Named Entity Recognition: A Maximum Entropy Approach Using Global Information

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

This paper presents a maximum entropy-based named entity recognizer (NER). It differs from previous machine learning-based NERs in that it uses information from the whole document to classify each word, with just one classifier. Previous work that involves the gathering of information from the whole document often uses a secondary classifier, which corrects the mistakes of a primary sentence-based classifier. In this paper, we show that the maximum entropy framework is able to make use of global information directly, and achieves performance that is comparable to the best previous machine learning-based NERs on MUC-6 and MUC-7 test data.

1 Introduction

Considerable amount of work has been done in recent years on the named entity recognition task, partly due to the Message Understanding Conferences (MUC). A named entity recognizer (NER) is useful in many NLP applications such as information extraction, question answering, etc. On its own, a NER can also provide users who are looking for person or organization names with quick information. In MUC-6 and MUC-7, the named entity task is defined as finding the following classes of names: person, organization, location, date, time, money, and percent (Chinchor, 1998; Sundheim, 1995) Machine learning systems in MUC-6 and MUC-7 achieved accuracy comparable to rule-based systems on the named entity task.

Statistical NERs usually find the sequence of tags that maximizes the probability $p(N|S)$, where S is the sequence of words in a sentence, and N is the sequence of named-entity tags assigned to the words in S. Attempts have been made to use global information (e.g., the same named entity occurring in different sentences of the same document), but they usually consist of incorporating an additional classifier, which tries to correct the errors in the output of a first NER (Mikheev et al., 1998; Borthwick, 1999). We propose maximizing $p(N|S, Doc)$, where N is the sequence of named-entity tags assigned to the words in the sentence S, and Doc is the information that can be extracted from the whole document containing S. Our system is built on a maximum entropy classifier. By making use of global context, it has achieved excellent results on both MUC-6 and MUC-7 official test data. We will refer to our system as MENERGI (Maximum Entropy Named Entity Recognizer using Global Information).

As far as we know, no other NERs have used information from the whole document (global) as well as information within the same sentence (local) in one framework. The use of global features has improved the performance on MUC-6 test data from 90.75% to 93.27% (27% reduction in errors), and the performance on MUC-7 test data from 85.22% to 87.24% (14% reduction in errors). These results are achieved by training on the official MUC-6 and MUC-7 training data, which is much less training data than is used by other machine learning systems that worked on the MUC-6 or MUC-7 named entity task (Bikel et al., 1997; Bikel et al., 1999; Borthwick, 1999).

We believe it is natural for authors to use abbreviations in subsequent mentions of a named entity (i.e., first *"President George Bush"* then *"Bush"*). As such, global information from the whole context of a document is important to more accurately recognize named entities. Although we have not done any experiments on other languages, this way of using global features from a whole document should be applicable to other languages.

2 Related Work

Recently, statistical NERs have achieved results that are comparable to hand-coded systems. Since MUC-6, BBN's Hidden Markov Model (HMM) based IdentiFinder (Bikel et al., 1997) has achieved remarkably good performance. MUC-7 has also seen hybrids of statistical NERs and hand-coded systems (Mikheev et al., 1998; Borthwick, 1999), notably Mikheev's system, which achieved the best performance of 93.39% on the official NE test data. MENE (Maximum Entropy Named Entity) (Borthwick, 1999) was combined with Proteus (a hand-coded system), and came in fourth among all MUC-7 participants. MENE without Proteus, however, did not do very well and only achieved an F-measure of 84.22% (Borthwick, 1999).

Among machine learning-based NERs, IdentiFinder has proven to be the best on the official MUC-6 and MUC-7 test data. MENE (without the help of hand-coded systems) has been shown to be somewhat inferior in performance. By using the output of a hand-coded system such as Proteus, MENE can improve its performance, and can even outperform IdentiFinder (Borthwick, 1999).

