Memory-based one-step named-entity recognition: Effects of seed list features, classifier stacking, and unannotated data

Iris Hendrickx and **Antal van den Bosch**
ILK / Computational Linguistics
Tilburg University, The Netherlands
{I.H.E.Hendrickx,Antal.vdnBosch}@uvt.nl

1 Outline

We present a memory-based named-entity recognition system that chunks and labels named entities in a one-shot task. Training and testing on CoNLL-2003 shared task data, we measure the effects of three extensions. First, we incorporate features that signal the presence of wordforms in external, language-specific seed (gazetteer) lists. Second, we build a second-stage stacked classifier that corrects first-stage output errors. Third, we add selected instances from classified unannotated data to the training material. The system that incorporates all attains an overall F-rate on the final test set of 78.20 on English and 63.02 on German.

2 Data and features

The CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003) supplied datasets in two languages, English and German, using four named entity categories: persons, organisations, locations, and "miscellany names". Manual annotation has been performed at the University of Antwerp. Apart from tokenized wordforms, the data provides predicted PoS-tags and chunks.

Additionally we computed the following features with each wordform, largely following those used by the best-performing submission of the 2002 shared task (Carreras et al., 2002):

- Orthographic features represented as binary features: *Begin_cap, All_caps, Internal_cap, Contains_digit, Contains_digit_en_alpha, Initial, Lower_case, First_word*
- The wordform's first letter and last three letters (as three separate features)
- The direct output of the memory-based lemmatizer (Van den Bosch and Daelemans, 1999), providing PoS tag, morphological features, and spelling change information
- PoS tag from a slow but accurate version of the memory-based tagger trained on a portion of the British National Corpus, according to the CLAWS-5 tagset (for English data only)

For example, for the English word *Indian* the following feature representation is made: `Indian NNP I-NP 1 0 0 0 0 0 0 0 I i a n AJ0-NN1 N-s I-MISC`, where `NNP` is the provided PoS tag, `I-NP` the chunk tag; the binary features represent the orthographic features (where in this case only *Begin_cap* is positive); `AJ0-NN1` is the PoS tag of the BNC-trained-tagger; `N-s` is the lemmatizer output for noun-singular; the last element, `I-MISC`, is the annotated class label.

In our experiments we construct instances around wordforms, where we take a windowed snapshot of the word in its direct local context. By default, we select a window of two words to the left and right. For all five words in each input instance (feature vector), in principle all of the above features are included.

3 Experimental setup

In two subsections we briefly detail how the memory-based learner works, and how we optimized its parameters through an automatic process called iterative deepening.

3.1 Memory-based learning

Memory-based learning is a supervised inductive learning algorithm for learning classification tasks. Memory-based learning treats a set of training instances as points in a multi-dimensional feature space, and stores them as such in an *instance base* in memory (rather than performing some abstraction over them).

New (test) instances are classified by matching them to all instances in memory, and by calculating with each match the *distance*, given by a distance function between the new instance X and each of the n memory instances $Y_{1...n}$. Classification in memory-based learning is performed by the k-NN algorithm that searches for the k 'nearest neighbours' among the memory instances according to the distance function. The majority class of the k nearest neighbours then determines the class of the new instance X. Cf. (Daelemans et al., 2002) for algorithmic details and background.

3.2 Iterative deepening

Iterative deepening (ID) is a heuristic search algorithm for the optimization of algorithmic parameter and feature selection, that combines classifier wrapping (using the training material internally to test experimental variants) (Kohavi and John, 1997) with progressive sampling of training material (Provost et al., 1999). We start with a large pool of experiments, each with a unique combination of input features and algorithmic parameter settings. In the first step, each attempted setting is applied to a small amount of training material and tested on a fixed amount of held-out data (a held-out part of the training set). Only the best settings are kept; all others are removed from the pool of competing settings. In subsequent iterations, this step is repeated, cutting the number of settings in the pool by a half and retaining the best-performing half, while at the same time doubling the amount of training material.

We selected 10% of the training set as held-out data. Six iterations were performed with increasing training set sizes, starting with 2000 instances, and doubling with each iteration up to 128,000 training instances, resulting in 16 best settings after the last iteration. Selection of the best experiments was based on their overall F-rate as computed by the `conlleval` script.

