Examining Machine Learning for Adaptable End-to-End Information Extraction Systems

Oren Glickman
Math and Computer Science Department
Bar Ilan University
Ramat Gan, Israel
glikmao@cs.biu.ac.il

Rosie Jones
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA USA
rosie@cs.cmu.edu

Abstract

All components of a typical IE system have been the object of some machine learning research, motivated by the need to improve time taken to transfer to new domains. In this paper we survey such methods and assess to what extent they can help create a complete IE system that can be easily adapted to new domains. We also lay out a general prescription for an IE system in a new domain, employing existing components and technologies where possible. The goal is a system that can be adapted to a new domain with minimal human intervention (say by someone who may be a domain expert but need not be a computational linguist). We propose research directions for automating the process further, reducing the need for hand-tagged training data by relying on biases intrinsic to the information extraction task, and employing boot-strapping and active learning.

Introduction

Information extraction (IE) is a process that takes unseen texts as input and produces fixed-format unambiguous data as output, as typified by the Message Understanding Conference (MUC) evaluations.

Building an information extraction system in a new domain is difficult and time consuming, often requiring months of effort by domain specialists and computational linguists. This is mainly due to the domain specific nature of the task (Cardie 97). To address this problem of portability, recent research has focused on applying empirical methods throughout the IE process. Indeed IE is an attractive prospect for the application of machine learning, since increasing availability of online texts encourages construction of data-intensive approaches which profit from some labeled data, and larger pools of unlabeled data. Traditional information extraction has consisted of the end-to-end concatenation of components, where it is assumed that separate research on these parts is a semi-solved problem and that the accuracy of individual pieces will not affect the whole too badly. A naive application of machine learning would apply to the individual pieces, leaving the overall set-up unchanged. A more sophisticated change would reflect trends in other areas of artificial intelligence including speech recognition and machine translation, which have seen the overhaul of the solution architecture. In the next sections we will discuss how component-by-component integration of machine learning technologies has been applied to information extraction. These methods of automating components employ domain independent methods and resources to varying degrees. We will discuss how each furthers the overall goal of system adaptability and portability.

In the final section we will discuss how an end-to-end portable information extraction system could be constructed; which components can be taken off the shelf, which should be modeled as part of the integrated whole, and where the borders between modules could be drawn to provide helpful bias for the overall system, while allowing task-specific components to inform one another. We will describe what data and techniques are necessary in moving to a new domain, given existing machine learning techniques, and in the ideal system.

Application of Machine Learning to the Components of IE

Information extraction systems can be classified into those operating on (a) structured texts (such as web pages with tabular information), (b) semi-structured texts (such as online personals) and (c) free text (such as news articles). In this paper we focus primarily on the more difficult problem of extraction from free text.

Though IE systems vary in their architecture, a typical IE system for extraction from free text is built from the following components (Cowie and Lehnart 96, Cardie 97, Appelt 97, Grishman 96):

- A *filtering* component (domain dependent),
- *linguistic processing* including a tokenization step, morphological and lexical processing, and syntactic analysis (all generally assumed to be domain independent) ,
- *semantic tagging* (arguably domain dependent),
- *scenario pattern matching*, *discourse analysis* and *template generation* (domain dependent).

The components above will be executed roughly in the order stated and might use the output of earlier stages. In the following sections we discuss how Machine Learning

(ML) methods have been applied and how they fit in or contribute to an adaptable IE system.

Filtering

Filtering (Lewis 92) determines the relevance of the text or parts of the text. This component filters text that will not create a template or be processed by the subsequent stages (Cowie et al 94). Such a filtering component is useful and sometimes essential if the system gets its input from an IR system or from a stream of data. In the case of long texts, relevant information may appear only in a few paragraphs. Although the main purpose of this component is of speed-up, an increase in IE performance has also been reported.

Systems differ in the point at which filtering is done, either before or after significant linguistic analysis is done. The tradeoff is clear – the earlier filtering is done the more can be gained from it in terms of system speed-up and improvement in precision, but the greater the risk of reducing system coverage.