Mikheev et al. (1998) did make use of information from the whole document. However, their system is a hybrid of hand-coded rules and machine learning methods. Another attempt at using global information can be found in (Borthwick, 1999). He used an additional maximum entropy classifier that tries to correct mistakes by using reference resolution. Reference resolution involves finding words that co-refer to the same entity. In order to train this error-correction model, he divided his training corpus into 5 portions of 20% each. MENE is then trained on 80% of the training corpus, and tested on the remaining 20%. This process is repeated 5 times by rotating the data appropriately. Finally, the concatenated 5 * 20% output is used to train the reference resolution component. We will show that by giving the first model some global features, MENERGI outperforms Borthwick's reference resolution classifier. On MUC-6 data, MENERGI also achieves performance comparable to IdentiFinder when trained on similar amount of training data.

In Section 5, we try to compare results of MENE, IdentiFinder, and MENERGI. However, both MENE and IdentiFinder used more training data than we did (we used only the official MUC-6 and MUC-7 training data). On the MUC-6 data, Bikel et al. (1997; 1999) do have some statistics that show how IdentiFinder performs when the training data is reduced. Our results show that MENERGI performs as well as IdentiFinder when trained on comparable amount of training data.

3 System Description

The system described in this paper is similar to the MENE system of (Borthwick, 1999). It uses a maximum entropy framework and classifies each word given its features. Each name class N is subdivided into 4 sub-classes, i.e., *N_begin*, *N_continue*, *N_end*, and *N_unique*. Hence, there is a total of 29 classes (7 name classes $\times$ 4 sub-classes $+$ 1 not-a-name class).

3.1 Maximum Entropy

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 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 et al., 1997):

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

where o refers to the outcome, h the history (or context), and $Z(h)$ is a normalization function. In addition, 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:

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

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 used the Java-based opennlp maximum entropy package[1].

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

3.2 Testing

During testing, it is possible that the classifier produces a sequence of inadmissible classes (e.g., *person_begin* followed by *location_unique*). To eliminate such sequences, we define a transition probability between word classes $P(c_i|c_j)$ to be equal to 1 if the sequence is admissible, and 0 otherwise. The probability of the classes $c_1, \ldots, c_n$ assigned to the words in a sentence s in a document D is defined as follows:

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

where $P(c_i|s, D)$ is determined by the maximum entropy classifier. A dynamic programming algorithm is then used to select the sequence of word classes with the highest probability.

4 Feature Description

The features we used can be divided into 2 classes: local and global. Local features are features that are based on neighboring tokens, as well as the token itself. Global features are extracted from other occurrences of the same token in the whole document.

The local features used are similar to those used in BBN's IdentiFinder (Bikel et al., 1999) or MENE (Borthwick, 1999). However, to classify a token w, while Borthwick uses tokens from w_{-2} to w_{+2} (from two tokens before to two tokens after w), we used only the tokens w_{-1}, w, and w_{+1}. Even with local features alone, MENERGI outperforms MENE (Borthwick, 1999). This might be because our features are more comprehensive than those used by Borthwick. In IdentiFinder, there is a priority in the feature assignment, such that if one feature is used for a token, another feature lower in priority will not be used. In the maximum entropy framework, there is no such constraint. Multiple features can be used for the same token.

Feature selection is implemented using a feature cutoff: features seen less than a small count during training will not be used. We group the features used into feature groups. Each feature group can be made up of many binary features. For each token w, zero, one, or more of the features in each feature group are set to 1.

4.1 Local Features

The local feature groups are:

Non-Contextual Feature: This feature is set to 1 for all tokens. This feature imposes constraints that are based on the probability of each name class during training.

Token satisfies	Example	Feature
Starts with a capital letter, ends with a period	*Mr.*	*InitCap-Period*
Contains only one capital letter	*A*	*OneCap*
All capital letters and period	*CORP.*	*AllCaps-Period*
Contains a digit	*AB3,* *747*	*Contain-Digit*
Made up of 2 digits	*99*	*TwoD*
Made up of 4 digits	*1999*	*FourD*
Made up of digits and slash	*01/01*	*Digit-slash*
Contains a dollar sign	*US$20*	*Dollar*
Contains a percent sign	*20%*	*Percent*
Contains digit and period	*$US3.20*	*Digit-Period*

Table 1: Features based on the token string

Zone: MUC data contains SGML tags, and a document is divided into zones (e.g., headlines and text zones). The zone to which a token belongs is used as a feature. For example, in MUC-6, there are four zones (*TXT, HL, DATELINE, DD*). Hence, for each token, one of the four features *zone-TXT, zone-HL, zone-DATELINE,* or *zone-DD* is set to 1, and the other 3 are set to 0.