The initial pool of experiments was created by systematically varying parameters of the memory-based learner and some limited feature selections, (for details, cf. (Daelemans et al., 2002)):

- Basic distance function: Overlap or modified value difference metric (MVDM)
- Feature weighting: gain ratio, information gain, χ^2, or shared variance
- k in the k-NN classifier: 5, 9, 13, 15, 17, 19, 21, 25, and 29
- Distance weighting: none, linear-inverse, inverse, exponential decay with α=1 and α=4
- Feature selection: apart from the wordform and its provided CoNLL-2003 PoS tag, create a local window of either no, 1, or 2 wordforms to the left and right of the focus word. For all words in a window, all features are selected.

The first round of the ID process therefore tests $2 \times 4 \times 9 \times 5 \times 3 = 1080$ systematic permutations of these parameter settings and feature selection.

4 Extensions

4.1 Seed list features

The first extension is to incorporate language-specific seed-list (gazetteer) information. Rather than using these lists external to the classifier, we encode them as internal features associated to wordforms. For each of the four named entity classes we gathered one list of names, containing material garnered from name sites on the internet, from the training set (for the MISC category), and from the CELEX English lexical data base (Baayen et al., 1993). These lists vary in size from 1269 names to 78,732 names. Each wordform in the training and test data is then enriched with four binary features, each representing whether the word occurs in the respective seed list. One problem with seed lists is that a word can occur in more than one seed list, so that more than one of these four bits may be active.

4.2 Second-stage stacking

The second extension is to use second-stage stacking. Stacking in general (Wolpert, 1992) encompasses a class of meta-learning systems that learn to correct errors made by lower-level classifiers. We adopt the particular method pioneered in (Veenstra, 1998) in which classifications of a first memory-based classifier are added as windowed features to the instances presented to the second classifier. Since the second-stage classifier also computes the similarities between instances using these extra features, it is able, in principle, to recognise and correct reoccurring patterns of errors within sub-sentential sequences. This could correct errors made due to the "blindness" of the first-stage classifier, which is unaware of its own classifications left or right of the wordform in the current focus position. We used stacking on top of the first extension.

4.3 Unannotated data

For both languages a large unannotated dataset was made available for extracting data or information. Alternative to using this data to expand or bootstrap seed lists (Cucerzan and Yarowsky, 1999; Buchholz and Van den Bosch, 2000), we use the unannotated corpus to select useful instances to be added directly to the training set. Not unlike (Yarowsky, 1995) we use confidence of our classifier on unannotated data to enrich itself; that is, by adding confidently-classified instances to the memory. We make the simple assumption that entropy in the class distribution in the nearest neighbour set computed in the classification of a new instance is correlated with the reliability of the classification, when $k > 1$. When k nearest neighbours all vote for the same class, the entropy of that class vote is 0.0. Alternatively, when the votes tie, the entropy is maximal.

A secondary heuristic assumption is that it is probably not useful to add (almost) exact matches to the memory, since adding those is likely to have little effect on the performance of the k-NN classifier. More effect can be expected from adding instances to memory that have a low-entropy class distribution in their nearest neighbour set *and* of which the nearest neighbours are at a relatively

	Precision	Recall	$F_{\beta=1}$
English devel.	84.54%	87.16%	85.83
English test	77.01%	80.74%	78.83
German devel.	64.01%	52.29%	57.56
German test	66.71%	56.47%	61.16

Table 1: Overall results (precision, recall, F-rate) of the initial system on the test sets of both languages.

settings	mvdm	feat. weight	k	dist.	w
Eng, initial	yes	gain ratio	21	IL	1
Eng, seedlist	yes	gain ratio	5	ID	1
Ger, initial	yes	gain ratio	21	IL	2
Ger, seedlist	yes	gain ratio	9	ID	2

Table 2: Optimal parameter settings estimated by iterative deepening. "w" stands for window.

large distance. A large distance entails that the instances contains previously unseen feature values (words), and assuming that the predicted class label is correct, these new values can be valuable in matching and therefore classifying new test material better.

We applied our selection method to the first 2 million words of the unannotated English dataset. For German we were able to process 0.25 million words. First we applied the classifier with two extensions, seed list information and second stage stacking, to classify the unannotated data. We selected instances with an entropy in the class distribution lower than 0.05 and a distance of the nearest neighbour of at least 0.1. For English, in total 179,391 instances (9%) were selected from the unannotated dataset and added to the training set. For German. markedly less instances were selected: 467 (0.19%).

