Memory-Based Named Entity Recognition using Unannotated Data

Fien De Meulder and **Walter Daelemans**
CNTS - Language Technology Group
University of Antwerp
{Fien.DeMeulder,Walter.Daelemans}@ua.ac.be

Abstract

We used the memory-based learner Timbl (Daelemans et al., 2002) to find names in English and German newspaper text. A first system used only the training data, and a number of gazetteers. The results show that gazetteers are not beneficial in the English case, while they are for the German data. Type-token generalization was applied, but also reduced performance. The second system used gazetteers derived from the unannotated corpus, as well as the ratio of capitalized versus uncapitalized use of each word. These strategies gave an increase in performance.

1 Introduction

This paper describes a memory-based approach to learning names in English and German newspaper text.

The first system used no unannotated data - only the provided training material, and a number of gazetteers. It was shown that the gazetteers made for a better performance in the German task, but not in the English task. Type-token generalization was helpful for neither English nor German.

The second system used unannotated data, but only for the English task. The extra data were used in two ways: first, more gazetteers were derived from the corpus by exploiting conjunctions: if in a conjunction of capitalized strings one string is recognized as being a certain type of name, the other strings are assumed to be of the same type and stored in a new gazetteer. This list was then used to construct an additional feature for training the machine learning algorithm. The second approach counts how often each word form in the additional corpus is capitalized, and how often it is not. This is used as another feature for the learning algorithm.

2 Memory Based Learning

We used Timbl (Daelemans et al., 2002), a memory-based learner. When presented with training instances, the learner stores them all, and then classifies new data on the basis of its *k* nearest neighbours in the training set. Before classification, the learner assigns weights to each of the features, marking their importance for the learning task. Features with higher weights are treated as more important in classification as those with lower weights.

Timbl has some parameters which can be adjusted in order to improve learning. For the NER system described in this paper, we varied the parameters *k* and *m*. *k* is the number of nearest neighbours Timbl looks at. *m* determines the feature metrics, i.e. the importance weights given to each feature, and the way similarity between values of the same feature is computed. This parameter can be adjusted separately for each feature. The two metrics used were *weighted overlap* and *modified value difference*.

3 System 1: Description

3.1 Features

For the basic English system, 37 features were used. The first seven features were the lowercase versions of the focus word, and a context of three words to the left and the right. The next seven features were the part-of-speech tags of the same seven words. Then followed seven features indicating for each of the seven words if they were capitalized or not. The next six features represented the first and last three letters of the word to be classified. These features were included in order to make it possible for the memory-based learner to use word-internal information. Frequent prefixes and suffixes can thus be used to learn names. Finally, ten features indicated if the focus word appears in any of the gazetteers used for this task. These gazetteers are discussed in more detail in the next section.

For the German system, the same features were used, with an additional seven features: for each word in the seven-word window, the stem of the word was also included.

3.2 Gazetteers

Ten gazetteers were used to provide features. These gazetteers listed names of all four kinds, as well as words which often appear inside names (such as *International* (for organization names) and *de* (for person names)).

3.3 Type – token generalization

A module was created to generalize NE tags from types to tokens. It is a simple program which assumes that if two capitalized words have the same form, they will also have the same NE tag. This is potentially problematic, because many words can be used either as part of a name or not, and in this case it indeed proved to be unhelpful.

4 System 2: Description

For the extended English system, four more features were added to each instance: the first four indicated if the focus word was part of a named entity found in a list of named entities derived from the unannotated data. The second new feature indicated if the focusword is capitalized or uncapitalized most often in the unannotated data.

4.1 Gazetteers extracted from conjunctions

First, potential names were identified in the unannotated data. This was done using the gazetteers which were used for the first system, and a simple grammar of names. Then we looked for all conjunctions of capitalized strings in the unannotated data. If one of the strings was tagged in its entirety as being of one NE type, and no other strings in the conjunction had another NE tag, it was hypothesized that all strings in this conjunction were of the same type. All strings would then be stored in a gazetteer of NEs of that type.

The next step was to add four more features to the training and test sets of the NE system. In the training and test texts, strings of capitalized words were matched with the strings in the newly made gazetteers. All instances were enlarged by four binary features, one for each type of NE (L, M, O, P). These features are on when the focus word (and its context in the case of a longer name) matches a string in the associated gazetteer, and off when it does not.

4.2 Ratio of capitalized to non-capitalized occurrence of tokens

A last feature added to all instances indicated if the focus word (the word to be classified) appears more often capitalized or uncapitalized in the unannotated corpus. This approach has been used earlier by (Collins, 2002). In order to make this feature, a list was made of all wordforms, converted to lowercase, in the corpus, and the ratio of capitalized to uncapitalized occurrences. The extra feature was binary: on if a word appears more often capitalized than not, and off otherwise.

