prev_token], word, "who")   // two tokens before B is the word "who"
```

Figure 7: A rule learned by SRV.

RAPIER is a relational learning system based on GOLEM, CHILLIN and PROGOL. It uses a bottom-up compression algorithm in which rules are iteratively merged, instead of generalized, from training examples. RAPIER considers training examples as specific rules. At each iteration, two rules (specific or not) are selected to be compressed into a new one. Rules used in this process are discarded and the resulting rule is added to the set of possible ones. The input documents are represented as token sequences, optionally POS tagged. No parsing process is required for them. Each training example consists of three text fragments, where one of them is the slot-filler value and the other two are the unbounded contexts to the left and right of the value. In order to learn rules from such examples, RAPIER takes into account the implicit token succession relation and some token generalizations: token sets, POS tags or semantics derived from the WordNet hierarchy. The incremental version of RAPIER uses active learning Thompson et al. [1999].

A more flexible system within the relational learning paradigm is WHISK Soderland [1999]. WHISK deals with both structured and semistructured documents, and also with free text. Following a different approach, WHISK represents documents as sequences of tokens, some of them tags representing metadata (HTML tags, delimiters of parsed chunks, features of heads,...) and allows

learning of both single-slot and multi-slot rules to extract exact slot values. Figure 8 shows an example of a single-slot rule learned by WHISK to extract *speakers* from semi-structured documents related to seminar announcements. Concretely, rules are represented as pairs $< pattern, output >$, in which *pattern* is meant to be matched by documents and *output* is required to be the output template when a match occurs. The pattern is a regular expression that represents possible slot fillers and their boundaries. For instance, pattern ** ':' ('Alan' *) ','* in Figure 8 represents possible fillers for one slot. These fillers are token sequences beginning with token *'Alan'* and enclosed by tokens *':'* and *','*. The special token *** matches any token sequence. The *output* format allows assigning the fillers to their related slots. This is done with variables that identify the i-th filler matching the pattern. For instance, the *output Seminar* {*speaker $1*} in the figure assigns the token sequence matching expression *('Alan' *)* to slot *speaker* within a template of type *Seminar*.

These rules are learned in a top-down fashion from a training set of positive examples. An unusual selective sampling approach is used by WHISK. Initially, a set of unannotated documents is randomly selected as training input out of those satisfying a set of key words. These documents are presented to the user who tags the slot-fillers. WHISK starts by learning a rule from the most general pattern (e.g., *'*(*)*'* for single-slot rules). The growth of the rule proceeds one slot at a time. This is done by adding tokens just within the slot-filler boundaries as well as outside them. The growth of a rule continues until it covers at least the training set. After a rule set has been created, a new set of unannotated documents can be selected as a new training input from those satisfying the rule set.

```
Pattern:: * ': ' ( 'Alan' *) ','
Output:: Seminar {Speaker $1}
```

Figure 8: A rule learned by WHISK.

Although WHISK is the most flexible state-of-the-art approach, it cannot generalize on semantics when learning from free text, as CRYSTAL, PALKA, SRV and RAPIER do. Another limitation of WHISK is that no negative constraints can be learned.

5.1.2 Towards Unsupervised Approaches

The learning approaches presented above require the user to provide positive training examples in order to automatically learn rules. One of the main drawbacks of this supervision is the high cost of annotating positive examples in the training documents. Some approaches focus on dealing with this drawback by requiring a lower degree of supervision.

One of the first supervised learning approaches to require less manual effort was AutoSlog-TS Riloff [1996]. It was a new version of AutoSlog, where the user only had to annotate documents containing text as relevant or non-relevant before learning. The strategy of AutoSlog-TS consists of two stages. In the first, it applies the heuristic-driven specialization used by AutoSlog in order to generate all possible rules (concept nodes, see Figure 5) with the relevant documents. This is done by matching a set of general linguistic patterns against

the previously parsed sentences of the relevant documents. In the second stage, a relevance rate is computed for each one of the resulting rules as the conditional probability that a text is relevant given that it activates the particular rule. The relevance formula is the following:

$$Pr(relevant\_text|text\_contains\_rule_i) = \frac{rel\_freq_i}{total\_freq_i}$$

where rel_freq_i is the number of matches of $rule_i$ found in the relevant documents, and $total - freq_i$ is the total number of matches of $rule_i$ found in the whole set of documents. Finally, each rule is ranked according to the formula[24]:

$$\begin{cases} relevance\_rate(rule_i) * log_2(freq_i) & if\ relevance\_rate(rule_i) > 0.5 \\ 0 & otherwise. \end{cases}$$

and the n best ranked rules (n according to the user criteria) are selected. The author presented a comparison between AutoSlog and AutoSlog-TS related to the learning of single-slot rules to extract 3 slots defined in MUC-4 domain (*perpetrator*, *victim* and *target* in the terrorism domain). The main conclusion was that AutoSlog-TS can extract relevant information with comparable performance to AutoSlog's, but requiring significantly less supervision and being significantly more effective at reducing spurious extractions. However, the relevance rate formula tends to rank many useful rules at the bottom and to rank high frequency rules at the top. This is why the author concludes that a better ranking function is needed.

In general, more recent approaches that learn from unannotated free text require some initial domain-specific knowledge (i.e., a few keywords or initial hand-crafted rules) and/or some validations from the user in order to learn effectively Català et al. [2000]; Català [2003]; Basili et al. [2000]; Harabagiu and Maiorano [2000]; Yangarber and Grishman [2000]; Yangarber [2000, 2003].

The approach presented by Harabagiu and Maiorano Harabagiu and Maiorano [2000] is also based on heuristic-driven specializations, similarly to AutoSlog. However, the authors pay special attention to mining the conceptual relations explicitly and implicitly represented in WordNet in order to minimize supervision as well as to overcome the low coverage produced by AutoSlog and AutoSlog-TS. On the one hand, supervision is minimized by requiring a set of keywords relevant to the domain from the user, instead of annotated examples (as AutoSlog does) or documents (as AutoSlog-TS does). On the other hand, coverage is increased by applying a set of linguistic patterns (heuristics) more general than those used in AutoSlog. The approach consists of three stages. In the first, references of the input keywords in WordNet (e.g., their synsets and their taxonomic relations, an occurrence of one keyword in the gloss of the synset corresponding to another keyword, keywords cooccurring in the gloss of a synset, etc) are found in order to achieve possible explicit and implicit relations among concepts relevant to the domain. As a consequence, a semantic representation of the relevant concepts is built. This semantic space can be

[24]Riloff assumes that the corpus is 50% relevant and, consequently, when the relevance rate is lower or equal to 0.5 the rule is negatively correlated with the domain

seen as a set of linguistic patterns more general than those used by AutoSlog and AutoSlog-TS. In the second stage, those parsed chunks labelled as subject, verb and object within sentences of the training corpus are scanned to allocate collocations of domain concepts within the semantic space. Using the principle of maximal coverage against these semantic collocations and taking into account the syntactic links emerging from them to the parsed chunks, a set of linguistic patterns is generated. Finally, in the third stage, only the most general linguistic patterns are selected. However, no automatic method for this selection is suggested by the authors, and no results of the coverage of the learned patterns are provided.

Basili Basili et al. [2000], however, used heuristic-driven generalizations to induce linguistic patterns useful for extracting events. The approach requires documents classified into a set of specific domains. At an initial step, the set of verbs that are relevant triggers of events is automatically selected for each domain D_i. This is done by considering the following assumption: *if events of a given type are included in the documents, it is reasonable to assume that their distribution in the sample is singular (i.e., non random).* Authors assume a $\mathcal{X}^2$ distribution of the events in the documents. They use the following $\mathcal{X}^2$-test to determine if a verb v occurring in D_i is a relevant trigger of an event:

$$f_i^v \geq \alpha$$
$$\mathcal{X}_v^2 = \frac{(f_i^v - F_v)^2}{F_v} \leq \beta$$

where f_i^v is the number of occurrences of verb v in documents belonging to D_i, and F_v is the overall number of v occurrences in all the documents. Values for α and β are determined according to the size and nature of the corpus. Those verbs accomplishing this statistical test are used as triggers for event matching and, for each one of them, a set of verb sub-categorization structures is extracted by applying a conceptual clustering algorithm. This is done by taking into account all the occurrences of the verb and their arguments found in the parsed sentences of the domain corpus. These occurrences are translated into vectors of attribute:value pairs in the form *syntactic_relation:argument_head* in order to be clustered. Each one of the resulting clusters represents a verb sub-categorization structure with its corresponding specific patterns (specific instances of *argument_head* for each *syntactic_relation*). The heads are semantically tagged using WordNet synsets and the resulting specific patterns are generalized by using the following heuristics: a) synsets of noun heads are semantically generalized using a measure of conceptual density Agirre and Rigau [1996] and, b) patterns are expanded via linguistically principled transformations (e.g., passivization and potential alternations). Finally, multi-slot IE rules are built from these generalized event patterns by manually marking the argument that fills each slot of a pre-defined event template. Validations from the user could be necessary to eliminate possible noisy verbs and overly specific patterns obtained during the learning process.

An alternative to a heuristic-driven approach is ESSENCE Català et al. [2000]; Català [2003]. It is based on inducing linguistic patterns from a set of observations, instead of examples. These observations are automatically generated from unannotated training documents as a keyword (provided by the

user) in a limited context. For instance, a possible observation to learn rules useful for the extraction of events from sentences could be defined by a relevant verb and the pairs <head,preposition> (preposition can be NULL) occurring in the k syntactic chunks closest to the left and the k ones to the right. These observations are generalized by performing a bottom-up covering algorithm and using WordNet. After the learning phase, the user is required to validate the resulting patterns, and this learning process may be repeated by using both the set of validated patterns and a set of new observations generated from new keywords. Finally, the user has to manually mark slot fillers occurring in the linguistic patterns. The resulting rules are similar to CRYSTAL's.

Some research groups have been focusing on the use of a certain form of learning known as *bootstrapping* Brin [1998]; Agichtein and Gravano [2000]; Yangarber [2000, 2003]. All of them are based on the use of a set of either seed examples or seed patterns from which they learn some context conditions that then enable them to hypothesize new positive examples, from which they learn new context conditions, and so on. In general, all the methods following such approach use a bottom-up covering algorithm to learn rules.

Following bootstrapping approach, DIPRE Brin [1998] is a system for acquiring patterns which is able to extract binary relations from web documents. Very simple patterns are learned from a set of seed word pairs that fulfil the target relation (for example, Company - Location). The seed word pairs are used to search web pages for text fragments where one word appears very close to the other. In this case, a pattern is created which expresses the fact that both semantic categories are separated by the same lexical items that separate the example seed words in the text fragment found. A pattern is composed by five string fields: *prefix category1 middle category2 suffix*. A text fragment matches the pattern if it can be split to match each field. For instance, to learn the relation (Author, Book Title) from web pages, DIPRE learned the pattern '<LI><B>*title*</B> by *author* (', where the text preceding the title is the *prefix*, the text between the title and the author is the *middle* and the *suffix* consists of the text following the author[25]. The set of patterns obtained from the example relations are used to find new pairs of related words by matching the patterns with the present set of web pages and the process is repeated. It remains open whether the success of this system is mainly due to the fact that the title is always linked to the same author.

Finally, ExDISCO Yangarber et al. [2000]; Yangarber [2000] is a bootstrapping method in which extraction patterns in the form of subject-verb-object (SVO) are learned from an initial set of SVO patterns manually build. The application of these initial patterns in a text indicates that the text is suitable for extracting a target event or a part of it. By applying the set of seed patterns, the unannotated corpus is divided into relevant and irrelevant texts. An exploratory search for patterns SVO statistically correlated with the set of relevant texts allows one to guess new extraction patterns that can be used to search for new relevant documents, and so on. The resulting patterns are in the form of basic syntactic chunks semantically annotated (depending on their heads). Like most of the other less unsupervised approaches, a human expert has to indicate which slots of the output template are to be filled by each learned

[25]Note that the learned pattern takes advantage of HTML tags but they are not necessary for the algorithm to work in free texts.

pattern.

