Named Entity Recognition with a Maximum Entropy Approach

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

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

1 Introduction

The named entity recognition (NER) task involves identifying noun phrases that are names, and assigning a class to each name. This task has its origin from the Message Understanding Conferences (MUC) in the 1990s, a series of conferences aimed at evaluating systems that extract information from natural language texts. It became evident that in order to achieve good performance in information extraction, a system needs to be able to recognize names. A separate subtask on NER was created in MUC-6 and MUC-7 (Chinchor, 1998).

Much research has since been carried out on NER, using both knowledge engineering and machine learning approaches. At the last CoNLL in 2002, a common NER task was used to evaluate competing NER systems. In this year's CoNLL, the NER task is to tag noun phrases with the following four classes: person (PER), organization (ORG), location (LOC), and miscellaneous (MISC).

This paper presents a maximum entropy approach to the NER task, where NER not only made use of local context within a sentence, but also made use of other occurrences of each word within the same document to extract useful features (global features). Such global features enhance the performance of NER (Chieu and Ng, 2002b).

2 A Maximum Entropy Approach

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. The features used in the maximum entropy framework are binary. An example of a feature function is

$$f_j(h,o) = \begin{cases} 1 & \text{if } o = \textit{org-B}, \text{ word} = \textit{PETER} \\ 0 & \text{otherwise} \end{cases}$$

The parameters α_j are estimated by a procedure called Generalized Iterative Scaling (GIS) (Darroch and Ratcliff, 1972). This is an iterative procedure that improves the estimation of the parameters at each iteration.

The maximum entropy classifier is used to classify each word as one of the following: the beginning of a NE (B tag), a word inside a NE (C tag), the last word of a NE (L tag), or the unique word in a NE (U tag). During testing, it is possible that the classifier produces a sequence of inadmissible classes (e.g., *PER-B* followed by *LOC-L*). 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. The Viterbi algorithm is then used to select the sequence of word classes with the highest probability.

3 Feature Representation

We present two systems: a system ME1 that does not make use of any external knowledge base other than the training data, and a system ME2 that makes use of additional features derived from name lists. ME1 is used for both English and German. For German, however, for features that made use of the word string, the lemma (provided in the German training and test data) is used instead of the actual word.

3.1 Lists derived from training data

The training data is first preprocessed to compile a number of lists that are used by both ME1 and ME2. These lists are derived automatically from the training data.

Frequent Word List (FWL) This list consists of words that occur in more than 5 different documents.

Useful Unigrams (UNI) For each name class, words that precede the name class are ranked using correlation metric (Chieu and Ng, 2002a), and the top 20 are compiled into a list.

Useful Bigrams (UBI) This list consists of bigrams of words that precede a name class. Examples are "CITY OF", "ARRIVES IN", etc. The list is compiled by taking bigrams with higher probability to appear before a name class than the unigram itself (e.g., "CITY OF" has higher probability to appear before a location than "OF"). A list is collected for each name class. We have attempted to use bigrams that appear after a name class, but for English at least, we have been unable to compile any such meaningful bigrams. A possible explanation is that in writing, people tend to explain with bigrams such as "CITY OF" before mentioning the name itself.

Useful Word Suffixes (SUF) For each word in a name class, three-letter suffixes with high correlation metric score are collected. This is especially important for the MISC class, where suffixes such as "IAN" and "ISH" often appear.

Useful Name Class Suffixes (NCS) A suffix list is compiled for each name class. These lists capture tokens that frequently terminate a particular name class. For example, the ORG class often terminates with tokens such as INC and COMMITTEE, and the MISC class often terminates with CUP, OPEN, etc.

Function Words (FUN) Lower case words that occur within a name class. These include "van der", "of", etc.

3.2 Local Features

The basic features used by both ME1 and ME2 can be divided into two classes: local and global (Chieu and Ng, 2002b). Local features of a token w are those that are derived from the sentence containing w. Global features are derived by looking up other occurrences of w within the same document.

