A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text

Hai Leong Chieu
DSO National Laboratories
20 Science Park Drive
Singapore 118230
chaileon@dso.org.sg

Hwee Tou Ng
Department of Computer Science
School of Computing
National University of Singapore
3 Science Drive 2, Singapore 117543
nght@comp.nus.edu.sg

Abstract

In this paper, we present a classification-based approach towards single-slot as well as multi-slot information extraction (IE). For single-slot IE, we worked on the domain of Seminar Announcements, where each document contains information on only one seminar. For multi-slot IE, we worked on the domain of Management Succession. For this domain, we restrict ourselves to extracting information sentence by sentence, in the same way as (Soderland 1999). Each sentence can contain information on several management succession events. By using a classification approach based on a maximum entropy framework, our system achieves higher accuracy than the best previously published results in both domains.

Introduction

Information Extraction (IE) can be defined as the task of automatically extracting fragments of text to fill slots in a database. Examples include extracting speaker and start-time of seminars from seminar announcements, or extracting persons moving in and out of corporate positions in a news article. Single-slot IE means that at most one template (or database record) is found in each document. Multi-slot IE means that zero or more templates can be found in one document. Recent research on machine learning IE focused mainly on single-slot, semi-structured domains (Califf 1998; Freitag and McCallum 1999; Ciravegna 2001). Work on IE on free text was mostly based on pattern-learning approaches (Soderland 1999; Yangarber et al. 2000). Both Soderland and Yangarber et al. have worked on the domain of Management Succession, where IE was rendered more difficult not only by the style of writing used in news articles, but also because one document might contain information on several distinct events (multi-slot IE).

Taira and Soderland (Soderland 2001; Taira and Soderland 2000) have also developed another system to do IE in the domain of medical reports. This system aims to extract templates from whole reports instead of individual sentences. They reported excellent results in the domain of thoracic radiology reports, but stated that human intervention is required before a rule is finally accepted during learning. McCallum, Freitag and Pereira (2000) used Maximum Entropy Markov Models for extracting question-answer pairs in lists of Frequently Asked Questions (FAQs). Although they made use of the maximum entropy framework, their method is still based on Markov Models. We show how IE can be addressed as a classification problem.

In this paper, we present our work on a single-slot, semi-structured domain (Seminar Announcements) as well as a multi-slot, free text domain (Management Succession). Both IE systems presented in this paper are built on maximum entropy classifiers. We have used the Java-based opennlp maximum entropy package[1].

Maximum Entropy Classifier

The maximum entropy framework estimates probabilities based on the principle of making as few assumptions as possible, other than the constraints imposed. Such constraints are usually derived from training data, expressing some relationship between features and outcome. The probability distribution that satisfies the above property is the one with the highest entropy. It is unique, agrees with the maximum-likelihood distribution, and has the exponential form (Della Pietra, Della Pietra, and Lafferty 1997):

$$p(o|h)=\frac{1}{Z(h)}\prod_{j=1}^{k}\alpha_j^{f_j(h,o)},$$

where *o* refers to the outcome and *h* the history (or context). *Z(h)* is a normalization function. Each feature function *f_j(h,o)* is a binary function. For example, in predicting if a word belongs to a word class, *o* is either true or false, and *h* refers to the surrounding context:

$$f_j(h,o)=\begin{cases}1 & if\ o=true\ and\ previous\,word=the\\ 0 & otherwise\end{cases}$$

[1] http://maxent.sourceforge.net

The parameters α_j are estimated by a procedure called Generalized Iterative Scaling (GIS) (Darroch and Ratcliff 1972). This is an iterative method that improves the estimation of the parameters at each iteration. We have run all our experiments with 300 iterations.

Due to sparse data, certain contexts are not seen with all outcomes. For example, in the seminar announcements domain, the context of "*previous word = at*" might never have been seen with the outcome *speaker*. Smoothing is implemented by simply adding a training instance for each context with each outcome.