Case and Zone: If the token w starts with a capital letter (*initCaps*), then an additional feature *(initCaps, zone)* is set to 1. If it is made up of all capital letters, then *(allCaps, zone)* is set to 1. If it starts with a lower case letter, and contains both upper and lower case letters, then *(mixedCaps, zone)* is set to 1. A token that is *allCaps* will also be *initCaps*. This group consists of (3 × *total number of possible zones*) features.

Case and Zone of w_{+1} and w_{-1}: Similarly, if w_{+1} (or w_{-1}) is *initCaps*, a feature *(initCaps, zone)*$_{NEXT}$ (or *(initCaps, zone)*$_{PREV}$) is set to 1, etc.

Token Information: This group consists of 10 features based on the string w, as listed in Table 1. For example, if a token starts with a capital letter and ends with a period (such as *Mr.*), then the feature *InitCapPeriod* is set to 1, etc.

First Word: This feature group contains only one feature *firstword*. If the token is the first word of a sentence, then this feature is set to 1. Otherwise, it is set to 0.

Lexicon Feature: The string of the token w is used as a feature. This group contains a large number of features (one for each token string present in the training data). At most one feature in this group will be set to 1. If w is seen infrequently during training (less than a small count), then w will not be selected as a feature and all features in this group are set to 0.

Lexicon Feature of Previous and Next Token: The string of the previous token w_{-1} and the next token w_{+1} is used with the *initCaps* information of w. If w has *initCaps*, then a feature *(initCaps,* $w_{+1})_{NEXT}$ is set to 1. If w is not *initCaps*, then *(not-initCaps,* $w_{+1})_{NEXT}$ is set to 1. Same for w_{-1}. In the case where the next token w_{+1} is a hyphen, then w_{+2} is also used as a feature: *(initCaps,* $w_{+2})_{NEXT}$ is set to 1. This is because in many cases, the use of hyphens can be considered to be optional (e.g., *third-quarter* or *third quarter*).

Out-of-Vocabulary: We derived a lexicon list from WordNet 1.6, and words that are not found in this list have a feature *out-of-vocabulary* set to 1.

Dictionaries: Due to the limited amount of training material, name dictionaries have been found to be useful in the named entity task. The importance of dictionaries in NERs has been investigated in the literature (Mikheev et al., 1999). The sources of our dictionaries are listed in Table 2. For all lists except locations, the lists are processed into a list of tokens (unigrams). Location list is processed into a list of unigrams and bigrams (e.g., *New York*). For locations, tokens are matched against unigrams, and sequences of two consecutive tokens are matched against bigrams. A list of words occurring more than 10 times in the training data is also collected (*commonWords*). Only tokens with *initCaps* not found in *commonWords* are tested against each list in Table 2. If they are found in a list, then a feature for that list will be set to 1. For example, if *Barry* is not in *commonWords* and is found in the list of person first names, then the feature *PersonFirstName* will be set to 1. Similarly, the tokens w_{+1} and w_{-1} are tested against each list, and if found, a corresponding feature will be set to 1. For example, if w_{+1} is found in the list of person first names, the feature *PersonFirstName*$_{NEXT}$ is set to 1.

Month Names, Days of the Week, and Numbers: If w is *initCaps* and is one of *January, February, ..., December*, then the feature *MonthName* is set to 1. If w is one of *Monday, Tuesday, ..., Sunday*, then the feature *DayOfTheWeek* is set to 1. If w is a number string (such as *one, two,* etc), then the feature *NumberString* is set to 1.