In this paper, w_{-i} refers to the ith word before w, and w_{+i} refers to the ith word after w. The features used are similar to those used in (Chieu and Ng, 2002b). Local features include:

First Word, Case, and Zone For English, each document is segmented by simple rules into 4 zones: headline (HL), author (AU), dateline (DL), and text (TXT). To identify the zones, a DL sentence is first identified using a regular expression. The system then looks for an AU sentence that occurs before DL using another regular expression. All sentences other than AU that occur before the DL sentence are then taken to be in the HL zone. Sentences after the DL sentence are taken to be in the TXT zone. If no DL sentence can be found in a document, then the first sentence of the document is taken as HL, and the rest as TXT. For German, the first sentence of each document is taken as HL, and the rest as TXT. Zone is used as part of the following features:

If w starts with a capital letter (i.e., initCaps), and it is the first word of a sentence, a feature *(firstword-initCaps, zone)* is set to 1. If it is initCaps but not the first word, a feature *(initCaps, zone)* is set to 1. If it is the first word but not initCaps, *(firstword-notInitCaps, 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*.

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.

Case Sequence Suppose both w_{-1} and w_{+1} are initCaps. Then if w is initCaps, a feature I is set to 1, else a feature NI is set to 1.

Token Information These features are based on the string w, such as contains-digits, contains-dollar-sign, etc (Chieu and Ng, 2002b).

Lexicon Feature The string of w is used as a feature. This group contains a large number of features (one for each token string present in the training data).

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}.

Hyphenated Words Hyphenated words w of the form $s1$-$s2$ have a feature U-U set to 1 if both $s1$ and $s2$ are initCaps. If $s1$ is initCaps but not $s2$, then the features U=$s1$, L=$s2$, and U-L are set to 1. If $s2$ is initCaps but not $s1$, then the features U=$s2$, L=$s1$, and L-U are set to 1.

Within Quotes/Brackets Sequences of tokens within quotes or brackets have a feature to indicate that they are within quotes. We found this feature useful for MISC class, where names such as movie names often appear within quotes.

Rare Words If w is not found in FWL, then this feature is set to 1.

Bigrams If (w_{-2}, w_{-1}) is found in UBI for the name class nc, then the feature BI-nc is set to 1.

Word Suffixes If w has a 3-letter suffix that can be found in SUF for the name class nc, then the feature SUF-nc is set to 1.

Class Suffixes For w in a consecutive sequence of initCaps tokens $(w, w_{+1}, \ldots, w_{+n})$, if any of the tokens from w_{+1} to w_{+n} is found in the NCS list of the name class nc, then the feature NCS-nc is set to 1.

Function Words If w is part of a sequence found in FUN, then this feature is set to 1.

3.3 Global Features

The global features include:

Unigrams If another occurrence of w in the same document has a previous word wp that can be found in UNI, then these words are used as features $Other$-$occurrence$-$prev$=wp.

Bigrams If another occurrence of w has the feature BI-nc set to 1, then w will have the feature $OtherBI$-nc set to 1.

Class Suffixes If another occurrence of w has the feature NCS-nc set to 1, then w will have the feature $OtherNCS$-nc set to 1.

InitCaps of Other Occurrences This feature checks for whether the first occurrence of the same word in an unambiguous position (non first-words in the TXT zone) in the same document is initCaps or not. 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.

Acronyms 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 InitCaps 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.

Name Class of Previous Occurrences The name class of previous occurrences of w is used as a feature, similar to (Zhou and Su, 2002). We use the occurrence where w is part of the longest name class phrase (name class with the most number of tokens). For example, if w is the second token in a person name class phrase of 5 tokens, then a feature $2Person5$ is set to 1. During training, the name classes are known. During testing, the name classes are the ones already assigned to tokens in the sentences already processed.

This last feature makes the order of processing important. As HL sentences usually contain less context, they are processed after the other sentences.

3.4 Name List

In additional to the above features used by both ME1 and ME2, ME2 uses additional features derived from name lists compiled from a variety of sources. These sources are the Internet and the list provided by the organizers of this shared task. The list is a mapping of sequences of words to name classes. An example of an entry in the list is "JOHN KENNEDY : PERSON". Words that are part of a sequence of words mapped to a name class nc will have a feature $CLASS$=nc set to 1. Another list of weekdays and month names is also used in the same way. For ME2, we have also manually added additional entries into the automatically compiled NCS lists.