Note that though this process is similar to information retrieval (IR), it is not identical. Usually it is used to filter documents retrieved by an IR or spidering system. By definition the filtering component is domain-specific. On the other hand, it requires only the labeling of whole documents or sections of documents, which is easy for the novice user. It also lends itself to unsupervised and semi-supervised learning (see for example Nigam et al 98), which means that rapid deployment in a new domain requires only labeling a small set of initial training documents.

Linguistic Processing

In this section we address linguistic processing components, which have been viewed as more-or-less domain independent. Hence little research has been conducted on applying ML techniques to adapt them to new domains in the context of IE.

Tokenization: locates word and sentence boundaries. Input text is divided into smaller units – headings, paragraphs, sentences and tokens. Tokens could be words, symbols, constructs or idiomatic phrases that appear in the dictionary. The algorithm may take into account special formatting such as HTML. This is usually a set of rule-of-thumb techniques with no machine learning applied. If the system gets inputs in similar formats across domains (streams of Reuters news articles, for example) there is no need to adapt this component. On extraction systems that take special formatting cues into account see the Wrapper Induction section under Scenario Pattern Matching. There is a body of research into automatic word segmentation for Asian languages (for example Matsumoto et al 97). Speech transcriptions have also been the object of research in finding punctuation, sentence boundaries and document structure (Beeferman et al, 98, 99).

In general, tokenization systems can be applied in a new domain without much modification. Special cases do exist, such as HTML mark-up, and speech recognizer output without capitalization and other cues present in written texts, but these form a small set which expands only when a new technology makes a new type of text available.

Morphological and Lexical Processing: determines word properties by looking them up in a lexicon. A part-of-speech tagger is usually applied to compute the most likely tag based on the context in which it occurs. These tags might be used by semantic-structure finding components downstream, but note the existence of named entity extractors such as Nymble (Bikel et al 97) which we will describe in the next section, which side-step part-of-speech tagging altogether.

There has been much work in corpus-based morphological analysis such as part-of-speech tagging (Brill 92, Church 88, Kupiec 92, and others). Most IE systems that use morphological information do use components that were trained using a machine learning algorithm. However the corpus used is a tagged one (such as the Wall street Journal or Brown Corpus) and might be very different from texts the IE system is supposed to process. Most systems assume that part-of-speech tagging is domain independent enough and that the high performance achieved from current taggers is sufficient. In the case of a standard Hidden Markov Model (HMM) based implementation, it is most probably the case that the markovian dependence of a tag given previous tags is the same across domains of similar style. But it is definitely not the case that the distributions of possible tags for each word are similar among different domains. There is no need to tag each domain text with morphological information in such a system, because an Expectation Maximization (EM), or Baum-Welch, algorithm could be applied to achieve an unsupervised adaptation of the tagger to a new domain. It is an interesting question as to how many errors in current IE systems are due to errors in the morphological analysis and if such domain adaptation could reduce their number.

Syntactic Analysis: discovers sentence level structure required by the information extraction task. Identifies syntactic constructs such as noun phrases, and may identify subject and objects of a verb. A detailed parse tree may not be needed, and most systems suffice with just partial parsing (Grishman et al 95).

Though there has been work on automatic grammar induction, current IE systems tend to use handcrafted grammars. A robust syntactic parser that infers parses for unknown words and structures can be applicable across domains. Another promising direction for out-of-the-box information extraction systems is semantic grammars, which can be augmented by novice users, such as SOUP (Gavalda and Waibel 98).

Semantic Processing: Named Entities, and Others

The semantic processing component of an information extraction system comprises tasks such as word sense tagging, name recognition and identification, and other

semantic tagging. This component might for example identify company names, locations and identify and classify various products. In MUC this was assigned as a sub-task referred to as Named Entity (NE) identification (and classification), for which a tagged corpus was given as training data. The following systems learn to identify and classify such entities automatically from a training corpus:

Nymble(BBN) (Bikel et al. 97) This system uses training data tagged with the presence of names, and their types, to build a hidden markov model (HMM) which can detect the presence of name types in unlabeled test data. It uses both word-identity, and word-features (capitalization, contains-digits, etc). An accuracy of over 90% for both English and Spanish, in both mixed-case and capitalized texts is reported. It is important to note that these results rely only on the presence of labeled training data; they do not require lists of names. The system's accuracy can be achieved with 100,000 words of training data. Note that labeling this amount of training data is a reasonable burden in switching to a new domain. However, it would be interesting to see this approach trained from a very small amount of labeled data and allowed to complete its learning using expectation maximization (EM).