In spite of the fact that the bootstrapping approach is very appealing due to its reduction in handcrafting, it does present some problems. The main disadvantage of bootstrapping approaches is that, although the initial set of seed examples could be very reliable for the task in hand, the accuracy of the learned patterns quickly decreases if any wrong patterns are accepted in a single round. Systems based on bootstrapping techniques must incorporate statistical or confidence measures for patterns in order to limit this problem Agichtein and Gravano [2000]; Yangarber [2003]. Yangarber Yangarber [2003] presents the counter-training method for unsupervised pattern learning that aims at finding a condition to stop learning while maintaining the method unsupervised. To do this, different learners for different scenarios are trained in parallel. Each learner computes the precision of each pattern in term of positive evidence (i.e., how much relevant the pattern is with respect to the particular scenario) and negative evidence (i.e., how relevant is with respect to the rest of scenarios). This negative evidence is provided by the rest of learners. If the pattern achieves greater negative evidence than positive one, then the pattern is not considered for acceptance to the particular scenario. The algorithm proceeds until just one learner remains active given that, in this case, negative evidence cannot be provided.

Another drawback of the bootstrapping techniques is that they need a large corpus (on the order of several thousand texts), which is not feasible in some domains. Finally, the bootstrapping approach is also dependent on the set of seed examples that are provided by the expert. A bad set of seed examples could lead to a poor set of extraction patterns.

5.2 Learning Statistical Models

Although rule learning techniques have been the most common ones used for IE, several approaches explore the use of well-known statistical machine learning methods which have not been previously applied to this area. These methods include Markov Models, Maximum Entropy Models, Dynamic Bayesian Networks or Hyperplane Separators. This section is devoted to the brief description of the application of some of these approaches to information extraction tasks. All these approaches belong to the propositional learning paradigm.

5.2.1 Markov Models

Within this framework, some efforts have focused on learning different variants of HMMs as useful knowledge to extract relevant fragments from online documents available on the Internet. Until recently, HMMs had been widely applied to several NL tasks (such as PoS tagging, NE recognition and speech recognition), but not in IE. Although they provide an efficient and robust probabilistic tool, they need large amounts of training data and in principle imply the necessity of an a priori notion of the model structure (the number of states and the transitions between the states). Moreover, as they are generative models (they assign a joint probability to paired observation and label sequences, and their parameters are trained to maximize the likelihood of training examples), it is extremely difficult for them to represent either non-independent features or long-range dependencies of the observations.

In general, the efforts on learning HMMs have taken into account only words. For instance, Freitag and McCallum Freitag and McCallum [1999] propose a methodology in which a separate HMM is constructed by hand for each target slot to be extracted, its structure focusing on modeling the immediate prefix, suffix, and internal structure of each slot. For each HMM, both the state transition and word emission probabilities are learned from labeled data. However, they integrate a statistical technique called *shrinkage* in order to be able to learn more robust HMM emission probabilities when dealing with data-sparseness in the training data (large emission vocabulary with respect to the number of training examples). In fact, the type of shrinkage used, which averages among different HMM states (the ones with poor data versus the data-rich ones), is the one known in speech recognition as *deleted interpolation*. The method has been evaluated on the domains of on-line seminar announcements and newswire articles on corporate acquisitions, in which relevant data must be recovered from documents containing a lot of irrelevant text (*sparse extraction*).

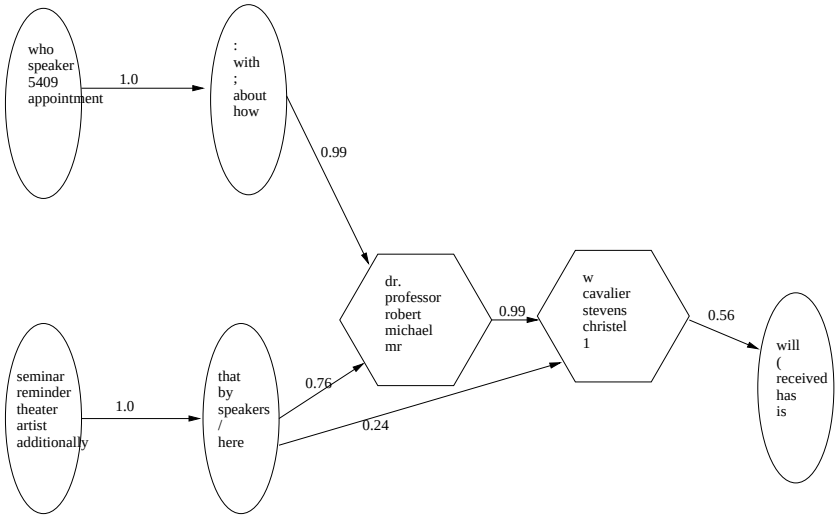

Figure 9: Part of the HMM structure for extracting the *speaker* field in Freitag and McCallum's system.

Figure 9 shows part of the structure of an HMM for extracting the *speaker* field in the on-line seminar announcement domain. The elliptical nodes represent the prefix/suffix states of the field to be extracted, whereas the polygonal nodes represent the field states themselves. In both types of nodes, the top 5 most probable tokens to be emited by that state are shown. Only those transition probabilities greater than 0.1 are depicted.

The authors claim better results than the SRV system (described above in Section 5.1.1, and developed by one of the authors), albeit needing the a priori definition of the topology of the model and the existence of labeled data.

In an extension of the previous approach, Freitag and McCallum [2000], the sparse extraction task is tackled again, but this time the work focuses on robustly learning an HMM structure for each target slot from limited specific training data. Starting from a simple model, a hill-climbing process is performed in the space of possible structures, at each step applying the seven possible defined

operations (state splitting, state addition, etc.) to the model and selecting the structure with the best score as the next model. The score used is F_1 (the harmonic mean of precision and recall), evaluated on the training data from the same two domains as their previous work Freitag and McCallum [1999], along with the semi-structured domains of job announcements and *Call for paper* announcements. Training data must be labeled. The estimation of the parameters of the HMMs obtained is performed as described in their previous work. Experimental results show a higher accuracy than the one achieved by their previous approach, as well as the ones from SRV and RAPIER systems (c.f., Section 5.1.1).

In contrast to the previous approach, Seymore Seymore et al. [1999] presents a method for both learning the HMM's topology and training the HMM (estimating the probabilities of both the transitions between the states and the emision of class-specific words from each state) from training data. The approach uses a single HMM to extract a set of fields from quite-structured texts (e.g. computer science research paper headers), taking into account field sequence. The fields are close to each other ("dense extraction"). While the selection of the model structure needs data labeled with information about the target-slot to be extracted in order to be accomplished, the HMM parameters can be estimated either from labeled data (via maximum likelihood estimates) or from unlabeled data (using the widely known Baum-Welch training algorithm, Baum [1972]). A good step towards portability is the introduction of the concept of distantly-labeled data (labeled data from another domain whose labels partially overlap those from the target domain), whose use improves classification accuracy. On the other hand, a clear drawback is the need of large amounts of training data in order to maximize accuracy.

Other approaches not only use words, but benefit from additional non-independent word features (e.g., POS tags, capitalization, position in the document, etc.), or from features of sequences of words (e.g., length, indentation, total amount of white-space, grammatical features, etc.). This is the case of the approach presented by McCallum McCallum et al. [2000], in which the task of segmenting frequently asked questions into their constituent parts is addressed. The approach introduces maximum entropy Markov models (MEMMs), a conditional-probability finite state model in which the generative HMM parameters are replaced by a single function combining the transition and emission parameters. This permits modelling transitions in terms of the multiple overlapping features mentioned above, by means of exponential models fitted by Maximum Entropy. The structure of the Markov model must be a priori defined, though a labeled training corpus is not strictly necessary for the estimation of the parameters of the model.

The work of Ray and Craven Ray and Craven [2001] represents the first application of HMMs to the extraction of information from free text. The approach aims at extracting and building n-ary relations in a single augmented finite state machine (that is, a multiple slot extraction task). However, since the intention is to represent grammatical information of the sentences in the HMM structure, it only operates over relations formed within one sentence. The states in the HMM represent annotated segments of a sentence (previously parsed with a shallow parser), starting from a fully connected model. Examples annotated with the relationships are needed, and the training algorithm maximizes the probability of assigning the correct labels to certain segments instead of max-

imizing the likelihood of the sentences themselves, akin to the optimization of parameters according to several features described in the previous approach, McCallum et al. [2000]. The methodology is used for extracting two binary relationships from biomedical texts. Skounakis Skounakis et al. [2003] provides an extension of the Ray and Craven's work in which *hierarchical* hidden Markov Models (HHMMs, HMMs with more than one level of states) are used to represent a richer multi-level grammatical representation of the sentences. HHMMs are further extended by incorporating information about context (*context hierarchical* HMMs).

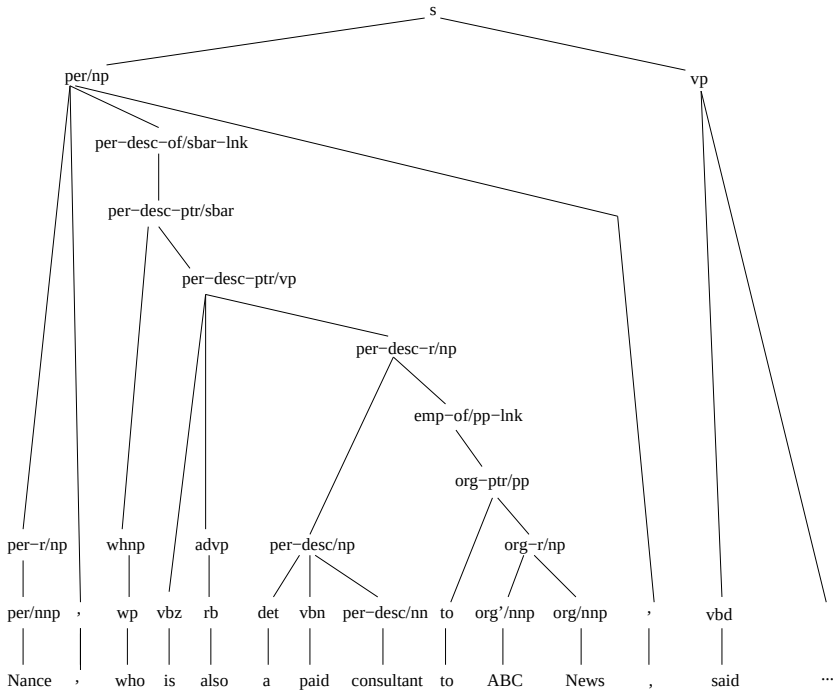

Figure 10: An example of augmented parse in Miller et al.'s formalism.

5.2.2 Other Generative Statistical Models

Along the lines of Vilain Vilain [1999] (c.f., Section 4.2), in which a set of grammatical relations among entities is defined, Miller Miller et al. [1998, 2000] proposes and approach to learning a statistical model that adapts a lexicalized, probabilistic context-free parser with head rules (LPCFG-HR) in order to do syntactico-semantic parsing and semantic information extraction. The parser uses a generative statistical model very similar to that of Collins Collins [1997], though parse trees are augmented with semantic information. Figure 10 depicts an example of these augmented parse trees. In the intermediate nodes, the possible prefix denotes the type of entity (e.g. *per* for person), plus an additional tag indicating whether the node is a proper name (*-r*) or its descriptor (*-desc*). Relations between entities are annotated by labeling the lowermost parse node that spans both entities (inserting nodes when necessary to distinguish the arguments of each relation).

This integrated model, which performs part-of-speech-tagging, name finding, parsing and semantic interpretation intends to avoid the error propagation mentioned in Section 4. Manual semantic annotation is required for training, although this is the only annotation needed, since the LPCFG parser (previously trained on the Penn Treebank Marcus et al. [1993]) is used to, withoutsupervision, create a syntactic training news corpus consistent with the supervised semantic annotation.

5.2.3 Maximum Entropy Models

Chieu and Ng Chieu and Ng [2002] make use of the maximum entropy framework, like McCallum (McCallum et al. [2000]), but instead of basing their approach on Markov models, they use a classification-based approach. A set of features are defined for each domain of application, from which the probability distribution is estimated that both satisfies the constraints between features and observations in the training corpus and makes as few additional assumptions as possible (according to the maximum entropy principle). They develop two techniques, one for single-slot information extraction on semi-structured domains and the other for multi-slot extraction on free text. The first one is applied to the Seminar Announcements domain. A trained classifier distributes each word into one of the possible slots to be filled (classes). The more complex multi-slot extraction task is applied to the Management Succession domain (using the same training and test data as WHISK). A series of classifiers is used to identify relations between slot fillers within the same template (an example is depicted in Figure 11). The parameters of the model are estimated by a procedure called Generalized Iterative Scaling (GIS).