4 Experiments

The English training and test data are part of the Reuters Corpus, Volume 1[1]. The German training and test data are part of the European Corpus Initiative, Multilingual Corpus 1. The best results obtained on the developement and test sets of the 2 languages are as shown in Table 2. Results in Table 1 are obtained by applying ME1, without the help of name lists, on the 2 languages.

The best results for English are obtained using ME2, which made use of name lists compiled from the Internet and the list provided with the training set (See Section 3.4). The best results on German are obtained by using part-of-speech tags (provided in both training and test data) as an additional feature to the features used by ME1.

For all experiments, features that occur only once in the training data are not used, and the GIS algorithm is run for 600 iterations. Running more iterations does not bring about any significant improvement to the accuracy.

Our system usually does well for the LOC and PER class, but fails to do as well for the MISC and ORG class. The bad performance on the MISC class agrees with the observations of (Carreras et al., 2002). We felt that the

[1] http://about.reuters.com/researchandstandards/corpus/

English devel.	Precision	Recall	$F_{\beta=1}$
LOC	93.77%	94.23%	94.00
MISC	89.20%	85.14%	87.13
ORG	87.25%	85.76%	86.50
PER	94.14%	95.98%	95.05
Overall	91.76%	91.45%	91.60

English test	Precision	Recall	$F_{\beta=1}$
LOC	89.27%	90.29%	89.78
MISC	80.38%	78.21%	79.28
ORG	82.43%	82.18%	82.30
PER	91.50%	91.84%	91.67
Overall	86.83%	86.84%	86.84

German devel.	Precision	Recall	$F_{\beta=1}$
LOC	74.42%	56.90%	64.49
MISC	72.49%	33.66%	45.98
ORG	81.00%	47.06%	59.53
PER	84.34%	58.03%	68.75
Overall	78.80%	49.84%	61.06

German test	Precision	Recall	$F_{\beta=1}$
LOC	72.08%	55.36%	62.62
MISC	64.04%	34.03%	44.44
ORG	75.95%	46.57%	57.74
PER	87.87%	61.84%	72.59
Overall	77.05%	51.73%	61.90

Table 1: Results for development and test set for the two languages by ME1

English devel.	Precision	Recall	$F_{\beta=1}$
LOC	95.39%	95.75%	95.57
MISC	90.94%	86.01%	88.41
ORG	89.12%	87.99%	88.56
PER	94.85%	96.96%	95.89
Overall	93.16%	92.86%	93.01

English test	Precision	Recall	$F_{\beta=1}$
LOC	90.88%	91.37%	91.12
MISC	80.15%	78.21%	79.16
ORG	83.82%	84.83%	84.32
PER	93.07%	93.82%	93.44
Overall	88.12%	88.51%	88.31

German devel.	Precision	Recall	$F_{\beta=1}$
LOC	71.08%	65.96%	68.42
MISC	72.23%	32.97%	45.28
ORG	80.86%	48.67%	60.76
PER	79.45%	65.95%	72.07
Overall	76.15%	54.62%	63.61

German test	Precision	Recall	$F_{\beta=1}$
LOC	69.23%	59.13%	63.78
MISC	62.05%	33.43%	43.45
ORG	76.70%	48.12%	59.14
PER	88.82%	75.15%	81.41
Overall	76.83%	57.34%	65.67

Table 2: Best results: For English, name lists are used. For German, part-of-speech tags are used

MISC class is particularly difficult due to its generality (it can refer to anything from movie titles to sports events).

Acknowledgements We would like to thank Yoong Keok Lee for helping us to apply boosting and feature selection to the maximum entropy algorithm, although these were not used in the final system.

References

Xavier Carreras, Lluis Marquez, and Lluis Padro. 2002. Named Entity Extraction using AdaBoost. In *Proceedings of the Sixth Conference on Natural Language Learning*, pages 167–170.

Hai Leong Chieu and Hwee Tou Ng. 2002a. A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pages 786–791.

Hai Leong Chieu and Hwee Tou Ng. 2002b. Named Entity Recognition: A Maximum Entropy Approach Using Global Information. In *Proceedings of the Nineteenth International Conference on Computational Linguistics*, pages 190–196.

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.

GuoDong Zhou and Jian Su. 2002. Named Entity Recognition using an HMM-based Chunk Tagger. In *Proceedings of the Fortieth Annual Meeting of the Association for Computational Linguistics*, pages 473–480.