Maximum Entropy Models for Named Entity Recognition

Oliver Bender[1] and **Franz Josef Och**[2] and **Hermann Ney**[1]

[1]Lehrstuhl für Informatik VI
Computer Science Department
RWTH Aachen - University of Technology
D-52056 Aachen, Germany
{bender,ney}@cs.rwth-aachen.de

[2]Information Sciences Institute
University of Southern California
Marina del Rey, CA 90292
och@isi.edu

Abstract

In this paper, we describe a system that applies maximum entropy (ME) models to the task of named entity recognition (NER). Starting with an annotated corpus and a set of features which are easily obtainable for almost any language, we first build a baseline NE recognizer which is then used to extract the named entities and their context information from additional non-annotated data. In turn, these lists are incorporated into the final recognizer to further improve the recognition accuracy.

1 Introduction

In this paper, we present an approach for extracting the named entities (NE) of natural language inputs which uses the maximum entropy (ME) framework (Berger et al., 1996). The objective can be described as follows. Given a natural input sequence $w_1^N = w_1 ... w_n ... w_N$ we choose the NE tag sequence $c_1^N = c_1 ... c_n ... c_N$ with the highest probability among all possible tag sequences:

$$\hat{c}_1^N = \operatorname*{argmax}_{c_1^N} \left\{ Pr(c_1^N | w_1^N) \right\} \quad .$$

The argmax operation denotes the search problem, i.e. the generation of the sequence of named entities. According to the CoNLL-2003 competition, we concentrate on four types of named entities: persons (`PER`), locations (`LOC`), organizations (`ORG`), and names of miscellaneous entities (`MISC`) that do not belong to the previous three groups, e.g.

> [`PER` Clinton] 's [`ORG` Ballybunion] fans invited to [`LOC` Chicago] .

Additionally, the task requires the processing of two different languages from which only English was specified before the submission deadline. Therefore, the system described avoids relying on language-dependent knowledge but instead uses a set of features which are easily obtainable for almost any language.

The remainder of the paper is organized as follows: in section 2, we outline the ME framework and specify the features that were used for the experiments. We describe the training and search procedure of our approach. Section 3 presents experimental details and shows results obtained on the English and German test sets. Finally, section 4 closes with a summary and an outlook for future work.

2 Maximum Entropy Models

For our approach, we directly factorize the posterior probability and determine the corresponding NE tag for each word of an input sequence. We assume that the decisions only depend on a limited window of $w_{n-2}^{n+2} = w_{n-2} ... w_{n+2}$ around the current word w_n and on the two predecessor tags. Thus, we obtain the following second-order model:

$$\begin{aligned} Pr(c_1^N | w_1^N) &= \prod_{n=1}^{N} Pr(c_n | c_1^{n-1}, w_1^N) \\ &\underset{model}{=} \prod_{n=1}^{N} p(c_n | c_{n-2}^{n-1}, w_{n-2}^{n+2}) \ . \end{aligned}$$

A well-founded framework for directly modeling the posterior probability $p(c_n | c_{n-2}^{n-1}, w_{n-2}^{n+2})$ is maximum entropy (Berger et al., 1996). In this framework, we have a set of M feature functions $h_m(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}), m = 1, \ldots, M$. For each feature function h_m, there exists a model parameter λ_m. The posterior probability can then be modeled as follows:

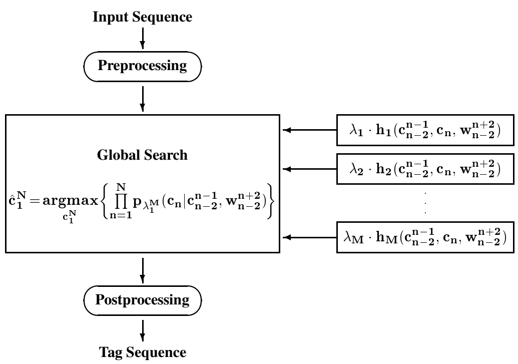

Figure 1: Architecture of the maximum entropy model approach.

$$p_{\lambda_1^M}(c_n|c_{n-2}^{n-1}, w_{n-2}^{n+2}) = \frac{\exp\left[\sum_{m=1}^{M} \lambda_m h_m(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2})\right]}{\sum_{c'} \exp\left[\sum_{m=1}^{M} \lambda_m h_m(c_{n-2}^{n-1}, c', w_{n-2}^{n+2})\right]} . \quad (1)$$

The architecture of the ME approach is summarized in Figure 1.

As for the CoNLL-2003 shared task, the data sets often provide additional information like part-of-speech (POS) tags. In order to take advantage of these knowledge sources, our system is able to process several input sequences at the same time.

2.1 Feature Functions

We have implemented a set of binary valued feature functions for our system:

Lexical features: The words w_{n-2}^{n+2} are compared to a vocabulary. Words which are seen less than twice in the training data are mapped onto an 'unknown word'. Formally, the feature

$$h_{w,d,c}(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}) = \delta(w_{n+d}, w) \cdot \delta(c_n, n) , \quad d \in \{-2, ..., 2\} ,$$

will fire if the word w_{n+d} matches the vocabulary entry w and if the prediction for the current NE tag equals c. $\delta(\cdot, \cdot)$ denotes the Kronecker-function.