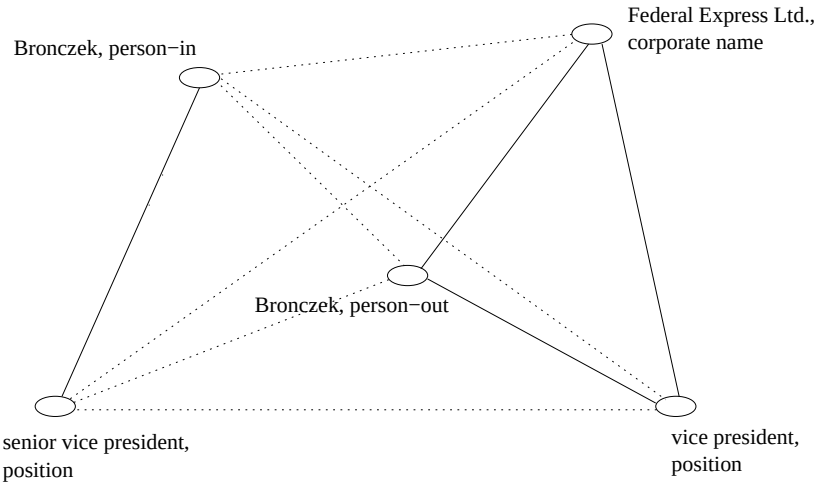

Figure 11: Result of relation classification for the sentence "*Bronczek, vice president of Federal Express Ltd., was named senior vice president, Europe, Africa and Mediterranean, at this air-express concern*" in Chieu and Ng's system.

Kambhatla Kambhatla [2004] applies a Maximum Entropy model to the hard ACE EDT task (c.f., Section 3). As in the previous approach, the prediction of the type of relation between every pair of entity mentions in a sentence is mod-

eled as a classification problem with up to two classes for each relation subtype defined by ACE (since most of them are not symmetric) plus one additional class for the case where there is no relation between the two mentions.

The ME models are trained using combinations of lexical, semantic and syntactic features (the latter derived in turn from a syntactic and dependency tree obtained using a ME-based parser). The ME framework allows the easy extension of the number and type of features considered. The author claims to have obtained the best results on the ACE 2003 evaluation set (though the ACE rules do not allow the publication of the actual ranking among the global set of participants).

5.2.4 Dynamic Bayesian Networks

Dynamic Bayesian networks (DBN) are a generalization of HMMs which allow the encoding of interdependencies among various features. Peshkin and Pfeffer Peshkin and Pfeffer [2003] introduce an approach which uses DBNs to integrate several language features (PoS tags, lemmas, forming part of a syntactic phrase, simple semantic tags, etc.) into a single probabilistic model. The structure of the bayesian network must be a priori manually defined, as well as the features to be considered. Then, the inference and training algorithms are similar to those for HMMs. Once more the IE problem is converted into a classification problem of returning the corresponding target slot of each token in the corpus. The approach has been evaluated on the Seminar Announcements domain and performed comparably to the previous systems described (Rapier, SRV, WHISK and HMMs).

5.2.5 Conditional Random Fields

Conditional random fields (CRFs) Lafferty et al. [2001] are another type of conditional-probability finite state model. Like the maximum entropy Markov models described above, they are discriminative instead of generative which allows them the use of different types of features to model the observed data. CRFs are undirected graphs which are trained to maximize conditional probability of outputs given inputs with unnormalized transition probabilites (i.e., they use a global exponential model for the entire sequence of labels given the observation sequence instead of the per-state models used by MEMMs).

CRFs represent a promising approach. For instance, McCallum and Jensen McCallum and Jensen [2003] propose what they refer to as *extraction-mining random fields*, a family of unified probabilistic models for both information extraction and data mining. Focusing on relational data, the use of a common inference procedure allows inferencing either bottom-up for extraction or top-down for data mining, and thus the intermediate results obtained can be compared so as to improve the accuracy of both processes. That is to say, the output of data mining can be used as additional features for the extraction model, while the output of information extraction can provide additional hypotheses to data mining. No experimental results have been reported.

The first strict application of CRFs to IE we are aware of is the system presented by Cox Cox et al. [2005] as part of the Pascal challenge shared task in the workshop announcements domain (c.f., Section 3). The global performance

of the system over the test set was quite good (obtaining global F-scores of 65%, third among the presented systems).

5.2.6 Hyperplane Separators

Also within the propositional learning paradigm, and given the success of hyperplane classifiers like Support Vector Machines (SVM) in classification tasks, several researchers have attempted to apply them to IE tasks. Prior to applying these methods it is necessary to represent the IE problem as a classification problem. Note that once the IE problem has been translated into a classification problem, several other ML methods can be applied, like Decision Trees, Naive Bayes and others. But it seems that hyperplane separators present some features that make them specially suitable for NLP tasks, for instance, their ability to deal with a large number of features.

Hyperplane separators learn a hyperplane in the space of features (the input space in SVM terminology) that separates positive from negative examples for the concept to be learned. When such a hyperplane cannot be found in the input space, it can be found in an extended space built from a combination of the features in the input space. Some hyperplane classifiers, like Support Vector Machines and Voted Perceptrons, are able to find such hyperplanes in the extended space by using Kernel functions. Kernel functions return the dot product between two examples in the extended space without explicitly going there. This information is enough for the mentioned algorithms to directly find the hyperplane in the extended space.

The work of Roth and Yih Roth and Yih [2001] is the first which attempted to learn relational features using hyperplane classifiers. They present a new approach for learning to extract slot fillers from semi-structured documents. This approach is named SNoW-IE and it follows a two-step strategy. In the first step, a classifier is learned to achieve high recall. Given that a common property of IE tasks is that negative examples are extremely more frequent than positive ones, this classifier aims at filtering most of the former without discarding the latter. In the second step, another classifier is learned to achieve high precision. Both classifiers are learned as sparse networks of linear functions from the manually annotated training set by performing SNoW Roth [1998], a propositional learning system. Training examples are represented as conjunctions of propositions. Basically, each proposition refers to an attribute of some token. This token can occur as part of the slot filler, in its left context (l_window) or in its right context (r_window). Positive and negative training examples are automatically generated by using a set of constraints, such as the appropriate length (in tokens) of the context windows, the maximum length of the slot filler, and the set of appropriate features for tokens within the filler or either of its context windows (e.g., word, POS tag or location within the context). These constraints are defined by the user for each concept to be learned.

For example, in Roth and Yih Roth and Yih [2001], the constraints defined to generate training examples related to concept *speaker* in the seminar announcement domain can be described as follows[26]: a *speaker* may be represented as the conjunction of its previous two words in the left context window (with their positions relative to the *speaker* itself), the POS tag corresponding to the slot filler

[26] See Roth and Yih [2001] for details on the formalism used to define these constraints.

Document fragment: ... room 1112 . **Professor Warren Baler** from ...

SPEAKER

	l_window		filler		r_window
negative example 1:	[1112 .]		[Professor]		[Warren]
propositionalization:	1112_−2&._−1	&	N	&	PN_1
negative example 2:	[1112 .]		[Professor Warren]		[Baler]
propositionalization:	1112_−2&._−1	&	N	&	PN_2
	1112_−3&._−1	&	PN	&	PN_1
positive example:	[1112 .]		[Professor Warren Baler]		[from]
propositionalization:	1112_−2&._−1	&	N	&	Prep_3
	1112_−3&._−2	&	PN	&	Prep_2
	1112_−4&._−3	&	PN	&	Prep_1
negative example 3:	[. Professor]		[Warren]		[Baler]
propositionalization:	._−2&Professor_−1	&	PN	&	PN_1
⋮			⋮		

Figure 12: Training examples generated by SNoW-IE.

and the first POS tag in the right context window (also with its position relative to the slot filler). Figure 12 shows some training examples generated from the fragment "*... room 1112. Professor Warren Baler from ...*" from a document in the seminar announcement domain. This fragment contains one correct filler for *speaker* ("*Professor Warren Baler*") and a set of incorrect ones (e.g., "*Professor*", "*Professor Warren*", "*Warren*", "*Warren Baler*"). For instance, the first negative example in the figure consists of the filler "*Professor*" (a noun -N-), the left context "*1112 .*" and the right context "*Warren*" (a proper noun -PN-). This example is represented by proposition 1112_-2&._-1&N&PN_1, where numbers represent positions of tokens with respect to "*Professor*". Moreover, three propositions are generated for the positive example occurring in the fragment. For instance, the first one (1112_-2&._-1&N&Prep_3) takes POS tag N related to word "*Professor*" as the tag representing the filler. Note that the preposition in the right context window is located three tokens to the right of word "*Professor*". The second one (1112_-3&._-2&PN&Prep_2) takes POS tag PN corresponding to word "*Warren*" in the filler. In this case the preposition in the right context window is located two tokens to the right of word "*Warren*", while the punctuation mark in the left window is two tokens to the left.

Sun Sun et al. [2003] presents the results of applying a SVM to the MUC-4 IE task about terrorism attacks. The methodology is divided into three steps: document parsing, feature acquisition and extraction model construction. In the first step, the system generates parse trees for each sentence in the documents. In the second step, each sentence is represented by a set of features derived from the parse tree that includes context features (information about

other surrounding constituents in the sentence) and content features about the noun phrase to be extracted from the sentence. For example, the parse tree corresponding to the sentence "*Two terrorists destroyed several power poles on 29th street and machinegunned several transformers.*" is shown in Figure 13. The target slot is "*several power poles*" and the context features are defined from the terms surrounding it. Not all surrounding terms have the same feature weight because it depends on how close to the target slot the term is found in the parse tree: a high value will be assigned to "*on*" and a smaller value to "*Two terrorists*". The context features for the target slot "*several power poles*" are the terms "*on*", "*29th street*", "*destroyed*" and "*Two terrorists*".

```
(S (NP Two terrorists)
   (VP (VP destroyed
           (NP several power poles)
           (PP on
               (NP 29th street)))
       and
       (VP machinegunned
           (NP several transformers)))
   .)
```

Figure 13: Parse tree example generated by Sun et al. system.

Each sentence is represented then as a list of attribute-value pairs and it is labeled as positive or negative for the target slot. A sentence is considered as positive if it contains an entity that matches the target slot answer keys. A SVM with a polynomial kernel is used to learn a hyperplane that separates positive from negative examples. Results in the MUC-4 domain are not very good in the test sets (F-scores of 36% for TST3 and 33% for TST4), and the authors claim that further research on additional features for training can be necessary in order to improve overall performance.

Chieu Chieu et al. [2003] uses also the MUC-4 IE task to show that by using state of the art machine learning algorithms in all steps of an IE system, it is possible to achieve competitive scores when compared to the best systems for that task (all of them handcrafted). After a preprocessing step, the system they propose, ALICE, generates a full parse tree for each sentence in the text (with linked coreferences where necessary). In order to learn to extract information from these sentences, the core of the system learns one classifier for each different target slot. Each sentence is represented in a propositional way by using generic features that can be easily derived from the parse tree, like agent of the verb, head-word, etc. The authors were not committed to any classifier and tried different approaches to learn the classifier for each slot. The best algorithms (that is, the ones achieving the best results with less tuning of parameters required), were Maximum Entropy and SVM (Maximum Entropy achieved a slightly better performance than SVM). Both algorithms show competitive results with respect to human engineered systems for the MUC-4 task[27]. The SVM was tested using a linear kernel, that is, the SVM tried to find a hyperplane in the input space directly.

[27]It is worth noting that the results are much better than the ones presented by Sun Sun et al. [2003], that also used a SVM in the same domain.

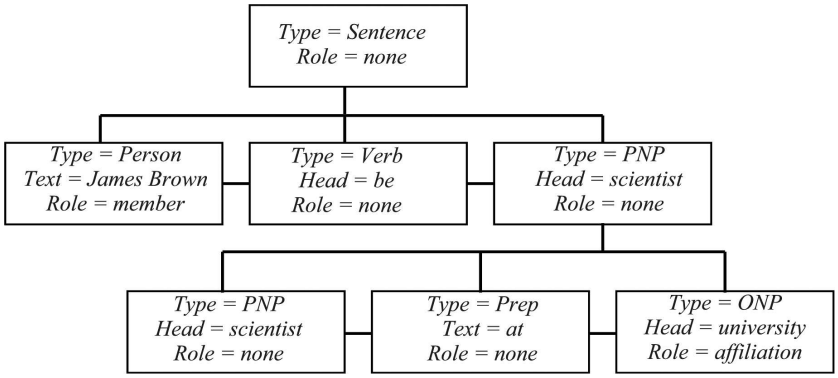

Figure 14: A relation example generated from the shallow parse of the sentence "*James Brown was a scientist at the University of Illinois*" by Zelenko et al.'s approach.