MENE(NYU) (Borthwick et al 98). This system uses many word-features similar to those in the Nymble system described above, as well as incorporating dictionaries where available. The maximum entropy approach effectively allows probabilistic incorporation of many possibly correlated information sources. This system relies on hand-labeled training sets, but can use the output of other systems (with errors which may be uncorrelated) to bootstrap to higher performance.

At NYU, decision trees were used for Japanese named entity extraction (Sekine et al 98). This system relies on Japanese tokenization and part-of-speech tagging, and dictionaries for semantic classes (including organizations, proper names of people, locations and so forth). Rules over word-tokenization and dictionary features, as well as word identity are learned from labeled training data using a decision tree. The authors demonstrate the domain independence of this approach by showing that the same techniques and dictionaries can be adapted to a new domain by retraining the tree on new labeled data, without the need to modify the underlying tokenization and dictionary components. New labeled training data is required for the new domain however, as cross-domain train-test conditions show degraded performance, even while the target semantic classes (such as names) remain unchanged.

The greatest weakness of these approaches, in terms of easy domain transfer, is that they include word-identity as one of the features. This means that statistics must be compiled over individual words, necessitating large amounts of labeled training data. To the extent that word-classes can be discovered from a small amount of hand-labeling, these systems will be more applicable. This could take the form of task-independent but document-collection-dependent clustering (Ushioda 96), or learning semantic classes by bootstrapping (Riloff and Jones 99).

An important observation is that the named entity task as defined in MUC has a fixed set of seven classes to be identified – person names, organizations, locations, dates, times, monetary values and percentages. Though these classes are very general and were important for the Management Succession task, they tend to be easily captured by information in the local context and rely strongly on word-features. For other tasks different entities would be of particular importance. It would be interesting to examine the performance of the systems described above on other sets of semantic classes.

Scenario Pattern Matching

This task consists of identifying domain specific relations among relevant entities in the text, and is basically the only component of an IE system that is unique to the task of information extraction. IE systems require a separate set of rules for each domain, making it a very domain specific task. Machine learning is therefore an attractive option (see Muslea 98, Cardie 97, Soderland 99). Following are a few of the systems that automatically identify domain specific extraction rules.

Autoslog (Riloff 93) acquires extraction patterns from training examples. Autoslog operates over syntactically bracketed fields (verb, subject, object, etc.) and assumes that entities are semantically tagged. Autoslog learns extraction patterns in the form of domain-specific semantic case frames called concept nodes with a maximum of one slot per frame. Each rule is associated with a 'trigger' which is the phrase that activates the pattern. As a training corpus it requires a set of texts and their answer keys and depends on the existence of a partial parser, a small lexicon with domain-specific semantic class information and a set of more-or-less domain independent linguistic patterns. Any proposed extraction pattern is presented to a person for acceptance or rejection. The extraction rules are simple enough for a domain expert who is not a computational linguist to understand. It is important to note that Autoslog learns a rule per slot of training set templates in a one shot manner. Autoslog has no automatic induction step – all output is passed to a human and is not passed back to the learning algorithm.

CRYSTAL (Soderland et al. 95), like AutoSlog, learns extraction patterns in the form of semantic case frames or *Concept Nodes* (CNs). It uses a covering algorithm to learn extraction patterns. First it creates a set of CN definitions for each positive training instance from the training corpus. It then gradually relaxes the constraints on these initial definitions to broaden their coverage, while merging similar definitions to form a more compact dictionary. The CN definitions in CRYSTAL's final dictionary are generalized as much as possible without producing extraction error on the training corpus. CRYSTAL unifies two similar definitions by finding the most restrictive constraints that cover both.

PALKA (Kim & Moldovan 95) learns extraction patterns that are similar in form to Autoslog's concept nodes. A pattern consists of a set of possible trigger verbs and generic semantic case frame definitions from a concept hierarchy of semantic classes. To learn extraction patterns, each new training instance could cause a rule to either be generalized moving up in the semantic hierarchy of specialized by replacing a semantic class with some of its children.