The ability of the maximum entropy framework to take into account features that are not independent makes it suitable for the two tasks we are working on.

Seminar Announcements

For the single-slot task, we have chosen to work in the domain of seminar announcements. Previous work in this domain includes (Ciravegna 2001; Freitag and Kushmerick 2000; Freitag and McCallum 1999). Our approach is similar to that of (Borthwick 1999), where the task attempted was that of identifying named entities, such as person names. The features we used are however different from those used by Borthwick. From each seminar announcement, 4 slots are to be extracted: *speaker, start time, end time,* and *location*. We further divide each slot into 4 sub-classes, e.g. s*peaker-begin, speaker-continue, speaker-end,* and *speaker-unique*. A word that does not fill any slot belongs to another class called *not-a-slot*. During training, each word is used to generate one training example, and during testing, the trained classifier will classify each word into one of the 17 classes.

Features for Seminar Announcement

We define below several groups of features. For each group, a training or test instance would typically have one feature set to 1, and the rest of the features in that group will be set to 0. The groups are:

(i) Unigrams. The string of each word w is used as a feature. So is that of the previous word w_{-1} and the next word w_{+1}. Each example has three features w, w_{-1} and w_{+1} set to 1 for this group.

(ii) Bigrams. The pair of word strings (w_{-2}, w_{-1}) of the previous two words is used as a feature. So is that of the next two words (w_{+1}, w_{+2}).

(iii) Zone and InitCaps. Some announcements contain sentence tags. The system will process each document sentence by sentence. Texts within the pair of tags <*sentence*> and </*sentence*> are taken to be one sentence. Texts that are outside of sentence tags are processed as one continuous sentence. Words within sentence tags are taken to be in *TXT* zone. Words outside such tags are taken to be in a *FRAG* zone. This group of features consists of 2 features *(InitCaps, TXT)* and *(InitCaps, FRAG)*. For words starting with a capital letter (*InitCaps*), one of the 2 features *(InitCaps, TXT)* or *(InitCaps, FRAG)* will be set to 1, depending on the zone the word appears in. If a word does not start with a capital letter, then both features are set to 0.

Who: Dr. Koji Ikuta
Center for Robotics in Microelectronics
University of California, Santa Barbara
Topic: Shape Memory Alloy Servo Actuator

Figure 1: Part of a seminar announcement

(iv) Zone and InitCaps of w_{-1} and w_{+1}. If the previous word has *InitCaps*, another feature *(InitCaps, TXT)*$_{PREV}$ or *(InitCaps, FRAG)*$_{PREV}$ will be set to 1. Same for the next word.

(v) Heading. Heading is defined to be the word before the last colon ":". For example, in Figure 1, the heading of the words "*Dr. Koji Ikuta*" and "*Center for Robotics in Microelectronics*" is "*Who*". The system will distinguish between words on the first line of the heading (e.g. *Who-first-line*) from words on other lines (*Who-other-lines).* There is at most one feature set to 1 for this group.

(vi) First Word. This group contains only one feature *FIRSTWORD*, which is set to 1 if the word is the first word of a sentence.

(vii) Time Expressions. If the word string of w matches the regular expression: *[digit]+:[digit]+,* then this feature will be set to 1.

(viii) Names. We used lists of first names and last names downloaded from the U.S. Census Bureau website[2]. If w has *InitCaps* and is found in the list of first names, the feature *FIRSTNAME* will be set to 1. If w_{-1} (or w_{+1}) has *InitCaps* and is found in the list of first names then *FIRSTNAME*$_{PREV}$ (*FIRSTNAME*$_{NEXT}$) will be set to 1. Similarly for *LASTNAME*.

(ix) New Word. If w is not found in /usr/dict/words on Linux, then a feature *NEW* will be set to 1.