Suffixes and Prefixes: This group contains only two features: *Corporate-Suffix* and *Person-Prefix*. Two lists, *Corporate-Suffix-List* (for corporate suffixes) and *Person-Prefix-List* (for person prefixes), are collected from the training data. For corporate suffixes, a list of tokens *cslist* that occur frequently as the last token of an organization name is collected from the training data. Frequency is calculated by counting the number of distinct previous tokens that each token has (e.g., if *Electric Corp.* is seen 3 times, and *Manufacturing Corp.* is seen 5 times during training, and *Corp.* is not seen with any other preceding tokens, then the "frequency" of *Corp.* is 2). The most frequently occurring last words of organization names in *cslist* are compiled into a list of corporate suffixes, *Corporate-Suffix-List*. A *Person-Prefix-List* is compiled in an analogous way. For MUC-6, for example, *Corporate-Suffix-List* is made up of $\{$ *ltd., associates, inc., co, corp, ltd, inc, committee, institute, commission, university, plc, airlines, co., corp.*$\}$ and *Person-Prefix-List* is made up of $\{$ *succeeding, mr., rep., mrs., secretary, sen., says, minister, dr., chairman, ms.*$\}$. For a token w that is in a consecutive sequence of *initCaps* tokens $(w_{-m}, \ldots, w, \ldots, w_{+n})$, if any of the tokens from w_{+1} to w_{+n} is in *Corporate-Suffix-List*, then a feature *Corporate-Suffix* is set to 1. If any of the tokens from w_{-m-1} to w_{-1} is in *Person-Prefix-List*, then another feature *Person-Prefix* is set to 1. Note that we check for w_{-m-1}, the word preceding the consecutive sequence of *initCaps* tokens, since person prefixes like *Mr., Dr.,* etc are not part of person names, whereas corporate suffixes like *Corp., Inc.,* etc are part of corporate names.

4.2 Global Features

Context from the whole document can be important in classifying a named entity. A name already mentioned previously in a document may appear in abbreviated form when it is mentioned again later. Previous work deals with this problem by correcting inconsistencies between the named entity classes assigned to different occurrences of the same entity (Borthwick, 1999; Mikheev et al., 1998). We often encounter sentences that are highly ambiguous in themselves, without some prior knowledge of the entities concerned. For example:

McCann initiated a new global system. (1)
CEO of McCann ... (2)

Description	Source
Location Names	http://www.timeanddate.com http://www.cityguide.travel-guides.com http://www.worldtravelguide.net
Corporate Names	http://www.fmlx.com
Person First Names	http://www.census.gov/genealogy/names
Person Last Names	

Table 2: Sources of Dictionaries

The McCann family . . . (3)

In sentence (1), McCann can be a person or an organization. Sentence (2) and (3) help to disambiguate one way or the other. If all three sentences are in the same document, then even a human will find it difficult to classify McCann in (1) into either person or organization, unless there is some other information provided.

The global feature groups are:

InitCaps of Other Occurrences (ICOC): There are 2 features in this group, checking for whether the first occurrence of the same word in an unambiguous position (non first-words in the *TXT* or *TEXT* zones) in the same document is *initCaps* or *not-initCaps*. For a word whose *initCaps* might be due to its position rather than its meaning (in headlines, first word of a sentence, etc), the case information of other occurrences might be more accurate than its own. For example, in the sentence that starts with *"Bush put a freeze on . . . "*, because *Bush* is the first word, the initial caps might be due to its position (as in *"They put a freeze on . . . "*). If somewhere else in the document we see *"restrictions put in place by President Bush"*, then we can be surer that *Bush* is a name.

Corporate Suffixes and Person Prefixes of Other Occurrences (CSPP): If *McCann* has been seen as *Mr. McCann* somewhere else in the document, then one would like to give person a higher probability than organization. On the other hand, if it is seen as *McCann Pte. Ltd.*, then organization will be more probable. With the same *Corporate-Suffix-List* and *Person-Prefix-List* used in local features, for a token w seen elsewhere in the same document with one of these suffixes (or prefixes), another feature *Other-CS* (or *Other-PP*) is set to 1.