LIEP (Huffman 95): rather than learning one extraction pattern for each slot in a training template, generates a single rule for all slots. LIEP does not rely on a tagged training set, but rather allows a user to interactively identify events in texts. For each sentence of a training text given by the user, entities of interest (e.g. people, companies and titles) are identified and the user can then choose which combinations of the entities signify events to be extracted. LIEP tries to build a set of extraction patterns that will maximize the number of extractions of positive examples and minimize spurious extractions. Given a new example that is not already matched by a known pattern, LIEP first attempts to build a new pattern based on the example. LIEP searches for syntactic relationships between each pair of constituents by performing a depth-first algorithm.

HASTEN (Krupka 95) uses a class of extraction patterns called Egraphs that contain semantic labels paired with constraints such as semantic class, verb root, verb voice or exact word. Weights for each pattern template are learned in a training phase from positive examples. In the extraction phase, HASTEN uses a similarity metric to compare an Egraph with the input text.

RAPIER (Califf & Mooney 97) is a system for semi-structured text that uses a form of Inductive Logic Programming and requires no prior syntactic analysis. Induces rules from a tagged corpus of texts with tier target templates. The text is considered to have three fields – the target field itself, a pre-filler of token before the target phrase and a post-filler of tokens after it. RAPIER's rules specify an ordered list of lexical, semantic and morphological constraints to be matched for each of these fields. The Inductive Logic Programming (ILP) algorithm starts with the most specific rule that matches each target slot from the training. It then pairs randomly chosen rules relaxing them taking their least general generalization (LGG) and adding constraints, if needed to operate correctly on the training set. This process is repeated until no progress is made.

WHISK (Soderland 99) is possibly the most general rule extraction system to date, using regular expression like extraction patterns. It is not restricted to specific pre-processing of the text and hence good for structured as well as semi-structured and free text. WHISK uses supervised learning inducing a set of rules from hand-tagged training examples. To minimize the human effort involved, active learning is employed by interleaving the annotation of the training data with the presentation instances that are likely to be near the decision boundaries of the system. WHISK rules are based on a form of regular expression patterns that identify the context of relevant phrases and the exact delimiters of those phrases. Predefined domain-specific semantic classes are used, and when applied to free text, the text is also segmented into syntactic fields. WHISK uses a covering algorithm inducing rules top-down, by first finding the most general rule that covers the seed and then extending the rule by adding terms one at a time as long as it is below a certain threshold of errors.

SRV (Freitag 98), targeted for extraction from semi-structured text, converts the extraction problem by considering all possible phrases as potential slot fillers. The system adopts a multi-strategy approach and combines evidence from three classifiers: a rote learner, a naive Bayes classifier and a relational rule learner. The rote learner simply compares the phrase to a list of all correct slot fillers found in training. The naive Bayes classifier computes an estimated probability that the tokens in the phrase are found in a correct slot filler. The relational learner induces in a top-down manner a set of constraints such as the length of the phrase or existence of words in or near it.

Note that all systems designed to handle free text depend on prior syntactic processing and semantic labeling.

Wrapper Generation (Kushmerick et al. 97, Ashish & Knoblock 97, Hsu & Dung 98) Independently of the traditional IE community, systems that learn to transform one or multiple web pages into the equivalent of database entries have emerged. Though such systems can not handle information extraction from free text some of the ideas and techniques used here could be incorporated as in the lower level components such as tokenization.

Automatic Template Deduction

All the systems described above perform supervised learning of extraction rules from annotated texts. The user is assumed to have an already developed set of templates. Recently, others have addressed the problem of generating such patterns automatically without the need for annotation.

AutoSlog-TS (Riloff & Shoen 95) is an extension of Autoslog. A user needs only to classify texts into relevant vs. not relevant. The system generates an extraction pattern for every phrase in the training corpus. It then evaluates the extraction patterns by processing the corpus a second time and generating relevance statistics for each pattern. The patterns are then ranked in order of importance to the domain. The user hand-filters the best patterns.