Testing

During testing, it is possible that the classifier produces a sequence of inadmissible classes (e.g. *speaker-begin* followed by *location-unique*). To eliminate such sequences, we define the transition probability between word classes $P(c_i|c_j)$ to be equal to 1 if the sequence is admissible, and 0 otherwise. The Viterbi algorithm is then used to select the sequence of word classes with the highest probability. The probability of a sequence s of words is defined as follows:

$$P(c_1, c_2, ..., c_n | s) = \prod_{i=1}^{n} P(c_i | s) * P(c_i | c_{i-1}),$$

where $P(c_i|s)$ is determined by the maximum entropy classifier. It is possible that for certain slots (e.g. *speaker*), more than one instance is found within the same seminar

[2] http://www.census.gov/genealogy/names

	SP	LOC	ST	ET	All
ME_2	72.6	82.6	99.6	94.2	86.9
$(LP)^2$	77.6	75.0	99.0	95.5	86.0
SNoW	73.8	75.2	99.6	96.3	85.3
ME_1	65.3	82.3	99.6	94.5	85.0
BWI	67.7	76.7	99.6	93.9	83.9
HMM	76.6	78.6	98.5	62.1	81.8
Rapier	53.0	72.7	93.4	96.2	77.3
SRV	56.3	72.3	98.5	77.9	77.0
Whisk	18.3	66.4	92.6	86.0	64.8

Table 1: F-measure on CMU seminar announcements. SP = speaker, LOC = location, ST = start time, and ET = end time. "All" is the weighted average of the 4 slots.

announcement. In that case, only the best instance (with the highest probability) is used.

Experimental Results

The data consists of 485 seminar announcements[3] (895 KB, 102K words). We did a 5-fold experiment. In each trial, we partition the data into two halves, using one half for training and the other half for testing. Our results in Table 1 are the average of these 5 trials. Other than our own and SNoW's (Roth and Yih 2001) results, accuracy on the 4 slots of all other systems are taken from (Ciravegna 2001).

We used the MUC7-scorer to score each slot. The all-slots score is the weighted average of the four slots, where each slot is weighted by the total number of possible slots in the whole data set (485 seminar announcements have *start times*, 464 have *locations*, 409 have *speakers*, and 228 have *end times*). We compare our two systems ME_1 and ME_2 with other published systems. ME_1 uses only feature groups (i) to (vii) (i.e. no external knowledge). ME_2 uses all the 9 feature groups. Comparing the all-slots scores, ME_2 outperforms $(LP)^2$ (Ciravegna 2001), SNoW (Roth and Yih 2001), BWI (Freitag and Kushmerick 2000), HMM (Freitag and McCallum 1999), Rapier (Califf 1998), SRV (Freitag 1998), and Whisk (Soderland 1999).

Both $(LP)^2$ and SNoW use shallow natural language processing: $(LP)^2$ uses a morphological analyzer, a part-of-speech tagger, and a user defined dictionary (e.g. *pm* is of semantic category *timeid*). SNoW also uses part-of-speech tagging. Without any external knowledge, ME_1 outperforms all systems other than $(LP)^2$ and SNoW. ME_2 used only three lists: first names, last names, and a lexicon list. BWI reported an improvement from 67.7 to 73.5 for the speaker slot when the same three lists are used. However, they did not report results of all 4 slots in this experimental setting.

We have shown that a classification-based approach like maximum entropy is able to achieve state-of-the-art accuracy when provided with informative features.

Management Succession

In this paper, we do not attempt the full MUC-6 Scenario Template task. We present a system that attempts IE on a sentence-by-sentence basis, extracting templates of 4 slots. The 4 slots are person-in (person moving into a corporate position), person-out (person leaving a corporate position), the corporate position, and the corporate name. This task has been defined by Soderland (1999). We show that by using a series of classifiers, our system outperforms WHISK (Soderland 1999). Collins and Miller (1998) also worked on this domain, and achieved excellent results. However, their method requires the test data to be manually tagged with indicator words, making the task a lot easier (sentences with two events are tagged with two indicator words).