Word features: Word characteristics are covered by the word features, which test for:

- Capitalization: These features will fire if w_n is capitalized, has an internal capital letter, or is fully capitalized.
- Digits and numbers: ASCII digit strings and number expressions activate these features.
- Pre- and suffixes: If the prefix (suffix) of w_n equals a given prefix (suffix), these features will fire.

Transition features: Transition features model the dependence on the two predecessor tags:

$$h_{c',d,c}(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}) = \delta(c_{n-d}, c') \cdot \delta(c_n, n) , \quad d \in \{1, 2\} .$$

Prior features: The single named entity priors are incorporated by prior features. They just fire for the currently observed NE tag:

$$h_c(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}) = \delta(c_n, c) .$$

Compound features: Using the feature functions defined so far, we can only specify features that refer to a single word or tag. To enable also word phrases and word/tag combinations, we introduce the following compound features:

$$h_{\{z_1,d_1\},...,\{z_K,d_K\},c}\,(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}) = \prod_{k=1}^{K} h_{z_k,d_k,c}\,(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}) , \quad z_k \in \{w, c'\} , \; d_k \in \{-2, ..., 2\} .$$

Dictionary features: Given a list L of named entities, the dictionary features check whether or not an entry of L occurs within the current window. Formally,

$$h_{L,c}(c_{n-2}^{n-1}, c_n, w_{n-2}^{n+2}) = \text{entryOccurs}(L, w_{n-2}^{n+2}) \cdot \delta(c_n, n) .$$

Respectively, the dictionary features fire if an entry of a context list appears beside or around the current word position w_n.

2.2 Feature Selection

Feature selection plays a crucial role in the ME framework. In our system, we use simple count-based feature reduction. Given a threshold K, we only include those features that have been observed on the training data at least K times. Although this method does not guarantee to obtain a minimal set of features, it turned out to perform well in practice.

Experiments were carried out with different thresholds. It turned out that for the NER task, a threshold of 1 for the English data and 2 for the German corpus achieved the best results for all features, except for the prefix and suffix features, for which a threshold of 5 (10 resp.) yielded best results.

2.3 Training

For training purposes, we consider the set of manually annotated and segmented training sentences to form a single long sentence. As training criterion, we use the maximum class posterior probability criterion:

$$\hat{\lambda}_1^M = \underset{\lambda_1^M}{\operatorname{argmax}} \left\{ \sum_{n=1}^{N} \log p_{\lambda_1^M}(c_n | c_{n-2}^{n-1}, w_{n-2}^{n+2}) \right\} .$$

This corresponds to maximizing the likelihood of the ME model. Since the optimization criterion is convex, there is only a single optimum and no convergence problems occur. To train the model parameters λ_1^M we use the Generalized Iterative Scaling (GIS) algorithm (Darroch and Ratcliff, 1972).

In practice, the training procedure tends to result in an overfitted model. To avoid overfitting, (Chen and Rosenfeld, 1999) have suggested a smoothing method where a Gaussian prior on the parameters is assumed. Instead of maximizing the probability of the training data, we now maximize the probability of the training data times the prior probability of the model parameters:

$$\hat{\lambda}_1^M = \underset{\lambda_1^M}{\operatorname{argmax}} \left\{ p(\lambda_1^M) \cdot \sum_{n=1}^{N} p_{\lambda_1^M}(c_n | c_{n-2}^{n-1}, w_{n-2}^{n+2}) \right\},$$

where

$$p(\lambda_1^M) = \prod_m \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{\lambda_m^2}{2\sigma^2} \right] .$$

This method tries to avoid very large lambda values and avoids that features that occur only once for a specific class get value infinity. Note that there is only one parameter σ for all model parameters λ_1^M.

2.4 Search

In the test phase, the search is performed using the so-called maximum approximation, i.e. the most likely sequence of named entities $\hat{c}_1^N$ is chosen among all possible sequences c_1^N:

$$\begin{aligned} \hat{c}_1^N &= \underset{c_1^N}{\operatorname{argmax}} \left\{ Pr(c_1^N | w_1^N) \right\} \\ &= \underset{c_1^N}{\operatorname{argmax}} \left\{ \prod_{n=1}^{N} p_{\lambda_1^M}(c_n | c_{n-2}^{n-1}, w_{n-2}^{n+2}) \right\} . \end{aligned}$$

Therefore, the time-consuming renormalization in Eq. 1 is not needed during search. We run a Viterbi search to find the highest probability sequence (Borthwick et al., 1998).