Other ongoing research (Collier 98, Nobata & Sekine 98, Pierce 98) further extends this approach, trying to automatically induce pertinent entities and relationships between them from a set of relevant texts. More recent work focuses on customizing templates by user interests. (Karkaletsis et al. 97, Yangarber & Grishman 97). This new trend is an interesting extension to the traditional IE paradigm in which a pre-defined template is assumed.

Discourse Processing and Template Generation

Discourse Processing. In this stage, coreference and anaphora resolution are done. Coreference resolution is responsible for knowing when different descriptions refer to the same entity. A text might contain, for example, the expressions: 'Bill Clinton', He, 'William J. Clinton' and 'The president' all referring to the same person. A discourse model will examine each entity encountered in the text and determined whether it is new and should be added to the discourse model.

While there have been previous investigations of empirical approaches to coreference outside of the scope IE scope (Dagan et al 95, Anone and Bennett 95), these have generally centered on the task of assigning correct references for anaphoric expressions and assume full parsed text as input. Because coreference resolution is crucial to the success of an IE system it was designated as another sub-task of MUC. A corpus with tagged entities was created for training for participating sites. Given all marked entities in a text, coreference resolution can be performed by training.

RESOLVE (McCarthy & Lehnert 95) and **MLR** (Aone & Bennett 95) both used an annotated corpus to train decision trees that decide whether two phrases refer to the same object. In both systems the attributes or their values might be domain dependent and therefore the system needs to be retrained for each domain for good results.

Template Generation. Assembles events, entities and relationships between them into a pre-specified template structure. In some cases set fillers are desired (human/non-human, for example) or normalization of fillers is needed. In **Wrap-Up** (Soderland & Lehnart 94) decision tree are used to handle the merging and co-reference simultaneously. Another system (Kehler 97) uses a probabilistic method to do the same, and in later work (Kehler 98) applies unsupervised and weakly supervised techniques to the task, albeit with little increase in performance. While one technique applied attempted to find an optimal distance metric for clustering, and still did not increase performance, the class of features considered for the distance metric may have lead to this result. Nevertheless this opens the way to the types of machine learning we see as most applicable to rapid-deployment IE systems.

Automating the Whole

As has been shown in the previous sections, all components of an IE system have been more or less automated. The remaining problem of a rapid shift to a new domain consists of putting all the pieces together, with minimum additional work.

It is useful for us to note that some machine learning techniques have been found effective for multiple components in an IE system, as follows:

- HMMs have been used for part-of-speech tagging and named entity extraction
- ILP for parsing (Zelle & Mooney 94), scenario pattern matching and coreference resolution.
- Decision trees for named entity extraction, parsing (Magerman 95) and coreference resolution.
- Maximum entropy for part-of-speech tagging and shallow parsing (Skut and Brants 98) and for named entity extraction.

This suggests a methodology in which a single learning mechanism is applied simultaneously or sequentially to multiple components in the system. We can be motivated in taking this tack by examining trends in other areas of AI. Complex tasks have been accomplished at first by identifying the pieces that need to be addressed, plugging those pieces together in a knowledge-intensive ad-hoc way, and later unifying the end-to-end process.

IE for the Commercial User: Our goal is a system for information extraction which novice end users can quickly train for their own needs through ongoing interaction.

As suggested throughout this paper, current portable techniques rely heavily on annotated texts in the new domain. In order to meet the goal of system adaptability, researchers in IE should investigate further ML algorithms that rely on only small amounts of labeled training data. This can be addressed with:

1) Methods which reduce the amount of labeled training data required, reducing human annotation time:
   - More fully unsupervised methods (which identify properties of text of new domain)
   - More weakly supervised methods (bootstrapping from small sets of annotated data).
   - Active Learning/Selective Sampling (use human time judiciously, once the maximum underlying structure has been identified).
2) Methods which allow the target extraction in the new domain to be used to increase accuracy at all levels simultaneously, and which allow components to operate at the maximum efficacy as an end-to-end set
   - Indirectly supervised methods with the end-task driving learning of lower-level components
   - Combination of learners of different types
   - Tightly coupling components by allowing multiple hypotheses from one be the input to the next, and even allowing them to iteratively learn from one another.
3) Online Algorithms (so the system keeps adapting after deployment).