In this domain, multi-slot IE is required. A sentence might contain zero, one, or more templates. The approach used for single-slot IE can only give us the possible candidates that can fill each slot. Another classifier is required to decide which candidates should fill the same template, and which should fill a different template.

One can build up a list of candidates for each slot by using our approach for single-slot extraction. In this case, all candidates should fill some template. Another way is to use all entities of a certain semantic class as candidates for a particular slot. For example, all persons can be candidates for the slot of person-in and person-out in the management succession task. Using this approach, some candidates will have to be rejected and not go into any template. The determination of semantic class might require considerable domain knowledge. Riloff and Jones (1996) used an unsupervised approach to build semantic lexicons. Grishman (2001) reiterated the importance of word class discovery in IE. In this paper, we use as input sentences syntactically analyzed by BADGER (Fisher et al. 1995). In these sentences, person, organization, and position names are tagged. In each sentence, we use as candidates for corporate positions all position names tagged and candidates for corporate names all the organization names tagged. For person-in and person-out, we built two separate classifiers to produce the list of candidates for the two slots. A relation classifier is then built to classify binary relationship between each pair of candidates. In a sentence in which a total of *n* candidates have been found, there are *n(n-1)/2* possible binary relations.

Figure 2 shows an example output by the relation classifier. The whole process is shown in Figure 3. This approach is new and different from other published methods on this task.

Overview of the System

The multi-slot IE system is made up of four components:

(i) Text-Filtering. During testing, a text categorization module is first used to eliminate documents that do not contain any relevant templates. For this module, we used

[3] Downloaded from http://www.isi.edu/~muslea/RISE/index.html

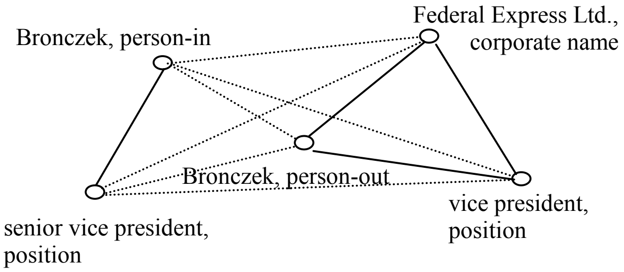
Federal Express Ltd., corporate name
Bronczek, person-in
Bronczek, person-out
vice president, position
senior vice president, position

Figure 2: 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*." A solid line means that the relation is classified as positive, and a dashed line means it is classified as negative. The difficulty in extracting two templates from this sentence is evident. The relation classifier got this one right.

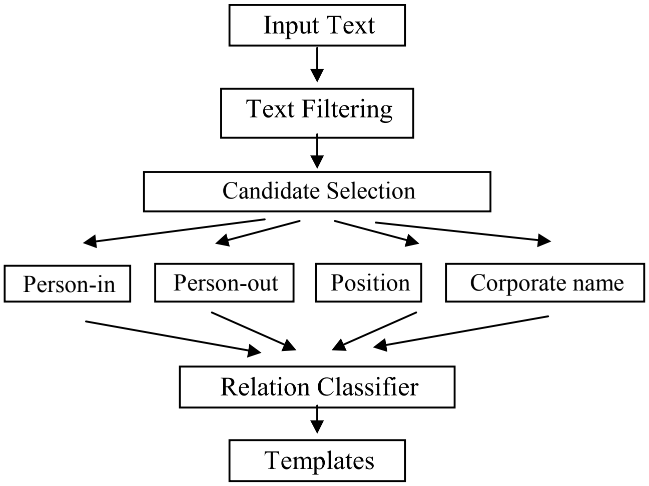

Figure 3: Process of multi-slot information extraction

svmlight (Joachims 1998), and trained it with documents containing relevant templates as positive examples and those that do not as negative examples. Each document is represented by a feature vector, which is the normalized vector of term frequencies of selected terms. The terms are selected using the correlation metric (Ng, Goh, and Low 1997):

$$C = \frac{(N_{r+}N_{n-} - N_{r-}N_{n+})\sqrt{N}}{\sqrt{(N_{r+} + N_{r-})(N_{n+} + N_{n-})(N_{r+} + N_{n+})(N_{r-} + N_{n-})}}$$

where N_{r+} (N_{n+}) is the number of relevant (non-relevant) training documents in which a term occurs, N_{r-} (N_{n-}) is the number of relevant (non-relevant) training documents in which a term does not occur, and N is the total number of documents. The top 1000 terms (with highest C), and the bottom 1000 terms (with lowest C) are used as features.