Similarly, Zelenko Zelenko et al. [2003] presents specific kernels forextracting relations using Support Vector Machines. The distinctive property of these kernels is that they do not explicitly generate features, i.e., an example is not a feature vector as is usual in ML algorithms. The new kernels are inspired by previously existing kernels that can be used to find similarities between tree structures. Sentences from the text are converted into examples as trees of partial parses where some nodes are enriched with semantic information about the role of the node (that is, the slot it should fill in the output template). Figure 14 shows an example for the "*person-affiliation*" relation. A relation example is the least common subtree containing two entity nodes.

The authors test their kernels using two hyperplane classifiers that can take advantage of kernel information: Support Vector Machines and Voted Perceptrons. They compare these results with the ones obtained using both Naive Bayes and Winnow algorithms. Note that the representation of the examples used in the latter class of algorithms is not trees but propositional descriptions, since these algorithms cannot deal with trees. The test IE task consists in the extraction of *person-affiliation* and *organization-location* relations from 200 news articles from different news agencies. The kernelized algorithms show a better F-score than both Naive Bayes and Winnow for both relation extraction tasks.

A different approach is presented by Finn and Kushmerick Finn and Kushmerick [2004] in which they convert the IE task into a token classification task, where every fragment in a document must be classified as the start position of a target slot, the end of a target slot, or neither. Their ELIE algorithm consists in the combination of the predictions of two sets of classifiers. The first set (L1) learns to detect the start and the end of fragments to be extracted; the second one (L2) learns to detect either the end of a fragment given its beginning, or the beginning of a fragment given its end. Whereas the L1 classifiers generally have high precision but low recall, the L2 classifiers are used to increase the recall of the IE system. ELIE has been evaluated on three different domains: seminar announcements, job postings and Reuters corporate acquisition. In these experiments and when compared with other kind of learning methods (rela-

tional learning, wrapper induction, propositional learning), ELIE$_{L1}$ alone often outperforms the other methods. ELIE$_{L2}$ improves recall while maintaining precision high enough. This gives the choice of using either one classifier alone or both classifiers depending on the recall/precision levels required for a specific task. ELIE has been also evaluated on the Pascal challenge on Evaluation of Machine Learning for Information Extraction obtaining a poorer performance than that obtained in the previous experiments. The authors suggest that the effect of data imbalance (many more negative than positive examples of a field start or end) is the cause of the poor results.

The approach presented by Zhao and Grishman Zhao and Grishman [2005], also based on kernel methods, investigates the incorporation of different features corresponding to different levels of syntactic processing (tokenization, parsing and deep dependency analysis) in relation extraction. After the definition of syntactic kernels representing results from shallow and deep processing, these kernels are combined into new kernels. The latter kernels introduce new features that could not be obtained by individual kernels alone. The approach is evaluated on the 2004 ACE Relation Detection task using two different classifiers (KNN and SVM). From the results obtained they show that the addition of kernels improves performance but that chunking kernels give the highest contribution to the overall performance.

5.3 Multi-strategy Approaches

The advantage of using a multi-strategy approach in learning to extract information was demonstrated by Freitag for learning from online documents Freitag [1998a]. Single strategy approaches for this purpose take a specific view of the documents (e.g., HTML tags, typographic information, lexical information). This introduces biases that make such approaches less suitable for some kinds of documents than for others.

In this experiment, Freitag Freitag [1998a] focused on combining three separate machine learning paradigms for learning single-slot rules: rote memorization, term-space text classification, and relational rule induction. When performing extraction, the confidence factors of the learning algorithms were mapped into probabilities of correctness by using a regression model. Such probabilities were combined in order to produce a consensus among learning algorithms. This combination of algorithms (each one of them using different kinds of information for learning) achieved better results than when applied individually.

Within the relational learning paradigm, a different multi-strategy approach is used by EVIUS Turmo and Rodríguez [2002]; Turmo [2002] to learn single-slot and multi-slot IE rules from semi-structured documents and free text. The learning systems explained so far learn single concept extractions. They learn knowledge useful to extract instances of a concept within the extraction scenario independently. Instead, EVIUS assumes the fact that the extraction scenario imposes some dependencies among concepts to be dealt with. When one concept depends on another one, knowledge about the former is useful for learning to extract instances of the target.

EVIUS is a supervised multi-concept learning system based on a multi-strategy constructive learning approach Michalski [1993] that integrates closed-loop learning, deductive restructuring Ko [1998] and constructive induction.

Closed-loop learning allows EVIUS to incrementally learn IE rules similar to Horn clauses for the whole extraction scenario. This is done by means of determining which concept to learn at each step. Within this incremental process, the learning of IE rules for each concept is basically accomplished using FOIL, which requires positive and negative examples. Positive examples are annotated in the training data using an interface, while negative examples are automatically generated. Once IE rules for a concept have been learned, the learning space is updated using deductive restructuring and constructive induction. These techniques assimilate knowledge which may be useful for further learning: the training examples of learned concepts and new predicates related to these concepts.

5.4 Wrapper Generation

Wrapper Generation (WG) can be considered as a special case of IE dealing with structured and semi-structured text. Other approaches are possible, however, and WG can be placed in the intersection of three loosely related disciplines: Heterogenous Databases, Information Integration and Information Extraction.

Following Eikvil [1999], the purpose of a wrapper is extracting the content of a particular information source and delivering the relevant content in a self-describing representation. Although wrappers are not limited to the web, most of their current applications belong to this domain. In the web environment, a wrapper can be defined as a processor that converts information implicitly stored as in an HTML document into information explicitly stored as a data structure for further processing.

Web pages can be ranked in terms of their format from structured to unstructured. Structured pages, follow a predefined and strict, but usually unknown, format where itemized information presents uniform syntactic clues. In semi-structured pages, some of these constraints are relaxed and attributes can be omitted, multi-valued or changed in its order of occurrence. Unstructured pages, usually consist of free text merged with HTML tags not following any particular structure.

Most existing WG systems are applied to structured or semi-structured web pages.

The performance of a wrapper does not differ basically from the performance of an Information Extractor. Knowledge is encoded in rules that are applied over the raw text (in a pattern matching process) or over a more elaborated or enriched data source (sequence of tokens, set of predicates, HTML tree, etc.). The most important difference is that tokens include not only words but also HTML tags. This fact has important consequences: On the one hand, HTML tags provide additional information that can be used for extraction; on the other hand, the presence of HTML tags makes it difficult to apply linguistic based approaches to extraction.

In the early systems, building wrappers was approached as a manual task. Several generic grammar development systems (Yacc, Perl, LL(k) grammars, Xpath) or specialized ones (as in WHIRL or ARANEUS) have been used, together with graphical user interfaces and other support tools. This approach is highly costly (e.g., Jango, a commercial system for comparison shopping on the Web, reports several hundred wrappers have to be built and maintained). Due

to this high cost, there is a growing interest in applying ML techniques (ILP, grammar induction, statistical methods) to automate the WG task.

WG systems can be classified according to different criteria: degree of elaboration of data sources, expressiveness of wrappers to be generated, ML techniques applied, etc. Some systems operate on raw texts or on the result of simple tokenization, usually focusing on the detection of words, punctuation signs, control characters and HTML tags. Other systems require more powerful tokenization (numeric, alphabetic, uppercase, etc.). In all these cases, the input to the wrapper consists of a sequence of tokens. Some other wrappers need the input to be organized as an HTML parse tree while, in others, additional linguistic processing is performed on the input data (POS tagging, NE recognition, semantic labeling, etc.). Finally, in some systems, input content has to be mapped into a propositional or relational (predicate based) representation.

We can consider a wrapper W as a parser able to recognize a particular language L_w. The expressivity of the wrapper is directly related to the power of the class of languages it can recognize. Regular grammars are quite expressive for dealing with most of the requirements of an extractor but, as pointed out by Chidlovskii Chidlovskii [2000], they cannot be learned with the usual grammatical induction methods if only positive examples have to be used. For WG to be effective, learning has to be carried out with a very small training set, and additional constraints have to be set. Chidlovskii Chidlovskii [2000] proposes using k-reversible grammars. Stalker Muslea et al. [2001, 2003] and SoftMealy Hsu and Dung [1998] use limited forms of Finite State Transducers (FST). WIEN Kushmerick [2000] limits itself to a set of 6 classes of PAC-learnable schemata.

Learning approaches range from ILP, frequently used by systems coming from the IE area (e.g., SRV, RAPIER, WHISK), to greedy covering algorithms with different kinds of generalization steps (Stalker, $(LP)^2$), Constraint Satisfaction (WIEN) or several types of combinations (e.g., BWI Freitag and Kushmerick [2000]).

One of the most influential systems is WIEN (Wrapper Induction Environment), presented by Nicholas Kushmerick in his thesis Kushmerick [1997], and summarized in Kushmerick [2000].

WIEN deals with 6 classes of wrappers (4 tabular and 2 nested). These classes are demonstrated to be PAC-learnable and Kushmerick reports a coverage of over 70% of common cases. Basically, multi-slot itemized page fragments are well covered by the system. The simplest WIEN class is LR. A wrapper belonging to this class is able to recognize and extract k-slot tuples guided by the left and right contexts (sequences of tokens) of each slot. So the wrapper has 2k parameters, $< l_1, r_1, \ldots, l_k, r_k >$, to be learned. WIEN learns its parameter set in a supervised way, with a very limited amount of positive examples. It uses a Constraint Satisfaction approach with constraints derived from some hard assumptions on the independence of parameters. The most complex and accurate tabular class, HOCLRT (Head Open Close Left Right Tail), considers four additional parameters for modeling the head and tail of the region from where the information has to be extracted, and the open and close contexts for each tuple.

SoftMealy Hsu [1998]; Hsu and Dung [1998] tries to overcome some of the limitations in WIEN's HOCLRT schemata by relaxing some of the rigid constraints that were imposed on the tuple's contents. SoftMealy allows for multiple-valued or missing attributes, variations on attribute order and the use of a candi-

date's features for guiding the extraction. A wrapper is represented as a nondeterministic FST. Input text is tokenized and is treated by the wrapper as a sequence of separators. A separator is an invisible borderline between two adjacent tokens. Separators are represented as pairs $< s^L, s^R >$, where s^L and s^R are sequences of tokens representing the left and right contexts, respectively. The learning algorithm proceeds by generalizing from labeled tuples. Generalization is performed by tree-climbing on a taxonomy of tokens (e.g., "IBM" < Alluppercase < word < string).

Stalker Muslea et al. [2001, 2003] is another well known WG system. While input to WIEN or SoftMealy was simply sequences of tokens, in the case of Stalker a description of the structure of the page, in terms of the so called Embedded Catalog formalism, is used as well. The Embedded catalog description of a web page is a tree-like structure where the items of interest are placed in the leaves. Wrappers are represented as linear landmark automata (LLA), a subclass of general landmark automata. Transitions of these automata are labeled with landmarks (i.e., sequences of tokens and wildcards, including textual wildcards and user defined domain specific ones). Stalker produces an ordered list of LLA using a sequential covering algorithm with a small set of heuristics.

Wrappers generated by Stalker can be considered as a generalization of HOCLRT wrappers (Stalker wrappers without disjunction and wildcards can be reduced to WIEN's).

In Knoblock et al. [2001], an extension of Stalker is presented using Cotesting. Co-testing is a form of active learning that analyzes the set of unlabeled examples to automatically identify highly informative examples for the user to label. After the learning of forward and backward sets of rules from labeled examples, both sets are applied to a set of unlabeled pages. Those examples on which the two sets of rules disagree are asked to the user for labeling next.

WHIRL, Word-based Heterogeneous Information Representation Language Cohen [2000], is a wrapper language that uses a powerful data model, Simple Texts in Relations (STIR), to build different extraction applications. WHIRL needs the information pages to be previously parsed in order to obtain the HTML parse tree. The result is represented in the form of tree-description predicates from which relational rules are learned.