3 Experiments

Experiments were performed on English and German test sets. The English data was derived from the Reuters corpus[1] while the German test sets were extracted from the ECI Multilingual Text corpus. The data sets contain tokens (words and punctuation marks), information about the sentence boundaries, as well as the assigned NE tags. Additionally, a POS tag and a syntactic chunk tag were assigned to each token. On the tag level, we distinguish five tags (the four NE tags mentioned above and a filler tag).

3.1 Incorporating Lists of Names and Non-annotated Data

For the English task, extra lists of names were provided, and for both languages, additional non-annotated data was supplied. Hence, the challenge was to find ways of incorporating this information. Our system aims at this challenge via the use of dictionary features.

While the provided lists could straightforward be integrated, the raw data was processed in three stages:

1. Given the annotated training data, we used all features except the dictionary ones to build a first baseline NE recognizer.

2. Applying this recognizer, the non-annotated data was processed and all named entities plus contexts (up to three words beside the classified NE and the two surrounding words) were extracted and stored as additional lists.

3. These lists could again be integrated straightforward. It turned out that a threshold of five yielded best results for both the lists of named entities as well as for the context information.

3.2 Results

Table 1 and Table 2 present the results obtained on the development and test sets. For both languages, 1 000 GIS iterations were performed and the Gaussian prior method was applied.

Test Set	Precision	Recall	$F_{\beta=1}$
English devel.	90.01%	88.52%	89.26
English test	84.45%	82.90%	83.67
German devel.	73.60%	57.73%	64.70
German test	76.12%	60.74%	67.57

Table 1: Overall performance of the baseline system on the development and test sets in English and German.

[1] The Reuters corpus was kindly provided by Reuters Limited.

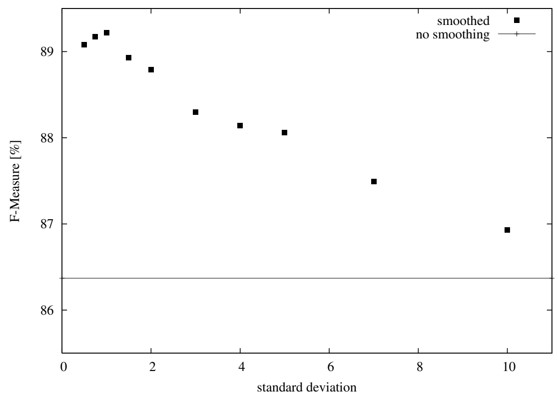

Figure 2: Results of the baseline system for different smoothing parameters.

As can be derived from table 1, our baseline recognizer clearly outperforms the CoNLL-2003 baseline (e.g. $F_{\beta=1} = 89.26$ vs. $F_{\beta=1} = 71.18$). To investigate the contribution of the Gaussian prior method, several experiments were carried out for different standard deviation parameters σ. Figure 2 depicts the obtained F-Measures in comparison to the performance of non-smoothed ME models ($F_{\beta=1} = 86.37$). The gain in performance is obvious.

By incorporating the information extracted from the non-annotated data our system is further improved. On the German data, the results show a performance degradation. The main reason for this is due to the capitalization of German nouns. Therefore, refined lists of proper names are necessary.

English devel.	Precision	Recall	$F_{\beta=1}$
LOC	93.27%	93.58%	93.42
MISC	88.51%	81.02%	84.60
ORG	84.67%	83.59%	84.13
PER	92.26%	91.91%	92.09
Overall	90.32%	88.86%	89.58

English test	Precision	Recall	$F_{\beta=1}$
LOC	86.44%	89.81%	88.09
MISC	78.35%	73.22%	75.70
ORG	80.27%	76.16%	78.16
PER	89.77%	87.88%	88.81
Overall	84.68%	83.18%	83.92

German devel.	Precision	Recall	$F_{\beta=1}$
LOC	72.23%	71.13%	71.67
MISC	66.08%	44.95%	53.51
ORG	71.90%	56.49%	63.27
PER	82.77%	68.59%	75.02
Overall	74.16%	61.16%	67.04

German test	Precision	Recall	$F_{\beta=1}$
LOC	69.06%	69.66%	69.36
MISC	66.52%	46.27%	54.58
ORG	68.84%	53.17%	60.00
PER	87.91%	75.48%	81.22
Overall	74.82%	63.82%	68.88

Table 2: Results of the final system on the development and test sets in English and German.

4 Summary

In conclusion, we have presented a system for the task of named entity recognition that uses the maximum entropy framework. We have shown that a baseline system based on an annotated training set can be improved by incorporating additional non-annotated data.

For future investigations, we have to think about a more sophisticated treatment of the additional information. One promising possibility could be to extend our system as follows: apply the baseline recognizer to annotate the raw data as before, but then use the output to train a new recognizer. The scores of the new system are incorporated as further features and the procedure is iterated until convergence.

References

A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–72, March.

A. Borthwick, J. Sterling, E. Agichtein, and R. Grisham. 1998. NYU: Description of the MENE named entity system as used in MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 6 pages, Fairfax, VA, April. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/.

S. Chen and R. Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical Report CMUCS-99-108, Carnegie Mellon University, Pittsburgh, PA.

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