By combining the training of components as we suggested in the previous section, results should improve over-all.

The Biases We Need

Weakly supervised information extraction systems could be built to operate out-of-the-box by assuming the following:

1. Our semantic classes can be described by giving a few prototypical exemplars – these can be used as seeds for clustering or boot strapping of the whole semantic class.
2. Relationships between semantic entities in the target domain – a small set of semantic classes and relationships can be postulated for template construction (eg Huffman 95).
3. Co-ocurrence of coreferring expressions in reasonably close proximity – nearby occurrences of the same semantic class may well correspond to the same entity; these are good candidates for a system which hypothesizes coreference rules.
4. Documents can be tagged as relevant or irrelevant to the domain – these can be used to generate a large corpus of relevant and irrelevant texts. Features that distinguish these two types of text may be good candidates for extraction patterns (e.g. Riloff 95).

These biases allow weakly supervised methods to limit their search space, and still generate reasonable results. The extent to which these assumptions fail in a particular new domain will determine the ability of our out-of-the-box methodology to be adapted to the new domain.

Information Extraction for a New Domain – Today and Tomorrow

Appelt and Israel (97) and Cunningham et al (97) describe methodologies for constructing complete information extraction systems. However, they do not address the full range of machine-learning and hence rapidly deployable techniques available now, and which could be built given current research knowledge.

Given the state-of-the-art for automated components for an information extraction system described in previous sections, we are now in a position to enumerate the steps necessary for transition to a new domain, while using existing data and technology as much as possible. We will address this both as a practical description of what is currently feasible, and the ideal, which we would like to see come to fruition.

Today:

Data Required:

1) Texts from new domain (100,000+ words)
2) publicly available dictionaries

Technology, Code Required:

1) Domain independent linguistic processing components (part-of-speech tagger, trained on publicly available corpus, shallow syntactic parser).
2) Learner(s) (e.g. HMM, decision tree, inductive and covering rule learners)

Steps Required:

1) Label named entities in domain-specific texts.
2) Fill in templates for domain specific texts.
3) Tag coreferring examples in domain specific texts.
4) Train named-entity, semantic classes using rule-builder and labeled text.
5) Tag and parse training data, incorporating named entity finding from step 4.
6) Use learner to construct extraction patterns.
7) Train coreference and anaphora resolution component, using learner and labeled training data (including output from previous stages).

Tomorrow:

Data Required:

1) NewText - Texts from new domain (100000+ words)
2) Publicly available dictionaries

Technology, Code Required:

1) Part-of-speech tagger, trained on publicly available corpus.
2) Syntactic Parser (shallow).
3) Rule-builder(s) (e.g. HMM, decision tree, covering rule learner, regular expression constructor).
4) Maximum entropy system.
5) Clustering code.
6) Active-learning, boot-strapping wrapper for learners.
7) Information Extraction Toolkit

Steps Required:

1) Provide 10 or so examples of each semantic class.
2) Label a small number of texts as relevant or irrelevant.
3) Watch IE toolkit while it:
   - Uses named entity examples as seeds for vocabulary clustering.
   - Uses labeled documents as seeds to build up large clusters of relevant and irrelevant documents.
   - Iteratively builds up semantic classes which distinguish relevant and irrelevant texts, and identifies domain-specific concepts.
   - Constructs candidate templates containing target semantic concepts provided as examples, and presents them to the user for verification using active learning.
   - Finds entities from the same semantic class occurring close to one another, and presents them to the user to classify to aid in active learning of coreference resolution.
   - Makes the tea, and writes the research paper.

Conclusion

We have outlined state-of-the-art research in machine learning applications to information extraction, and summarized how these can be tied together to port to a new domain. We have emphasized the dependence of these approaches on supervised learning from labeled data, and have outlined the biases which would enable more weakly supervised methods to perform the task with much less user involvement and expertise. A research agenda following these lines will develop information extraction

systems which can be rapidly deployed in new domains, and which will require only the minimum of human assistance for successful adaptation.