5 System 1: Discussion of results

5.1 Role of gazetteers

Two experiments were run to assess the importance of the gazetteers in this experiment: the first used only the word to be classified and its context, the second used binary features indicating inclusion in gazetteers, as well as the features used in the first experiment. Perhaps surprisingly, the English system did worse when gazetteer information was used. This was true using the default parameter settings, and also after (limited) separate parameter optimization. The German system did slightly better on the development data when gazetteers were used.

The difference between the English and German systems is very surprising, as the lists were not adjusted to include extra German names. They contain mainly English and Dutch names, as a result of previous work on Dutch and English. In order to find an explanation, we looked at the performance (not optimized) of the lists on their own, not using any context or word-internal information at all. The result did not make things at all clearer: the precision of the lists on the German data was striking, even more so than on the English data.

English devel.	Precision	Recall	$F_{\beta=1}$
No gazetteers	84.09%	85.20%	84.64
With gazetteers	78.27%	78.11%	78.19
Only gazetteers	49.20%	33.82%	40.08

German devel.	Precision	Recall	$F_{\beta=1}$
No gazetteers	60.63%	48.36%	53.80
With gazetteers	61.35%	49.87%	55.02
Only gazetteers	29.53%	5.75%	9.62

Table 1: Role of gazetteers

5.2 Type – token generalization

Type-token generalization was attempted only on the English data. The intuition behind this approach is that a memory-based learner may recognize a name due to its context, but it will not generalize the classification to other tokens of the same type. However, a concern is that mistakes will be introduced by generalizing ambiguous words to the wrong type, and by repeating mistakes which would otherwise occur only sporadically. In the end, introducing generalization did not make much of a difference. While precision declines marginally (two more phrases were incorrectly tagged as names), recall is unaffected.

The results in Table 2 were derived using Timbl with default parameters. The lack of optimization explains the low result even without generalization.

5.3 Parameter optimization and feature selection

Parameter optimization was used both for system 1 and for system 2. This was combined with limited feature selection. The difference feature selection can make, is already obvious from the results above, and will be shown

English devel.	Precision	Recall	$F_{\beta=1}$
No generalization	75.90%	82.88%	79.23
With generalization	75.87%	82.88%	79.22

Table 2: Role of type – token generalization

in the rest of the paper also. Parameter optimization can have a major effect on performance of machine learning systems in general, and Timbl in particular, as can be seen in Table 3.

As was shown by Daelemans and Hoste (2002), parameter optimization and feature selection heavily interact in machine learning: separate optimization leads to inferior results to interleaved optimization. Different parameter settings might be best for different feature selections, and vice versa. It would therefore be best to optimize both at the same time, treating feature selection and parameter optimization together as one search space. This was done to a very limited extent for this problem, but because of the time needed for each experiment, a full search of the solution space was impossible.

Another restriction is the fact that not all parameters of the learner were optimized, again due to time constraints. The two that were found to have a great effect were used only. These are *k*, the number of nearest neighbours taken into account when classifying a new instance, and *m*, the feature metric. *m* was toggled between *weighted overlap* and *modified value difference*.

The results shown in Table 3 are those on the consistently best featureset found, i.e. the one using all information minus gazetteers.

On the German data, parameter optimization and feature selection were also found to be beneficial, but optimization had to be cut short due to time constraints.

English devel.	Precision	Recall	$F_{\beta=1}$
k=1, overlap	75.88%	82.88%	79.22
k=1, mvdm	82.28%	84.69%	83.47
k=3, overlap	74.04%	80.51%	77.14
k=3, mvdm	84.09%	85.20%	84.64
k=5, overlap	72.67%	79.21%	75.80
k=5, mvdm	83.94%	84.77%	84.35

Table 3: Role of parameter optimization

6 System 2: Discussion of results

In this system, extra information is added to the training set in the following way: the number of the instances in the training set remains the same, but the number of features for each instance is increased. The information for the extra instances is found in the unannotated data, so this should bring the benefit of using this extra information source. At the same time, only the hand-tagged training set is used, which means that no extra noise is introduced into the training set.

6.1 Gazetteers extracted from conjunctions

In this step, four new features were added to each instance in the training and test sets, one for each type of NE.

Even though gazetteers were already in use, we extracted new gazetteers from the unannotated data. The hope was that these gazetteers would be more useful for this particular task, as they would be corpus-specific. The gazetteers which were used originally, and which did not improve performance, were mainly taken off the internet, and partially hand-crafted. This means that they are general-purpose gazetteers. Also, they were a mixture of Dutch and English names. The new gazetteers were only English, and only included those names which were found in the Reuters corpus.