In Cohen and Jensen [2001]; Cohen et al. [2002] the authors propose an extensible architecture, following basically the same ideas of WHIRL with a sounder formalization. In their system a wrapper consists of an ordered set of builders, where each builder is associated with a restricted sublanguage. Each builder is assumed to implement two basic operations (Least General Generalization, and Refine) in such a way as to allow several forms of composition in order to implement complex wrappers.

Ciravegna Ciravegna [2001] describes Learning Pattern by Language Processing ($(LP)^2$), a general IE system that works very well in wrapper tasks. $(LP)^2$ proceeds in two learning steps: tagging rules and correction rules. Rules are conventional condition-action rules, where the conditions are constraints on the k tokens preceding and following the current token and the action part inserts a single tag (beginning or ending a string to be extracted). Initially, the constraints are set on words but incrementally, as learning proceeds, some generalizations are carried out and constraints are set on additional knowledge (e.g., POS tagging, shallow NLP, user defined classes). It uses a sequential covering algorithm and a beam-search for selecting the best generalizations that can be

applied at each step.

Freitag and Kushmerick Freitag and Kushmerick [2000] present Boosted Wrapper Induction (BWI), a system that uses boosting for learning accurate complex wrappers by combining simple, high precision, low coverage basic wrappers (boundary detectors). Boundary detectors consist of a pair of patterns (prefix and suffix of the boundary) and a confidence score, while a wrapper is a triple $< F, A, H >$ where $F = \{F_1, \ldots, F_T\}$ is the set of fore detectors, $A = \{A_1, \ldots, A_T\}$ is the set of aft detectors and $H(k)$ is the probability that the field has length k.

5.5 Comparison of Performance of ML Approaches

An exhaustive direct comparison of performance across the different ML approaches is impossible, since generally they have been tested on different domains. However, some domains, such as MUCs' and the seminar announcements, have become standard evaluation domains for a significant set of authors.

In this section we try to provide a comparison among those systems working on the seminar announcement domain. We have chosen this domain because it represents the most commonly used domain among the ML approaches presented in this survey. However, we are aware that there may still be differences in the evaluation framework (e.g., different partitions of the corpus for testing and training, different evaluation procedures, different definitions of what is considered a correct slot filler, etc). To our knowledge, there are insufficient comparable results of ML approaches for IE in regard to one of the MUC or ACE domains. Which is why we do not report on them.

The seminar announcements corpus consists of 485 semi-structured documents of on-line university seminar announcements[28] (an example is shown in Section 2). As mentioned, the extraction scenario consists of four single-slot tasks where information about the starting time (**stime**), the ending time (**etime**), the speaker (**speaker**) and the location (**location**) of each seminar announcement must be extracted. Table 2 lists the F-scores obtained by the different approaches using this domain. Most approaches explicitly consider that the tasks consist in extracting only one single correct filler for each target slot (possible slot fillers can occur several times in the document). Most of the approaches adopt the same validation methodology, partitioning the document collection several times into training and testing sets of the same size and averaging over the results. However, they differ on the number of runs: three for HMM2, five for Naive Bayes, SRV, Rapier, Evius, SNOW-IE and ME_2, and ten for the rest. The only exceptions are HMM1, that does not report the validation methodology used[29], and WHISK, that uses a ten-fold cross validation with 100 documents.

Table 2 indicates that, generally, the statistical methods (those in the first block of the table) outperform the rule learners in the seminar announcement tasks. Rule learners show a similar performance to each other, except WHISK (though as mentioned above its evaluation method is completely different).

All the systems perform well on the **stime** slot. This seems to be the easiest task, given that a Naive Bayes approach is enough to achieve good results. This

[28] http://www.cs.cmu.edu/ dayne/SeminarAnnouncements/_Source_.html

[29] Other authors claim different scores obtained for HMM1 approach. We provide the results from the original work.

APPROACH	stime	etime	location	speaker
Naive Bayes Roth and Yih [2001]	98.3	94.6	68.6	35.5
SNOW-IE Roth and Yih [2001]	**99.6**	96.3	75.2	73.8
ELIE$_{L1}$ Finn and Kushmerick [2004]	96.6	87.0	84.8	84.9
ELIE$_{L2}$ Finn and Kushmerick [2004]	98.5	96.4	86.5	**88.5**
HMM1 Freitag and McCallum [1999]	99.1	59.5	83.9	71.1
HMM2 Freitag and McCallum [2000]	-	-	**87.5**	76.9
ME$_2$ Chieu and Ng [2002]	**99.6**	94.2	82.6	72.6
BIEN Peshkin and Pfeffer [2003]	96.0	**98.8**	87.1	76.9
SRV Freitag [1998a]	98.5	77.9	72.2	56.3
RAPIER Califf [1998]	95.9	94.6	73.4	53.1
EVIUS Turmo and Rodríguez [2002]	96.1	94.8	73.3	55.3
WHISK Soderland [1999]	92.6	86.1	66.6	18.3
LP^2 Ciravegna [2001]	99.0	95.5	75.1	77.6

Table 2: Results in F-score achieved by some ML approaches in the seminar announcements domain

is also true for the **etime** slot. However, HMM1 is significantly worse than the rest of the approaches. SRV performance is also low with respect to this slot. The reason for these lower scores is that these approaches tend to favor recall, and precision seems to be damaged because of two facts: on one hand, there are seminars without any value for **etime**, and on the other hand, values for **etime** are very similar to those for **stime**.

With respect to **location** and **speaker**, they are more difficult to learn than **etime** and **stime**. A Naive Bayes approach seems to be insufficient. The statistical approaches perform significanly better than the rule learners, with the exception of LP^2.

In general, different specific issues may affect performance, and may affect it differently depending on the approach. For instance, ELIE obtains the highest F-score for the **speaker** slot thanks to the use of an external gazeteer of first and last names. Similarly, LP^2 achieves the highest score among the rule learners for the same reason. However, a gazeteer is also used by ME$_2$, and the authors note that it does not seem to improve the performance on this slot very much. Another example of how specific issues may improve performance is shown in BIEN. It obtains the highest value for the **etime** slot thanks to the use of hidden variables reflecting the order in which the target information occurs in the document. However, it requires the manual definition of the structure of the Dynamic Bayesian Network used.

The preprocess required is other factor to take into account. Some methods (SVM methods in general and ELIE in particular for this domain) have to deal with a large number of features, and make use of a previous filtering process by means of information gain, whose high cost must be considered. Most methods use contextual features, and most of them require the definition of a context window that can largely differ in length among the approaches. Different definitions represent different learning biases. Most methods generally use some sort of previous shallow NL processing. There are approaches such as HMMs and ME$_2$ which do not need this preprocess, avoiding on one hand this cost and preventing on the other hand from the corresponding error propagation.

6 Methodologies and Use of Knowledge in IE Systems

Extraction from structured or semi-structured documents can be performed without making use of any post-process, and frequently with the use of few preprocessing steps. Within this framework, automatically induced wrappers and IE rules learned by using SRV, RAPIER, or WHISK, can either be directly applied to the extraction task as an independent IE system, or integrated as a component into an already existing IE system for specific tasks. This is why this section aims at comparing architectures of IE systems for free text only and specifically for the 15 most representative of the state of the art: CIRCUS Lehnert et al. [1991, 1992, 1993] and its successor BADGER Fisher et al. [1995], FASTUS Appelt et al. [1992, 1993b,a, 1995], LOUELLA Childs et al. [1995], PLUM Weischedel et al. [1991, 1992, 1993]; Weischedel [1995], IE^2 Aone et al. [1998], PROTEUS Grishman and Sterling [1993]; Grishman [1995]; Yangarber and Grishman [1998], ALEMBIC Aberdeen et al. [1993, 1995], HASTEN Krupka [1995], LOLITA Morgan et al. [1995]; Garigliano et al. [1998], LaSIE Gaizauskas et al. [1995], its successor LaSIE-II Humphreys et al. [1998], PIE Lin [1995], SIFT Miller et al. [1998, 2000] and TURBIO Turmo [2002]. Most of these systems participated in MUC competitions, and their architectures are well documented in proceedings up to 1998. More recent IE systems have participated in ACE. However, as described in Section 3, there are no published proceedings for the ACE evaluations, and although some ACE participants have published work related to the learning of IE patterns in international conferences (c.f., Section 5), we have not found any descriptions of complete IE systems.

<table>
<tr><th>SYSTEM</th><th>SYNTAX</th><th>SEMANTICS</th><th>DISCOURSE</th></tr>
<tr><td>LaSIE</td><td colspan="3" rowspan="3">in-depth understanding</td></tr>
<tr><td>LaSIE-II</td></tr>
<tr><td>LOLITA</td></tr>
<tr><td>CIRCUS</td><td rowspan="5">chunking</td><td rowspan="3">pattern matching</td><td rowspan="2">template merging</td></tr>
<tr><td>FASTUS</td></tr>
<tr><td>BADGER</td></tr>
<tr><td>HASTEN</td><td rowspan="2">-</td></tr>
<tr><td>PROTEUS</td></tr>
<tr><td>ALEMBIC</td><td>grammatical relation interpretation</td><td rowspan="3">traditional semantic interpretation procedures</td></tr>
<tr><td>PIE</td><td rowspan="3">partial parsing</td><td rowspan="3">pattern matching</td></tr>
<tr><td>TURBIO</td><td>-</td></tr>
<tr><td>PLUM</td><td rowspan="2">template merging</td></tr>
<tr><td>IE^2</td><td colspan="2">pattern matching</td></tr>
<tr><td>LOUELLA</td><td colspan="2" rowspan="2">-</td></tr>
<tr><td>SIFT</td><td colspan="2">syntactico-semantic parsing</td></tr>
</table>

Table 3: Methodology description of state-of-the-art IE systems.

The comparisons do not take into account either the preprocess or the output template generation methods because there are no important differences among different IE systems. Table 3 summarizes each system's approach to syntactic parsing, semantic interpretation and discourse analysis from the viewpoint of the methodology they use to perform extraction. The pros and cons of each method have been presented in Section 4. Table 4 describes the kind of knowledge

SYSTEM	SYNTAX	SEMANTICS	DISCOURSE
LaSIE	general grammar extracted from the Penn TreeBank corpus Gaizauskas [1995]	λ-expressions	
LaSIE-II	hand-crafted stratified general grammar		-
LOLITA	general grammar	hand-crafted semantic network	
CIRCUS	phrasal grammars	concept nodes learned from AutoSlog	
FASTUS		hand-crafted IE rules	
BADGER		concept nodes learned from CRYSTAL	trainable decision trees
HASTEN		E-graphs	
PROTEUS		IE rules learned from ExDISCO	
ALEMBIC		hand-crafted grammatical relations	-
TURBIO		IE rules learned from EVIUS	
PIE	general grammar	hand-crafted IE rules	
PLUM			
IE^2	hand-crafted IE rules		hand-crafted rules and trainable decision trees
LOUELLA			-
SIFT	statistical model for syntactico-semantic parsing learned from the Penn TreeBank corpus and on-domain annotated texts Miller et al. [2000]		

Table 4: Description of knowledge used by state-of-the-art IE systems.

SYSTEM	TASK		
	TE	TR	ST
IE^2	86.76	75.63	50.79
SIFT	83.49	71.23	
LaSIE-II	77.17	54.7	44.04
PROTEUS	76.5		42

Table 5: Results in F-score for the best MUC-7 systems

representation used by the selected IE systems. As shown in this table, only a few of the systems take advantage of ML techniques to automatically acquire the domain-specific knowledge useful for extraction (i.e., CIRCUS, BADGER, PROTEUS, TURBIO and SIFT). These IE systems are more easily portable than the rest. In the case of ALEMBIC, although the authors suggested the use of a statistical model to identify some grammatical relations Vilain [1999], in the end they relied on hand-crafting(c.f., Section 4.2) in the last MUC in which they participated (MUC-6). We have not found recent evidence of the integration of automatically acquired knowledge in ALEMBIC. PROTEUS can use the PET interface to assist in manually building IE rules (as it did in MUC-7 competition), as well as ExDISCO (c.f., 5.1.2) to automatically learn them.