Acronyms (ACRO): Words made up of all capitalized letters in the text zone will be stored as acronyms (e.g., *IBM*). The system will then look for sequences of initial capitalized words that match the acronyms found in the whole document. Such sequences are given additional features of *A_begin*, *A_continue*, or *A_end*, and the acronym is given a feature *A_unique*. For example, if *FCC* and *Federal Communications Commission* are both found in a document, then *Federal* has *A_begin* set to 1, *Communications* has *A_continue* set to 1, *Commission* has *A_end* set to 1, and *FCC* has *A_unique* set to 1.

Sequence of Initial Caps (SOIC): In the sentence *Even News Broadcasting Corp., noted for its accurate reporting, made the erroneous announcement.*, a NER may mistake *Even News Broadcasting Corp.* as an organization name. However, it is unlikely that other occurrences of *News Broadcasting Corp.* in the same document also co-occur with *Even*. This group of features attempts to capture such information. For every sequence of initial capitalized words, its longest substring that occurs in the same document as a sequence of *initCaps* is identified. For this example, since the sequence *Even News Broadcasting Corp.* only appears once in the document, its longest substring that occurs in the same document is *News Broadcasting Corp.* In this case, *News* has an additional feature of *I_begin* set to 1, *Broadcasting* has an additional feature of *I_continue* set to 1, and *Corp.* has an additional feature of *I_end* set to 1.

Unique Occurrences and Zone (UNIQ): This group of features indicates whether the word w is unique in the whole document. w needs to be in *initCaps* to be considered for this feature. If w is unique, then a feature *(Unique, Zone)* is set to 1, where *Zone* is the document zone where w appears. As we will see from Table 3, not much improvement is derived from this feature.

5 Experimental Results

The baseline system in Table 3 refers to the maximum entropy system that uses only local features. As each global feature group is added to the list of features, we see improvements to both MUC-6 and

	MUC-6	MUC-7
Baseline	90.75%	85.22%
+ ICOC	91.50%	86.24%
+ CSPP	92.89%	86.96%
+ ACRO	93.04%	86.99%
+ SOIC	93.25%	87.22%
+ UNIQ	93.27%	87.24%

Table 3: F-measure after successive addition of each global feature group

Systems	MUC-6		MUC-7	
	No. of Articles	No. of Tokens	No. of Articles	No. of Tokens
MENERGI	318	160,000	200	180,000
IdentiFinder	–	650,000	–	790,000
MENE	–	–	350	321,000

Table 4: Training Data

MUC-7 test accuracy.[2] For MUC-6, the reduction in error due to global features is 27%, and for MUC-7, 14%. ICOC and CSPP contributed the greatest improvements. The effect of UNIQ is very small on both data sets.

All our results are obtained by using only the official training data provided by the MUC conferences. The reason why we did not train with both MUC-6 and MUC-7 training data at the same time is because the task specifications for the two tasks are not identical. As can be seen in Table 4, our training data is a lot less than those used by MENE and IdentiFinder[3]. In this section, we try to compare our results with those obtained by IdentiFinder '97 (Bikel et al., 1997), IdentiFinder '99 (Bikel et al., 1999), and MENE (Borthwick, 1999). IdentiFinder '99's results are considerably better than IdentiFinder '97's. IdentiFinder's performance in MUC-7 is published in (Miller et al., 1998). MENE has only been tested on MUC-7.

For fair comparison, we have tabulated all results with the size of training data used (Table 5 and Table 6). Besides size of training data, the use of dictionaries is another factor that might affect performance. Bikel et al. (1999) did not report using any dictionaries, but mentioned in a footnote that they have added list membership features, which have helped marginally in certain domains. Borthwick (1999) reported using dictionaries of person first names, corporate names and suffixes, colleges and universities, dates and times, state abbreviations, and world regions.