Acknowledgments

We would like to thank Ido Dagan, Zvika Marx and Yuval Krymolowski for their helpful feedback on this paper. Rosie Jones' research is supported in part by the DARPA HPKB program under contract F30602-97-1-0215, and by the National Science Foundation under grant SBR-9720374.

References

Aone, C.; Bennett, S. W. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In Proceedings of the *33rd Annual Meeting of the Association for Computational Linguistics*, 122-129. Cambridge, MA.

Appelt, D. E.; Israel D. 1997. Building Information Extraction Systems. *ANLP-97 Tutorial*.

Ashish, N.; Knoblock, C. 1997. Wrapper Generation for semi-structured Internet sources. *SIGMOD 26(4):8-15.*

Bikel, D.; Miller, S.; Schwartz, R.; Weischedel, R. 1997. NYMBLE: a High-Performance Learning Name-finder. In proceedings of the *Second Conference on Empirical Methods in Natural Language Processing*, Providence, RI, 109-116.

Beeferman, D.; Berger, A.; and Lafferty, J, 1999 Statistical models for text segmentation *Machine Learning* 34, Special Issue on Natural Language Learning (Cardie, C. and Mooney R. Eds)

Beeferman, D.; Berger, A.; and Lafferty, J, 1998 Cyberpunc: a lightweight punctuation annotation system for speech, IEEE conference on acoustic, speech and signal processing, Seattle, WA 1998

Borthwick, A.; Sterling, J.; Agichtein, E.; Grishman, R. 1998. Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition. In Proceedings of the *Sixth Workshop on Very Large Corpora*, Universite de Montreal, Montreal, Canada..

Brill, E. 1992. A simple rule based part-of-speech tagger. In *Third Conference on Applied Natural Language Processing*, 152-155, Trento, Italy.

Brown, P.; Cocke, J.; Della Pietra, A; Della Pietra, V; Jelinek, F.; Lafferty, J.; Mercer, R.; Roossin, P. 1990. A Statistical Approach to Machine Translation

Califf, M. E.; Mooney, R. J. 1997. Relational Learning of Pattern-Match Rules for Information Extraction. *In Proceedings of the ACL Workshop on Natural Language Learning, 9-15*. Somerset, NJ: Association for Computational Linguistics.

Cardie, C. 1997. Empirical Methods in Information Extraction. *AI Magazine* 39(1):65–79.

Cardie, C. 1993. A Case-Based Approach to Knowledge Acquisition for Domain-Specific Sentence Analysis. In Proceedings of the *Eleventh National Conference on Artificial Intelligence*, 798-803. Menlo Park, CA: American Association for Artificial Intelligence.

Church, K. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In proceedings of *ANLP*, 136-143. Morristown, NJ: Association for Computational Linguistics.

Collier, R. 1998. Automatic Template Creation for information Extraction, an Overview. Ph.D. Proposal, University of Sheffield.

Cowie, J; Wakao, T.; Guthrie, L.; Jin, W.; Pustejovsky, J.; Waterman, S. 1994. The Diderot Information Extraction System *PACLing-94*

Cowie, J.; Lehnert, W. 1996. Information Extraction. *Communications of the ACM* 39(1):80–91.

Cunningham, H.; Humphreys, K.; Wilks, Y.; Gaizauskas, R. 1997 Software Infrastructure for Natural Language Processing *ANLP-97*

Dagan, I.; Justenson, J.; Lappin, S.; Leass, H.; Ribak, A. 1995. *Syntax and Lexical statistics in anaphora resolution. Applied Artificial Intelligence*, 9(6):633-644.

Freitag, D. 1998. Multistrategy Learning for Information Extraction. *ICML-98*.

Grishman, R. 1995. The NYU system for MUC-6 or where's the syntax? In Proceedingsof the *Sixth Message Understanding Conference*. San Fransisco, CA: Morgan Kaufmann.

Grishman, R. 1997. Information Extraction: Techniques and Challenges. *SCIE-97* 10–27.

Hsu, C.; Dung, M. 1998. Wrapping Semistructured Web Pages with Finite State Transducers. *CONALD-98.*

Huffman, S., B. 1995; Learning Information Extraction Patterns from Examples. *IJCAI-95 workshop on New Approaches to Learning for Natural Language Processing*.