Table 5 shows the F-score per IE task achieved by the best systems in MUC-7. As described in Section 3, TE, TR and ST refer to Template Element, Template Relationship, and Scenario Template, respectively. Note that the

best results were achieved by IE^2, which used hand-crafted knowledge. SIFT did not participate in event extraction (ST task). However, the results of SIFT were close to those achieved by IE^2 in the tasks in which both systems were involved. This is an interesting result, considering that SIFT used automatically learned knowledge (c.f., Table 4). Similarly, the results achieved by LaSIE-II and PROTEUS were close, but the latter system is more easily adaptable to new domains by using either the PET interface or ExDISCO. Note that although the authors of PROTEUS did not provide results for TR tasks, the system is able to deal with relation extraction.

The following sections describe the architecture of IE^2, as the best system in MUC-7, and both PROTEUS and SIFT, as the ones that are more easily portable to new domains. In these descriptions, the modules of the different architectures are grouped according to their functionalities (c.f., Section 4). Those modules referred to by the methodologies in Table 3 appear colored in the descriptive figures.

6.1 The IE^2 System

This system uses a total of six modules as shown in Figure 15, and none is devoted to adapt the system to new domains (although the integration of automatically learned knowledge -IE rules- may be possible).

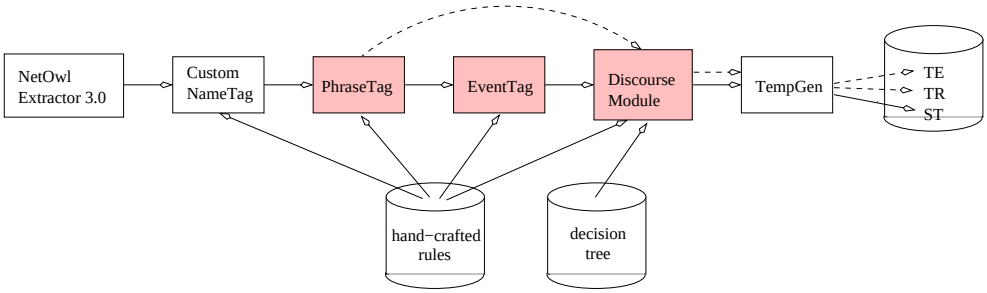

Figure 15: IE^2 system architecture.

6.1.1 Preprocessing

The first two modules focus on NE recognition and classification tasks. Given an input document, they automatically annotate every NE occurring in the document with XML-tags. The first module is a commercial software (*NetOwl Extractor 3.0*) to recognize general NE types. It deals with time and numerical expressions, names of persons, places and organizations, aliases (e.g., acronyms of organizations and locations), and their possible semantic subtypes (e.g., company, government organization, country, city).

The second module (*Custom NameTag*) is used to recognize restricted-domain NE types by means of pattern matching. In the case of MUC-7 domain, Launch events, it was manually tunned to deal with different types (air, ground and water) and subtypes of vehicle names (e.g., plane, helicopter, tank, car, ship, submarine). All these phrases are SGML-tagged into the same document.

6.1.2 Syntactico-semantic interpretation

The modules *PhraseTag* and *EventTag* focus on SGML-tagging those phrases in each sentence that are values for slots defined in TE, TR and ST templates. This goal is achieved by using a cascaded, partial syntactico-semantic parser, and the process generates partially filled templates.

First, the module *PhraseTag* applies syntactico-semantic rules to identify the noun phrases in which the previously recognized NEs occur (including complex noun phrases with modifiers). Next, the same module finds TE and TR slot values by means of a noun phrase tagger especially designed to recognize specific noun phrases (e.g., names of people, organizations and artifacts). These rules take into account the presence of appositions and copula constructions in order to find local links between entities. This is due to the fact that the authors of IE^2 suggest that appositions and copula constructions are commonly found in documents to represent information related to TE and TR tasks of MUC. Normally, this process generates partial templates for TE tasks, given that in general the slot values are found in different sentences. This can also occur for TR tasks.

Finally, the module *EventTag* applies a set of hand-crafted syntactico-semantic multi-slot rules to extract values for slots of events from each sentence (i.e., for the ST task).

6.1.3 Discourse analysis

A post-process of template merging is required for the three tasks (TE, TR and ST) in order to integrate the partial event structures achieved from the different sentences. The *Discourse Module* focuses on coreference resolution, in order to merge the noun phrases describing slot values obtained in the previous stage. It is implemented with three different strategies, so that it can be configured to achieve its best performance depending on the extraction scenario:

- The *rule-base strategy*, that uses a set of handcrafted rules to resolve definite noun phrases and singular personal pronoun coreference.
- The *machine-learning strategy*, that uses a decision tree learned from a corpus tagged with coreferents.
- The *hybrid strategy*, that applies the first strategy to filter spurious antecedents and the second strategy to rank the remaining candidates.

In general, this process merges partial TE, TR and ST templates. The merging of the latter templates, however, involves additional knowledge which is not integrated in the *Discourse Module*.

6.1.4 Output template generation

The last module (*TemGen*) focuses on two functionalities. The first one completes the merging of partial ST templates. This is done by taking into account the consistency of the slot values in each pair of event templates, after the *Discourse Module* has resolved noun phrase coreferences. The authors of IE^2, however, explain that the integration of the process of ST template merging in the discourse analysis is necessary.

The second functionality of the module *TemGen* is the generation of the output in the desired format. It takes the SGML output of the previous module and maps it into TE, TR and ST MUC-style templates.

6.2 The PROTEUS System

Like the previous IE system, the architecture of PROTEUS is based on cascaded pattern matching. However, they differ on the level of discourse analysis. While IE^2 uses template merging procedures, PROTEUS finds logical forms and applies traditional semantic interpretation procedures. The architecture of PROTEUS is depicted in Figure 16.

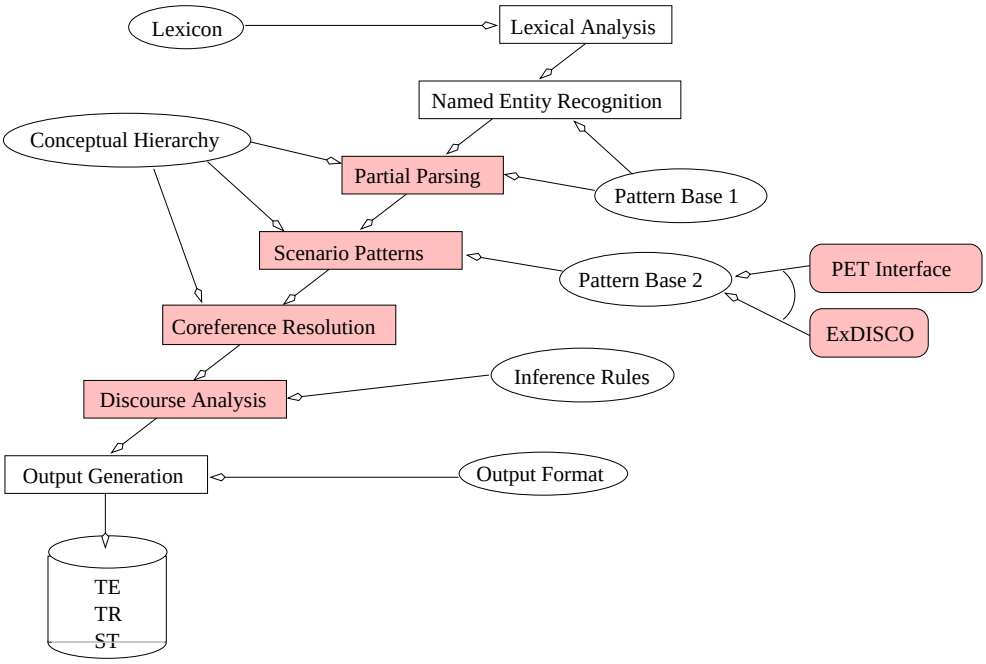

Figure 16: PROTEUS system architecture.

6.2.1 Preprocessing

First, the *Lexical Analysis* module focuses on tokenizing each sentence. This is done by using the *Lexicon* which consists of a general syntactic dictionary (COMLEX) and domain specific lists of words. Later on, the resulting tokens are POS tagged. Finally, like IE^2, the *Named Entity Recognition* module identifies proper names using a set of rules (*Pattern Base 1*).

6.2.2 Syntactico-semantic interpretation

The *Partial Parsing* module finds small syntactic phrases within sentences, such as basic NPs and VPs, and marks them with the semantic category of their heads (e.g., the class of a named entity recognized by the previous stage). Finally, similar to IE^2, the module finds appositions, prepositional phrase attachments,

and certain conjuncts by using special rules (*Pattern Base 1*) and creates logical form representations of the relations found between entities.

The *Scenario Patterns* module applies rules for clausal identification (*Pattern Base 2*). These rules create the logical forms related to those events represented in the clauses. The authors of PROTEUS consider that, the contrary to the previous stages in which domain independent rules are applied, this module uses domain specific rules. This is due to the fact that events are the most dependent information related to a specific domain. Given that is hard to hand-craft these rules, they use either the *PET Interface* (c.f., Section 5.1.1) or the *ExDISCO* learning approach (c.f., Section 5.1.2) to acquire them more easily.

Both the rules in *Pattern Base 1* and in *Pattern Base 2* contain syntactico-semantic information, with links to the concepts of the *Conceptual Hierarchy*. These concepts refer to types of slot fillers, and they are imposed by the extraction scenario and defined *a priori*.

6.2.3 Discourse analysis

As a consequence of the performance of the previous stage, the discourse analysis consists of a set of logical forms corresponding to entities, relationships and events found within each sentence. The *Coreference Resolution* module links anaphoric expressions to their antecedents. It proceeds by seeking the antecedent in the current sentence and, sequentially, in the preceeding ones until it is found. An entity within the discourse is accepted as antecedent if a) its class (in the *Conceptual Hierarchy*) is equal or more general than that of the anaphor, b) the expression and the anaphor match in number, and c) the modifiers in the anaphor have corresponding arguments in the antecendent.

The *Discourse Analysis* module, then, uses a set of inference rules to build more complex event logical forms from those explicitly described in the document. For instance, given the sentence:

"Fred, the president of Cuban Cigar Corp., was appointed vice president of Microsoft"
it is possible to infer that *"Fred"* left the *"Cuban Cigar Corp."*.

6.2.4 Output template generation

Finally, the *Output Generation* module executes another set of rules (*Output Format*) in order to translate the resulting logical forms into the MUC template structure.

6.3 The SIFT System

The architecture of SIFT is based on the application of statistical models in cascade, as shown in Figure 17. The authors explain that a better architecture would consist of a unique statistical model integrating the models corresponding to both the *Sentence level* and *Cross-sentence level* modules, since every choice can be made based on all the available information Miller et al. [1998].

6.3.1 Preprocessing

SIFT starts with the annotation of the NEs occurring in the input documents. This is the goal of *IdentiFinder*^{TM} Bikel et al. [2004], which is based on the use

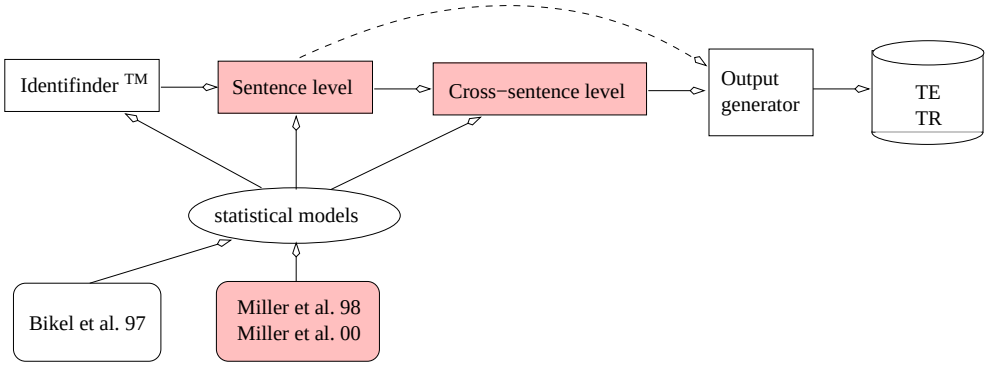

Figure 17: SIFT system architecture.

of an HMM trained to recognize the types of NEs defined in MUC tasks.

6.3.2 Syntactico-semantic interpretation

The *Sentence level* module focuses on the search of local information useful for extracting TE instances and TR instances. To do this, the module tries to find the best syntactico-semantic interpretation for each sentence using the generative statistical model trained following the approach of Miller Miller et al. [1998, 2000] (c.f., Section 5.2.2).