Once the gazetteers were extracted, their entries were matched against the text in the training data. When a string of words in the training data matched a name, this would be reflected in the new features. For example, if *New York* was found both in the locations gazetteer and in the training set, then both *New* and *York* would receive a feature value *Ltag* (for *location tag*) for the newly added location feature. The results in Table 4 show that this strategy was successful.

The results were found using Timbl with default settings.

English devel.	Precision	Recall	$F_{\beta=1}$
Only context	75.88%	82.88%	79.22
With old lists	70.40%	75.73%	72.97
With new lists	77.58%	83.81%	80.58

Table 4: Effect of corpus-specific gazetteers

6.2 Ratio of capitalized to non-capitalized occurrence of tokens

Next, another feature was added to the training and test instances. This feature is another binary feature, and it indicates if the focus word of the instance is found more often in its capitalized form, or in its non-capitalized form. This feature can help the process of NER in different ways. One of them is the identification of sentence-initial words. They are always capitalized in English, but if they tend to appear uncapitalized more often, they are probably not a name. Another way they can help is in finding words which are sometimes names, and sometimes ordinary words (e.g. *Apple*). They should not be tagged as a name if the uncapitalized version occurs more frequently.

This approach was also successful. Results shown in Table 5 were once again obtained by using Timbl with default settings.

English devel.	Precision	Recall	$F_{\beta=1}$
No cap. info	75.88%	82.88%	79.22
With cap. info	77.18%	84.20%	80.54

Table 5: Effect of capitalization/non-capitalization ratio

6.3 Combination of conjunction lists and capitalization information

Finally, all features were combined, and a number of optimization and (limited) feature selection runs were executed. The best run found used all five of the extra features derived from the unannotated data. This is good news, because it means that using unannotated data can help to improve NER of English.

Both results shown in Table 6 are those of the best runs after optimization.

English devel.	Precision	Recall	$F_{\beta=1}$
No extra data	84.09%	85.20%	84.64
With extra data	84.75%	87.06%	85.89

Table 6: Effect of using unannotated data and optimization runs

7 Conclusion

In the plain learning problem (i.e. using only annotated data), our system used only context and word-internal information. Type – token generalization was never beneficial, and gazetteers helped only for the German task.

When using unannotated data, performance was improved in two ways: extra gazetteers were constructed by exploiting conjunctions, and words which appear mostly in capitalized form were set apart from those that do not.

8 Acknowledgements

The authors would like to thank Erik Tjong Kim Sang for his help with data preprocessing, as well as for helpful hints in the construction of the system.

English devel.	Precision	Recall	$F_{\beta=1}$
LOC	87.59%	91.02%	89.27
MISC	84.97%	81.56%	83.23
ORG	74.54%	79.27%	76.83
PER	89.49%	91.53%	90.50
overall	84.75%	87.06%	85.89

English test	Precision	Recall	$F_{\beta=1}$
LOC	77.25%	87.35%	81.99
MISC	71.67%	73.50%	72.57
ORG	70.36%	68.03%	69.18
PER	81.52%	81.01%	81.27
overall	75.84%	78.13%	76.97

German devel.	Precision	Recall	$F_{\beta=1}$
LOC	56.75%	69.43%	62.45
MISC	74.82%	41.78%	53.62
ORG	52.49%	36.58%	43.11
PER	67.74%	50.96%	58.17
overall	61.35%	49.87%	55.02

German test	Precision	Recall	$F_{\beta=1}$
LOC	59.68%	62.22%	60.93
MISC	66.49%	37.01%	47.56
ORG	48.77%	35.83%	41.31
PER	76.67%	61.59%	68.31
overall	63.93%	51.86%	57.27

Table 7: Best results obtained for English using the unannotated data, and for German using only the training data and gazetteers

References

Michael Collins. 2002. Ranking Algorithms for Named-Entity Extraction: Boosting and the Voted Perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 489–496, Philadelphia.

Walter Daelemans and Véronique Hoste. 2002. Evaluation of Machine Learning Methods for Natural Language Processing Tasks. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, pages 755–760, Las Palmas, Gran Canaria.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2002. TiMBL: Tilburg Memory Based Learner, version 4.3, Reference Guide. ILK Technical Report 02-10, ILK. Available from http://ilk.kub.nl/downloads/pub/papers/ilk0210.ps.gz.

Fien De Meulder, Walter Daelemans, and Véronique Hoste. 2002. A Named Entity Recognition System for Dutch. In M. Theune, A. Nijholt, and H. Hondrop, editors, *Computational Linguistics in the Netherlands 2001. Selected Papers from the Twelfth CLIN Meeting*, pages 77–88, Amsterdam – New York. Rodopi.