For the 100 test documents, 50 of them contain relevant templates. This module found 60 test documents to be relevant, out of which 49 are really relevant: false negative of 1 document and false positive of 11 documents.

(ii) Candidate Selection. For corporate positions and corporate names, all positions and organizations tagged in the BADGER output are used as candidates. The selection process can be considered to be the tagging of all position and organization names. For person-in and person-out, as there are usually more clues within a sentence indicating a person as in or out, we have built a classifier for each slot. Each classifier is trained using sentences from relevant documents only (out of the 498 training documents, 298 are relevant). During testing, sentences belonging to the 60 documents found to be positive in the text-filtering module will be processed, and each person appearing in these sentences can be classified as person-in, person-out, both, or neither.

(iii) Relation Classification. The relation classifier finds pair-wise relations between entities, for example *(Bronczek, person-in)* and *(senior vice president, position)*. It is possible to have position-position where a person either moves into or leaves two different posts at once. As a result, we have allowed all 4^2=16 relations.

(iv) Template Building. Only templates that contain at least a person-in or a person-out will be considered as valid templates to be output. Given a graph of relations between entities in a sentence (see Figure 2), an edge exists between two entities if and only if their relation is classified as positive. The system will first try to find the largest clique (complete subgraph). Among cliques of the same size, it will choose the one with the highest product of the probabilities of relations. The use of product implicitly assumes that the relations within the same template are independent, which is untrue. However, this is just used as a means of selecting between templates of the same size, and serves well for this purpose.

Once a template is formed, the entities that form that template will be removed from the graph, and the system will start anew with the new graph of remaining entities. From this new graph, if there are still persons left, then a second (and possibly a third, fourth, and so on) template can be formed. In Figure 2, two templates will be formed, one from each clique.

Features for Person-in and Person-out Classifier

These features are derived from the BADGER output format (see Figure 4). A BADGER sentence is divided into phrases, of the following 10 types: *SUBJ, VB, OBJ, PP, REL_S, REL_V, REL_O, REL_P, ADVP,* and *SAID.* Verbs are tagged with their lemma. Verbs in the *VB* phrase in the passive voice are also tagged. The feature groups are:

(i) Candidate Phrase Type and VB Phrase. Candidate phrase type refers to the type of the phrase in which the candidate is found. For the example in Figure 4, "*Alan G. Spoon*" is found in the *SUBJ* phrase. For the candidate "*Alan G. Spoon*", the feature *(SUBJ, WILL_SUCCEED)* will be set to 1.

```
{SUBJ @PN[ ALAN G. SPOON ]PN }
{VB   WILL SUCCEED @SUCCEED }
{OBJ  MR. @PN[ GRAHAM ]PN }
{PP   AS @PS[ PRESIDENT OF THE COMPANY ]PS . }
```

Figure 4: Example of BADGER output of sentence: "*Alan G. Spoon will succeed Mr. Graham as president of the company.*"

(ii) Nearest Seed Word in Sentence. We automatically determined a list of seed words from the training data. These are words with the highest correlation metric *C* as defined in the last section. The seed words found are *President, Officer, Succeed, Nam (lemma of name), Vice, Chief, Executive,* and *Chairman.* Intuitively, the presence of these seed words in a sentence will improve the chance of a person name in the sentence to fill a person-in or person-out slot. In Figure 4*, "Alan G. Spoon*" will have the feature *seed_SUCCEED* set to 1.