Systems	Size of training data	F-measure
SRA '95	Hand-coded	96.4%
IdentiFinder '99	650,000 words	94.9%
MENERGI	160,000 tokens	93.27%
IdentiFinder '99 (from graph)	$>$ 200,000 words	About 93%
IdentiFinder '97	450,000 words	93%
IdentiFinder '97	about 100,000 words	91%-92%

Table 5: Comparison of results for MUC-6

Systems	Size of training data	F-measure
LTG system '98	Hybrid hand-coded	93.39%
IdentiFinder '98	790,000 words	90.44%
MENE + Proteus '98	Hybrid hand-coded 321,000 tokens	88.80%
MENERGI	180,000 tokens	87.24%
MENE+reference-resolution '99	321,000 tokens	86.56%
MENE '98	321,000 tokens	84.22%

Table 6: Comparison of results for MUC-7

In MUC-6, the best result is achieved by SRA (Krupka, 1995). In (Bikel et al., 1997) and (Bikel et al., 1999), performance was plotted against training data size to show how performance improves with training data size. We have estimated the performance of IdentiFinder '99 at 200K words of training data from the graphs.

For MUC-7, there are also no published results on systems trained on only the official training data of 200 aviation disaster articles. In fact, training on the official training data is not suitable as the articles in this data set are entirely about aviation disasters, and the test data is about air vehicle launching. Both BBN and NYU have tagged their own data to supplement the official training data. Even with less training data, MENERGI outperforms Borthwick's MENE + reference resolution (Borthwick, 1999). Except our own and MENE + reference resolution, the results in Table 6 are all official MUC-7 results.

The effect of a second reference resolution classifier is not entirely the same as that of global features. A secondary reference resolution classifier has information on the class assigned by the primary classifier. Such a classification can be seen as a not-always-correct summary of global features. The secondary classifier in (Borthwick, 1999) uses

[2] MUC data can be obtained from the Linguistic Data Consortium: http://www.ldc.upenn.edu

[3] Training data for IdentiFinder is actually given in words (i.e., 650K & 790K words), rather than tokens

information not just from the current article, but also from the whole test corpus, with an additional feature that indicates if the information comes from the same document or from another document. We feel that information from a whole corpus might turn out to be noisy if the documents in the corpus are not of the same genre. Moreover, if we want to test on a huge test corpus, indexing the whole corpus might prove computationally expensive. Hence we decided to restrict ourselves to only information from the same document.

Mikheev et al. (1998) have also used a maximum entropy classifier that uses already tagged entities to help tag other entities. The overall performance of the LTG system was outstanding, but the system consists of a sequence of many hand-coded rules and machine-learning modules.

6 Conclusion

We have shown that the maximum entropy framework is able to use global information directly. This enables us to build a high performance NER without using separate classifiers to take care of global consistency or complex formulation on smoothing and backoff models (Bikel et al., 1997). Using less training data than other systems, our NER is able to perform as well as other state-of-the-art NERs. Information from a sentence is sometimes insufficient to classify a name correctly. Global context from the whole document is available and can be exploited in a natural manner with a maximum entropy classifier. We believe that the underlying principles of the maximum entropy framework are suitable for exploiting information from diverse sources. Borthwick (1999) successfully made use of other hand-coded systems as input for his MENE system, and achieved excellent results. However, such an approach requires a number of hand-coded systems, which may not be available in languages other than English. We believe that global context is useful in most languages, as it is a natural tendency for authors to use abbreviations on entities already mentioned previously.

References

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: A high-performance learning name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 194–201.

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1/2/3):211–231.

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

Nancy Chinchor. 1998. MUC-7 named entity task definition, version 3.5. In *Proceedings of the Seventh Message Understanding Conference*.

J. N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480.

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

George R. Krupka. 1995. SRA: Description of the SRA system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference*, pages 221–235.

Andrei Mikheev, Claire Grover, and Marc Moens. 1998. Description of the LTG system used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference*.

Andrei Mikheev, Marc Moens, and Claire Grover. 1999. Named entity recognition without gazetteers. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 1–8.

Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, and the Annotation Group. 1998. Algorithms that learn to extract information BBN: Description of the SIFT system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference*.

Beth M. Sundheim. 1995. Named entity task definition, version 2.1. In *Proceedings of the Sixth Message Understanding Conference*, pages 319–332.