The *Sentence level* module explores the search space bottom-up using a chart based search. In order to keep the search tractable, the module applies the following procedures:

- When two or more constituents are equivalent, the most likely one is kept in the chart. Two constituents are considered equivalents if they have identical category labels and heads, their head constituents have identical labels, and both their leftmost modifiers and their rightmost ones have also identical labels.
- When multiple constituents cover identical spans in the chart, only those constituents with probabilities higher than a threshold are kept in the chart.

6.3.3 Discourse analysis

The *Cross-sentence level* module focuses on the recognition of possible relations between entities that occur in different sentences of the document. The module is a classifier of pairs of entities into the types of relations defined in the extraction scenario. The classifier uses a statistical model trained on annotated examples Miller et al. [1998], and only applies to pairs of entities with the following properties:

- The entities have been found by the *Sentence level* module in different sentences without taking part in any local relation.
- The types of the entities are compatible with some relation of the scenario.

6.3.4 Output template generation

Finally, SIFT applies procedures (*Output generator*) to build the output in the MUC style. On the one hand, it generates the TE instances and local TR instances from the syntactico-semantic parse trees achieved by the*Sentence level* module. On the other hand, it builds the global TR instances recognized by the module *Cross-sentence level*.

7 Conclusion

Information Extraction is now a major research area within the text-based intelligent systems discipline mainly thanks to two factors. On the one hand, there are many applications that require domain-specific knowledge, and manually building this knowledge can become very expensive. On the other hand, given the growing availability of on-line documents, this knowledge might be automatically extracted from them.

One of the main drawbacks of IE technology, however, refers to the difficulty of adapting IE systems to new domains. Classically, this task involves the manual tuning of domain-dependent linguistic knowledge, such as terminological dictionaries, domain-specific lexico-semantics, extraction patterns, and so on.

Since the early 90's, the research efforts have focused on the use of empirical methods to automate and reduce the high cost of dealing with these portability issues. Most efforts have concentrated on the use of ML techniques for the automatic acquisition of the extraction patterns useful for dealing with a specific domain, which is one of the most expensive issues. Supervised learning approaches are the most commonly applied in the state of the art. However, the task of annotating positive examples within training documents is hard, and so research is being directed at the development of less supervised learning approaches, such as those using observation-based learning or different forms of bootstrapping, and so on.

This survey describes different adaptive IE approaches that use ML techniques to automatically acquire the knowledge needed when building an IE system. It is difficult to determine which technique is best suited for any IE task and domain. There are many parameters that affect this decision, but the current evaluation framework for adaptive IE tasks do not provide sufficient data yet for performing significant comparisons.

References

J. Aberdeen, J. Burger, D. Connolly, S. Roberts, and M. Vilain. Description of the Alembic system as Used for MUC-5. In *Proceedings of the 5th Message Understanding Conference (MUC–5)*, 1993.

J. Aberdeen, J. Burger, D. Day, L. Hirschman, P. Robinson, and M. Vilain. Description of the Alembic system Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–5)*, 1995.

S. Abney. *Principle-Based Parsing: Computation and Psycholinguistics*, chapter Parsing by Chunks, pages 257–278. Kluwer Academic Publishers, Dordrecht, 1996.

Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plaintext collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, 2000.

Eneko Agirre and German Rigau. Word Sense Disambiguation Using Conceptual Density. In *Proceedings of the 16th International Conference on Computational Linguistics, COLING*, Copenhagen, Denmark, 1996.

C. Aone and W. Bennet. Evaluation Automated and Manual Acquisition of Anaphora Resolution. In E. Riloff, S. Wermter, and G. Scheler, editors, *Lecture Notes in Artificial Intelligence*, volume 1040. Springer, 1996.

C. Aone, L. Halvenson, T. Hampton, and M. Ramos-Santacruz. Description of the IE2 System Used for MUC–7. In *Proceedings of the 7th Message Understanding Conference (MUC–7)*, 1998.

D. Appelt, J. Bear, J. Hobbs, D. Israel, M. Kameyama, and M. Tyson. Description of the JV-FASTUS System Used for MUC-5. In *Proceedings of the 5th Message Understanding Conference (MUC–5)*, 1993a.

D. Appelt, J. Bear, J. Hobbs, D. Israel, and M. Tyson. Description of the jv-fastus system used for muc-3. In *Proceedings of the 4th Message Understanding Conference (MUC–4)*, 1992.

D. Appelt, J. Hobbs, J. Bear, D. Israel, M. Kameyama, and M. Tyson. Description of the FASTUS System as Used for MUC-6. In *Proceedings of the 6th Message Understanding Conference (MUC-6)*, 1995.

D. Appelt, J. Hobbs, J. Bear, D. Israel, and M. Tyson. FASTUS: A finite-state Processor for Information Extraction. In *Proceedings of the 13th International Join Conference On Artificial Intelligence (IJCAI)*, 1993b.

J.H. Aseltine. WAVE: An incremental Algorithm for Information Extraction. In *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction*, 1999.

R. Baeza-Yates and B. Ribeiro-Neto, editors. *Modern Information Retrieval.* Addison Wesley, 1999.

S. Baluja, V. Mittal, and R. Sukthankar. Applying Machine Learning for High Performance Named-Entity Extraction. In *Proceedings of the International Conference of Pacific Association for Computational Linguistics (PACLING)*, 1999.

R. Basili, M.T. Pazienza, and M. Vindigni. Corpus-driven Learning of Event Recognition Rules. In *Proceedings of the ECAI Workshop on Machine Learning for Information Extraction*, 2000.

L. Baum. An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process. *Inequalities*, 3:1–8, 1972.

D. Bikel, S. Miller, R. Schwartz, and R. Weischedel. NYMBLE: A High-Performance Learning Name-finder. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, 2004.

A. Borthwick. *A Maximum Entropy Approach to Named Entity Recognition.* PhD thesis, Computer Science Department. New York University, 1999.

A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman. Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition. In *Proceedings of the 6th ACL Workshop on Very Large Corpora.*, 1998.

Sergey Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, (EDBT'98)*, 1998.

M.E. Califf. *Relational Learning Techniques for Natural Language Information Extraction.* PhD thesis, University of Texas at Austin, 1998.

C. Cardie, W. Daelemans, C. Nédellec, and E. Tjong Kim Sang, editors. *Proceeding of the 4th Conference on Computational Natural Language Learning*, 2000.

C. Cardie and K. Wagstaff. Noun Phrase Coreference as Clustering. In *Proceedings of the joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP / VLC)*, 1999.

J. Carroll, T. Briscoe, and A. Sanfilippo. Parser Evaluation: A Survey and a New Proposal. In *Proceedings of 1st International Conference on Language Resources and Evaluation (LREC)*, pages 447–454, Granada, Spain, 1998.

N. Català. *Acquiring Information Extraction Patterns from Unannotated Corpora*. PhD thesis, Technical University of Catalonia, 2003.

N. Català, N. Castell, and M. Martín. ESSENCE: a Portable Methodology for Acquiring Information Extraction Patterns. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, pages 411–415, 2000.

J.Y. Chai and A.W. Biermann. The Use of Lexical Semantics in Information Extraction. In *Proceedings of the ACL Workshop on Natural Language Learning*, 1997.

J.Y. Chai, A.W. Biermann, and C.I. Guinn. Two Dimensional Generalization in Information Extraction. In *Proceedings of the 16th AAAI National Conference on Artificial Intelligence (AAAI)*, 1999.

B. Chidlovskii. Wrapper generation by k-reversible grammar induction. In *Proceedings of the ECAI Workshop on Machine Learning for Information Extraction.*, 2000.

H. L. Chieu and H. T. Ng. A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI).*, 2002.

H. L. Chieu, H. T. Ng, and Y. K. Lee. Closing the Gap: Learning-Based Information Extraction Rivaling Knowledge-Engineering Methods. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 216–223, 2003.

L. Childs, D. Brady, L. Guthrie, J. Franco, D. Valdes-Dapena, B. Reid, J. Kielty, G. Dierkes, and I. Sider. LOUELLA PARSING, an NL-Toolset System for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

F. Ciravegna. (LP)2, an adaptive algorithm for information extraction from Web-related texts. In *Proceedings of the IJCAI Workshop on Adaptive Text Extraction and Mining*, 2001.

W. Cohen. WHIRL: A word-based information representation language. *Artificial Intelligence*, 118:163–196, 2000.

W. Cohen, M. Hurst, and L. S. Jensen. A flexible learning system for wrapping tables and lists in html documents. In *Proceedings of the 11th International World Wide Web Conference (WWW)*, 2002.

W. Cohen and L. S. Jensen. A structured wrapper induction system for extracting information from semi-structured documents. In *Proceedings of the IJCAI Workshop on Adaptive Text Extraction and Mining*, 2001.

M. Collins. Three Generative, Lexicalised Models for Statistical Parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics.*, 1997.

C. Cox, J. Nicolson, J.R. Finkel, C. Manning, and P. Langley, editors. *Template Sampling for Leveraging Domain Knowledge in Information Extraction*, 2005. First PASCAL Challenges Workshop.

M. Craven. Learning to Extract Relations from MEDLINE. In *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction*, 1999.

M. Craven, D. Dipasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to Extract Symbolic Knowledge from the World Wide Web. In *Proceedings of the 15th AAAI National Conference on Artificial Intelligence (AAAI)*, 1998.

L. Eikvil. Information extraction from world wide web - a survey. Technical report 945, http://www.nr.no/documents/samba/research_areas-/BAMG/Publications/webIE_rep945.ps, 1999.

A. Finn and N. Kushmerick. Information Extraction by Convergent Boundary Classification. In *Proceedings of the AAAI Workshop on Adaptive Text Extraction and Mining*, 2004.

D. Fisher, S. Soderland, J. McCarthy, F. Feng, and W. Lehnert. Description of the UMass System Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

D. Freitag. *Machine Learning for Information Extraction in Informal Domains.* PhD thesis, Computer Science Department. Carnegie Mellon University, 1998a.

D. Freitag. Toward General-Purpose Learning for Information Extraction. In *Proceedings of the joint 17th International Conference on Computational Linguistics and 36th Anual Meeting of the Association for Computational Linguistics (COLING-ACL)*, 1998b.

D. Freitag and N. Kushmerick. Boosted Wrapper Induction. In *Proceedings of the ECAI Workshop on Machine Learning for Information Extraction*, 2000.

D. Freitag and A. McCallum. Information Extraction with HMMs and Shrinkage. In *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction.*, 1999.

D. Freitag and A. McCallum. Information Extraction with HMM Structures Learned by Stochastic Optimization. In *Proceedings of the 17th AAAI National Conference on Artificial Intelligence (AAAI).*, 2000.

R. Gaizauskas. Investigations into the Grammar Underlying the Penn Treebank II. Research Memorandum CS-95-25, Department of Computer Science, University of Sheffield, UK, 1995.

R. Gaizauskas, T. Wakao, K. Humphreys, H. Cunningham, and Y. Wilks. Description of the LaSIE System as Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

R. Garigliano, A. Urbanowicz, and D. Nettleton. Description of the LOLITA System as Used for MUC–7. In *Proceedings of the 7th Message Understanding Conference (MUC–7)*, 1998.

B. Glasgow, A. Mandell, D. Binney, L. Ghemri, and D. Fisher. MITA: An Information Extraction Approach to Analises of Free-form Text in Life Insurance Applications. *Artificial Intelligence*, 19:59–72, 1998.

O. Glickman and R. Jones. Examining Machine Learning for Adaptable End-to-End Information Extraction Systems. In *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction*, 1999.

G. Grefenstette, editor. *Cross-Language Information Retrieval.* Kluwer AP., 1998.

R. Grishman. Where is the syntax? In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

R. Grishman and J. Sterling. Description of the PROTEUS System as Used for MUC-5. In *Proceedings of the 5th Message Understanding Conference (MUC–5)*, 1993.

S.M. Harabagiu and S.J. Maiorano. Acquisition of Linguistic Patterns for Knowledge-Based Information Extraction. In *Proceedings of 3rd International Conference on Language Resources and Evaluation (LREC)*, 2000.

J. Hobbs. The Generic Information Extraction System. In *Proceedings of the 5th Message Understanding Conference (MUC–5)*, 1993.