(iii) Agent Verbs. Verbs for which the candidate is an agent. These verbs are determined by a few rules listed in Table 2. For example, if a candidate is in the *SUBJ* phrase, all verbs of the active voice found in phrases of type *VB, REL_S*, and *REL_V* will be taken to be agent verbs. Each of these verbs will have a feature set to 1. In Figure 4*, "Alan G. Spoon*" will have only one agent verb feature set to 1: *agent_SUCCEED.*

(iv) Patient Verbs. Verbs for which the candidate is a patient. The rules are analogous to those of Table 2. In Figure 4, "*Alan G. Spoon*" has no patient verbs, and "*Graham*" has one patient verb feature *patient_SUCCEED.*

For the example in Figure 4, "*Alan G. Spoon*" will have the following 3 features set to 1: *(SUBJ, WILL_SUCCEED), seed_SUCCEED,* and *agent_SUCCEED.* "*Graham*" will have the following 3 features set to 1: *(OBJ, WILL_SUCCEED), seed_PRESIDENT,* and *patient_SUCCEED.*

Features for Relation Classifier

The input to the relation classifier is a pair of entities {*(name1, class1), (name2, class2)*}, for example, {*(Spoon, person-in), (Graham, person-out)*}. One classifier is used to classify all such input pairs into true or false. All features used by the relation classifier are characterized by the class combination *(class1, class2)*, e.g. *(person-in, person-out).*

The following feature groups are used:

(i) Same Phrase. If *name1* and *name2* are identical, the feature *(class1, class2, same-phrase)* is set to 1. In this case, all other features used in the relation classifier will be set to 0. The same phrase will never fill two different slots in the same template. There are altogether 4^2=16 features in this group, one for each class combination.

(ii) Words between *Name1* and *Name2*. This is the feature *(class1, class2, STRING)*, where *STRING* is the exact string between *name1* and *name2*.

Candidate	Voice	Agent Verb Phrase
SUBJ	Active	*VB, REL_S, REL_V*
OBJ	Active	*REL_O*
	Passive	*VB, REL_V*
PP	Active	*REL_P*
REL_O *REL_S*	Active	Verbs after the candidate in the same phrase

Table 2: Rules for the determination of agent verbs.

System	Recall	Precision	F-measure
Soderland	46.4	68.9	55.5
Our results	49.1	74.6	59.2

Table 3: Comparison of recall, precision, and F-measure

(iii) Phrase Types. This is the feature *(class1, class2, phrase_type1, phrase_type2)*, where *phrase_type1* and *phrase_type2* are the phrase types of the two entities.

(iv) Other Entities. These features indicate whether there exist a person (including pronouns), position, or organization between *name1* and *name2*. If there are no person or pronoun between name1 and name2, then a feature *(class1, class2, NO_PERSON_BETWEEN)* is set to 1. Similarly for corporate names and positions.

For the input pair of entities {*(Spoon, person-in), (Graham, person-out)*}, the following 5 features are set to 1: *(person-in, person-out, WILL_SUCCEED_MR)*
(person-in, person-out, SUBJ, OBJ)
(person-in, person-out, NO_PERSON_BETWEEN)
(person-in, person-out, NO_ORG_BETWEEN)
(person-in, person-out, NO_POST_BETWEEN)

Experimental Results

We used the same training and test data, and the same scoring criteria as Soderland (1999). In order for an output template to be considered correct, all the slots of the template must match a key template in the manually annotated templates. If the output template contains an extra slot, all slots of the output template are considered as false positives. The data provided by Soderland contains 6,915 training instances and the test data are sentences extracted from 100 test documents, comprising 2,840[4] instances, with a total of 169 templates, 84 person-ins, 100 person-outs, 148 positions, and 92 organizations. In Table 3, "Soderland" refers to his best results in terms of F-measure, obtained by using all 6,900 instances for training. Our system achieves higher accuracy than Soderland's.