Gavalda, M; and Waibel, A. 1998 Growing Semantic Grammars *COLING/ACL-98*

Karkaletsis, V.; Spyropoulos, D.; Benaki, E. 1997. Customising Information Extraction Templates according to Users Interests. In Proceedings of the *International Workshop on Lexically Driven Information Extraction*, Frascatti, Rome.

Kehler, A. 1997. Probabilistic Coreference in Information Extraction. In proceedings of *The Second Conference on Empirical Methods in Natural Language Processing*, 163-173. Somerset, NJ: Association for Computational Linguistics.

Kehler, A. 1998. Learning Embedded Discourse Mechanisms for Information Extraction. In proceedings of *AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*. CA.

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

Kupiec, J. 1992. Robust part-of-speech-tagging using a hidden markov model. *Computer Speech and Language*, 6:225-242.

Krupka, G. 1995. Description of the SRA system as used for MUC-6. *In Proceedings of the sixth Message Understanding Conference*, 221-236. San Fransisco, CA: Morgan Kaufmann.

Kushmerick, N.; Weld, D.; Doorenbos, R. 1997. Wrapper Induction for Information Extraction. *IJCAI-97*.

Lewis, D. D. 1992. Text Filtering in MUC-3 and MUC-4. In Proceedings of the *Fourth Message Understanding Conference*, 51-66. Morgan Kaufmann, San Mateo, CA.

Magerman, D. M. 1995. Statistical decision-tree models for parsing. In Proceeding of the 33rd Annual Meeting of the *Association for Computational Linguistics*, 276-283. Cambridge MA.

McCarthy, J.; Lehnert W. 1995. Using Decision Trees for Coreference Resolution. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*.

Muslea, I. 1998. Extraction Patterns: from Information Extraction to Wrapper Generation. Unpublished.

Matsumoto, Y.; Kurohashi, S.; Yamaji, O.; Taeki, Y. and Nagao, M. 1997 Japanese morphological analyzing System: JUMAN *Kyuoto University and Nara Institute of Science and Technology*

Nigam, K.; McCallum, A.; Thrun, S.; Mitchell, T. 1998. Learning to Classify Text from Labeled and Unlabeled Documents *AAAI-98*

Nobata, C.; Sekine, S. 1998 Automatic Acquisition of Patterns for Information Extraction, unpublished.

Pierce, D. 1998. An Interactive Information Extraction Environment. Unpublished.

Riloff, E. 1993. Automatically Constructing a Dictionary for Information Extraction Tasks. *AAAI-93*.

Riloff, E.; Shoen, J. 1995. Automatically Acquiring Conceptual Patterns Without an Annotated Corpus. *Proceedings Third Workshop on Very Large Corpora*.

Riloff, E.; Jones, R. 1999. Learning Dictionaries for Information Extraction by Multi-level Boot-strapping *AAAI-99* (forthcoming)

Sekine, S.; Grishman, R.; Shinou, H. 1998. A Decision Tree Method for Finding and Classifying Names in Japanese Texts Sixth Workshop on Very Large Corpora, Canada

Skut, W.; Brants, T. 1998. A maximum-entropy partial parser for unrestricted text, In Proceedings of *the Sixth Workshop on Very Large Corpora*, Montreal, Canada.

Soderland, S.; Fisher, D.; Aseltine, J.; Lehnert, W. 1995. Crystal: Inducing a Conceptual Dictionary. *IJCAI-95*.

Soderland, S. 1999. Learning Information Extraction Rules for Semi-structured and Free Text. *Machine Learning*.

Ushioda, A. 1996 Hierarchical Clustering of Words and Application to NLP Tasks *Proceedings of the 4th Workshop on Very Large Corpora*

Yangarber, R.; Grishman, R. 1997. Customization of Information Extraction Systems. In Proceedings of the *International Workshop on Lexically-Driven Information Extraction*, Frascati, Italy.

Zelle, J.; Mooney, R. 1994. Inducing Deterministic Prolog Parsers from Tree Banks: A Machine-Learning Approach. In Proceedings of the *Twelfth National Conference on Artificial Intelligence*, 748-753. Menlo Park, CA: American Association for Artificial Intelligence.