R.D. Holowczak and N.R. Adam. Information Extraction based Multiple-Category Document Classification for the Global Legal Information Applications. In *Proceedings of the 14th AAAI National Conference on Artificial Intelligence (AAAI)*, pages 992–999, 1997.

C.-N. Hsu. Initial results on wrapping semistructured web pages with finite-state transducers and contextual rules. In *Proceedings of the AAAI Workshop on AI and Information Integration*, 1998.

C.-N. Hsu and N-T. Dung. Learning Semistructured Web Pages with Finite-State Transducers. In *Proceedings of the Conference on Automated Learning and Discovering*, 1998.

S. Huffman. Learning information extraction patterns from examples. In *Proceedings of the IJCAI Workshop on New Approaches to Learn for NLP*, 1995.

K. Humphreys, R. Gaizauskas, S. Azzam, C. Huyck, B. Mitchell, H. Cunningham, and Y. Wilks. Description of the LaSIE-II System as Used for MUC–7. In *Proceedings of the 7th Message Understanding Conference (MUC–7)*, 1998.

P.S. Jacobs. *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval.* Hillsdale, NJ: Lawrence Erlbaum Associates, 1992.

N. Kambhatla. Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Extracting Relations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL).*, 2004.

J. Kim and D. Moldovan. Acquisition of Linguistic Patterns for Konwledge-based Information Extraction. In *IEEE Transactions on Knowledge and Data Engineering*, 1995.

C. A. Knoblock, K. Lerman, S. Minton, and I. Muslea. A machine-learning approach to accurately and reliably extracting data from the web. In *Proceedings of the IJCAI Workshop on Adaptive Text Extraction and Mining*, 2001.

H. Ko. Empirical Assembly Sequence Planning: A Multistrategy Constructive Learning Approach. In I. Bratko R. S. Michalsky and M. Kubat, editors, *Machine Learning and Data Mining.* John Wiley & Sons LTD, 1998.

G.R. Krupka. Description of the SRA System Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

N. Kushmerick. *Wrapper Induction for Information Extraction.* PhD thesis, University of Washington, 1997.

N. Kushmerick. Wrapper Induction: Efficiency and Expressiveness. *Artificial Intelligence*, 118:15–68, 2000.

J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning (ICML).*, 2001.

A. Lavelli, M.E. Califf, F. Ciravegna, D. Freitag, C. Giuliano, N. Kushmerik, and L. Romano. IE evaluation: Criticisms and recommendations. In *Proceedings of the AAAI Workshop on Adaptive Text Extraction and Mining*, 2004.

W. Lehnert, C. Cardie, D. Fisher, J. McCarthy, E. Riloff, and S. Soderland. Description of the CIRCUS System as Used for MUC–4. In *Proceedings of the 3rd Message Understanding Conference (MUC–4)*, 1992.

W. Lehnert, C. Cardie, D. Fisher, E. Riloff, and R. Williams. Description of the CIRCUS System as Used for MUC–3. In *Proceedings of the 3rd Message Understanding Conference (MUC–3)*, 1991.

W. Lehnert, J. McCarthy, S. Soderland, E. Riloff, C. Cardie, J. Peterson, F. Feng, C. Dolan, and S. Goldman. Description of the CIRCUS System as Used for MUC–5. In *Proceedings of the 3rd Message Understanding Conference (MUC–5)*, 1993.

D. Lin. Description of the PIE System Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2): 313–330, 1993.

A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, 2000.

A. McCallum and D. Jensen. A note on the unification of information extraction and data mining using conditional-probability, relational models. In *Proceedings of the IJCAI-03 Workshop on Learning Statistical Models from Relational Data.*, 2003.

J.F. McCarthy and W.G. Lehnert. Using Decision Trees for Coreference Resolution. In *Proceedings of the 14th International Join Conference on Artificial Intelligence (IJCAI)*, 1995.

R.S. Michalski. Towards a unified theory of learning: Multistrategy task–adaptive learning. In B.G. Buchanan and D. Wilkins, editors, *Readings in Knowledge Acquisition and Learning.* Morgan Kauffman, 1993.

G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Five Papers on WordNet. *Special Issue of International Journal of Lexicography*, 3(4): 235–312, 1990.

S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schwartz, R. Stone, and R. Weischedel. Description of the SIFT System Used for MUC–7. In *Proceedings of the 7th Message Understanding Conference (MUC–7)*, 1998.

S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. A Novel Use of Statistical Parsing to Extract Information from Text. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics.*, 2000.

R. Mitkov. Robust Pronoun Resolution with Limited Knowledge. In *Proceedings of the joint 17th International Conference on Computational Linguistics and 36th Anual Meeting of the Association for Computational Linguistics (COLING-ACL)*, pages 869–875, 1998.

R.J. Mooney and C. Cardie. Symbolic Machine Learning for Natural Language Processing. Tutorial in Workshop on Machine Learning for Information Extraction. AAAI, 1999.

R. Morgan, R. Garigliano, P. Callaghan, S. Poria, M. Smith, A. Urbanowicz, R. Collingham, M. Constantino, and C. Cooper. Description of the LOLITA System as Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

S. Muggleton. Inverse Entailment and Progol. *New Generation Computing Journal*, 13:245–286, 1995.

S. Muggleton and W. Buntine. Machine Invention of First-Order Predicates by Inverting Resolution. In *Proceedings of the 5th International Conference on Machine Learning (ICML)*, 1988.

S. Muggleton and C. Feng. Efficient Induction of Logic Programs. In S. Muggleton, editor, *Inductive Logic Programming*. Academic Press, New York, 1992.

I. Muslea. Extraction Patterns for Information Extraction Tasks: A Survey. In *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction.*, 1999.

I. Muslea, S. Milton, and C. Knoblock. Hierarchical Wrapper Induction for Semistructured Information sources. *J. Autonomous Agents and Multi-Agent Systems*, 4:93–114, 2001.

I. Muslea, S. Milton, and C. Knoblock. A Hierarchical Approach to Wrapper Induction. In *Proceedings of the 3th Annual Conference on Autonomous Agents*, 2003.

V. Ng and C. Cardie. Bootstrapping Coreference Classifiers with Multiple Machine Learning Algorithms. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics.*, 2003.

M. Pasca. *Open-Domain Question Answering from Larg Text Collections.* CSLI Studies in Computational Linguistics, 2003.

L. Peshkin and A. Pfeffer. Bayesian Information Extraction Network. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003).*, 2003.

J. R. Quinlan. Learning Logical Definitions from Relations. *Machine Learning*, 5(3):239–266, 1990.

J.R. Quinlan and R.M. Cameron-Jones. FOIL: A Midterm Report. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 3–20, Vienna, Austria, 1993.

D. Radev. Text Summarization. In *Tutorial in 27st Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, 2004.

S. Ray and M. Craven. Representing Sentence Structure in Hidden Markov Models for Information Extraction. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001).*, 2001.

E. Riloff. Automatically constructing a dictionary for information extraction tasks. In *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI)*, pages 811–816, 1993.

E. Riloff. Automatically Generating extraction patterns from untagged texts. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI)*, pages 1044–1049, 1996.

D. Roth. Learning to Resolve Natural Language Ambiguities: A Unified Approach. In *Proceedings of the 15th AAAI National Conference on Artificial Intelligence (AAAI)*, pages 806–813, 1998.

D. Roth and W. Yih. Relational Learning via Propositional Algorithms: An Information Extraction Case Study. In *Proceedings of the 15th International Conference On Artificial Intelligence (IJCAI)*, 2001.

S. Sekine, R. Grishman, and H. Shinnou. A Decision Tree Method for Finding and Classifying Names in Japanese Texts. In *Proceedings of the SIG NL/SI of Information Processing Society of Japan*, 1998.

K. Seymore, A. McCallum, and R. Rosenfeld. Learning Hidden Markov Model Structure for Information Extraction. In *Proceedings of the 16th AAAI National Conference on Artificial Intelligence (AAAI).*, 1999.

M. Skounakis, M. Craven, and S. Ray. Hierarchical Hidden Markov Models for Information Extraction. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003).*, 2003.

S. Soderland. Learning to Extract Text-based Information from the World Wide Web. In *Proceedings of the 3th International Conference on Knowledge Discovery and Data Mining (KDD)*, 1997.

S. Soderland. Learning Information Extraction Rules for Semi–Structured and Free Text. *Machine Learning*, 34:233–272, 1999.

S. Soderland, D. Fisher, J. Aseltine, and W. Lehnert. CRYSTAL: Inducing a conceptual dictionary. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1314–1321, 1995.

T. Strzalkowski, editor. *Natural Language Information Retrieval.* Kluwer, 1999.

A. Sun, M. Naing, E. Lim, and W. Lam. Using Support Vector Machines for Terrorism Information Extraction. In *Proceedings of the First NSF/NIJ Symposium on Intelligence and Security Informatics (ISI 2003)* , pages 1–12, 2003.

K. Takeuchi and N. Collier. Use of Support Vector Machines in Extended Named Entities. In *Proceedings of the IV Conference on Computational Natural Language Learning (CoNLL).*, 2002.

B. Thomas. Anti-Unification Based Learning of T-Wrappers for Information Extraction. In *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction*, 1999.

C.A. Thompson, M.E. Califf, and R.J. Mooney. Active Learning for Natural Language Parsing and Information Extraction. In *Proceedings of Sixteenth International Machine Learning Conference*, pages 406–414, 1999.

J. Turmo. *An Information Extraction System Portable to New Domains.* PhD thesis, Technical University of Catalonia, 2002.

J. Turmo and H. Rodríguez. Learning Rules for Information Extraction. *Natural Language Engineering. Special Issue on Robust Methods in Analysis of Natural Language Data.*, 8:167–191, 2002.

M. Vilain. Inferential Information Extraction. In M.T. Pazienza, editor, *Information Extraction: Towards Scalability, Adaptable Systems.*, volume 1714. Springer–Verlag, Berlin, 1999.

R. Weischedel. Description of the PLUM System as Used for MUC–6. In *Proceedings of the 6th Message Understanding Conference (MUC–6)*, 1995.

R. Weischedel, D. Ayuso, S. Boisen, H. Fox, H. Gish, and R. Ingria. Description of the PLUM System as Used for MUC–4. In *Proceedings of the 6th Message Understanding Conference (MUC–4)*, 1992.

R. Weischedel, D. Ayuso, S. Boisen, H. Fox, R. Ingria, T. Matsukawa, C. Papageorgiou, D. MacLaughlin, M. Kitagawa, T. Sakai, L. Abe, H. Hosihi, Y. Miyamoto, and S. Miller. Description of the PLUM System as Used for MUC–5. In *Proceedings of the 6th Message Understanding Conference (MUC–5)*, 1993.

R. Weischedel, D. Ayuso, S. Boisen, R. Ingria, and J. Palmucci. Description of the PLUM System as Used for MUC–3. In *Proceedings of the 6th Message Understanding Conference (MUC–3)*, 1991.

R. Yangarber. *Scenario Customization of Information Extraction.* PhD thesis, Courant Institute of Mathematical Sciences. New York University, 2000.

R. Yangarber. Counter-Training in Discovery of Semantic Patterns. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics (ACL).*, 2003.

R. Yangarber and R. Grishman. Description of the PROTEUS/PET System as Used for MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC–5)*, 1998.

R. Yangarber and R. Grishman. Machine Learning of Extraction Patterns from Unnanotated Corpora: Position Statement. In *Proceedings of the ECAI Workshop on Machine Learning for Information Extraction*, 2000.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. Automatic acquisition of domain knowledge for information extraction. In *In Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, 2000.

D. Yarowsky. Bootstrapping Multilingual Named-Entity Recognizers. In *Proceedings of the ACL Workshop on Multilingual and Mixed-language Named Entity Recognition.*, 2003.

S. Young and G. Bloothooft, editors. *Corpus-Based Methods in Language and Speech Processing.* Kluwer Academic Press, 1997.

D. Zelenko, C. Aone, and A. Richardella. Kernel methods for Relation Extraction. *Journal of Machine Learning Research*, 3(2003):1083–1106, 2003.

J.M. Zelle and R. J. Mooney. Inducing Deterministic Prolog Parsers from Treebanks: A Machine Learning Approach. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI)*, pages 748–753, 1994.

S. Zhao and R. Grishman. Extracting Relations with Integrated Information Using Kernel Methods. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 419–426, 2005.