Conclusion

Most previous work on machine learning approaches to information extraction focused on single-slot IE for semi-structured text. Relatively less research was done on multi-

[4] Soderland stated that he used 2,839 instances. This difference is due to a formatting error in the test data for the instance 9301060123-29.

slot IE. Past work on multi-slot IE for free text mainly used pattern-learning approaches. In this paper, we have tackled the problem using a classification approach instead, incorporating two separate steps: candidate selection and template building (by a relation classifier). On two benchmark data sets, our system achieves higher accuracy than the best previously published results on IE from semi-structured as well as free text. In order to do a statistical significance test, we need to know the detailed test results of previous systems. Since these are not available, we are unable to conduct such a significance test.

In recent years, the emphasis on IE has shifted to adaptive IE. We feel that a classification approach allows systems to adapt to a new domain simply by using a standard set of features. Besides, there are many machine learning classifier algorithms available (such as support vector machines, decision trees, and neural networks). This should offer an attractive alternative to pattern-learning methods.

Acknowledgements

Many thanks to Stephen Soderland for sharing with us his training and test data for the Management Succession task.

References

Borthwick, A. 1999. A Maximum Entropy Approach to Named Entropy Recognition. Ph.D. dissertation. Computer Science Department. New York University.

Califf, M. E. 1998. Relational Learning Techniques for Natural Language Information Extraction. Ph.D. dissertation, University of Texas at Austin.

Ciravegna, F. 2001. Adaptive Information Extraction from Text by Rule Induction and Generalisation. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI),* 1251-1256.

Collins, M. and Miller, S. 1998. Semantic Tagging using a Probabilistic Context Free Grammar. In *Proceedings of the Sixth Workshop on Very Large Corpora,* 38-48.

Darroch, J. N. and Ratcliff, D. 1972. Generalized Iterative Scaling for Log-Linear Models. *The Annals of Mathematical Statistics,* 43(5*)*: 1470-1480.

Della Pietra, S., Della Pietra, V., and Lafferty J. 1997. Inducing Features of Random Fields*. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(4),* 380-393.

Fisher, D., Soderland, S., McCarthy, J., Feng, F., and Lehnert, W. 1995. Description of the UMass System as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conferenc*e, 221-236.

Freitag, D. 1998. Information Extraction from HTML: Application of a General Machine Learning Approach. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 517-523.

Freitag, D. and Kushmerick, N. 2000. Boosted Wrapper Induction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI),* 577-583.

Freitag, D. and McCallum, A. 1999. Information Extraction with HMM and Shrinkage. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI),* 31-36.

Grishman, R. 2001. Adaptive Information Extraction and Sublanguage Analysis. In *Proceedings of IJCAI Workshop on Adaptive Text Extraction and Mining,* 77-79.

Joachims, T. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of the Tenth European Conference on Machine Learning,* 137-142.

McCallum, A., Freitag, D., and Pereira, F. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning,* 591-598.

Ng, H. T., Goh, W. B., and Low, K. L. 1997. Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR),* 67-73.

Riloff, E. and Jones, R. 1996. Learning Dictionaries for Information Extraction by Multi-level Bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI),* 1044-1049.

Roth, D. and Yih, W. T. 2001. Relational Learning via Propositional Algorithms: An Information Extraction Case Study. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI),* 1257-1263.

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

Soderland, S. 2001. Building a Machine Learning Based Text Understanding System. In *Proceedings of IJCAI Workshop on Adaptive Text Extraction and Mining*, 64-70.

Taira, R. K. and Soderland, S. 2000. A Statistical Natural Language Processor for Medical Reports. In *Proceedings of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences.*

Yangarber, R., Grishman, R., Tapanainen, P., and Huttunen, S. 2000. Unsupervised Discovery of Scenario-Level Patterns for Information Extraction. In *Proceedings of the Sixth Applied Natural Language Processing Conference*. 282-289.