5 Results

5.1 Initial classifier: Iterative deepening

Iterative deepening produced estimations of optimal parameter settings for our initial systems for the two languages, displayed in the first and third row of Table 2. With this setting we achieved an overall F-rate of 78.83 for English and 61.16 for German. Table 1 lists the full evaluation results.

5.2 First and second extension: seed list features and stacking

We have also performed iterative deepening in the experiment with the seed list information. This altered the best setting found by the iterative deepening process (the second and fourth rows of Table 2). The results on the English development set are slightly better than the initial system, as can be seen in Table 3. The classifier with

	Precision	Recall	$F_{\beta=1}$
English devel.	85.04%	87.26%	86.14
English test	75.03%	79.75%	77.32
German devel.	65.27%	50.76%	57.11
German test	69.31%	55.70%	61.77

Table 3: Overall results (precision, recall, F-rate) of the system with seed-list features on the test sets of both languages.

	Precision	Recall	$F_{\beta=1}$
English devel.	85.98%	87.63%	86.80
English test	76.26%	80.21%	78.18
German devel.	68.80%	52.29%	59.42
German test	71.19%	56.38%	62.93

Table 4: Overall results (precision, recall, F-rate) of the system with seed-list features and second-stage stacking on the test sets of both languages.

seed list information performs worse on the English test set than the one without seed lists. The reverse effect is seen on the German data. On the development set, using the seed list information gave a slight lower performance, but on the test set it has a slightly positive effect.

Our second extension, stacking, improves on all overall F-scores of both languages as compared to the seed-list extended systems, as shown in Table 4.

5.3 Third extension: Selecting instances from unannotated data

The three extensions, using seed list information, performing second stage stacking and adding information from unannotated data, are combined in the final experiment. This experiment achieves the highest result on the English development set, and on both German test sets, as listed in Table 5. The positive effect of adding selected unannotated data on the German test sets is rather minimal, but we added only a very small amount of unlabeled material. The performance on the English test set is not better than the initial classifier.

6 Discussion

In this paper we have presented a memory-based named-entity recognition system that chunks and labels named entities in one shot. We reported on three extensions; incorporating seed list information, second-stage stacking and adding selected instances from classified unannotated data to the training material.

First, we trained and tested a basic classifier without any of the extensions. Subsequently, we found that (i) incorporating seed list information as binary features does not always help; only in two of the four test sets the seedlists had a positive effect. There can be several explanations for this, such as the quality of the seed lists, the chosen parameter setting from the iterative deepening process or overestimated weights given to the features by the classifier. Due to the tight time schedule we could not further investigate this.

Second, second-stage stacking improves generalisation performance consistently on all test sets as compared to the seed-list extended systems.

Third, only in the final experiment we added selected classified instances from unannotated data. This gave an additional reasonable boost in performance on the English development set, it attains an overall F-rate of 86.97 (an error reduction of 8%) over the initial classifier. The same effect was seen on both German test sets, on which the combination of the three extensions achieved a Fscore of 59.58 (5% error reduction) and 63.02 (5% error reduction). This effect is not seen on the English test set; here the initial classifier performs best. This can partly be explained by the fact that the last two extensions were built upon the first extension, which had a markedly lower score than the initial classifier to begin with.

In sum, our results suggest that two of the three extensions, the stacking method, and the unlabeled instance selection method, have been consistently helpful. Seed list features, however, have not.

English devel.	Precision	Recall	$F_{\beta=1}$
LOC	89.42%	91.13%	90.27
MISC	91.36%	80.26%	85.45
ORG	74.32%	83.30%	78.55
PER	90.16%	91.53%	90.84
Overall	86.16%	87.80%	86.97

English test	Precision	Recall	$F_{\beta=1}$
LOC	80.81%	86.33%	83.48
MISC	66.96%	75.93%	71.16
ORG	69.24%	73.99%	71.54
PER	83.98%	82.00%	82.98
Overall	76.33%	80.17%	78.20

German devel.	Precision	Recall	$F_{\beta=1}$
LOC	61.90%	69.60%	65.52
MISC	83.25%	32.97%	47.23
ORG	67.55%	49.32%	57.01
PER	73.40%	54.96%	62.86
Overall	68.88%	52.49%	59.58

German test	Precision	Recall	$F_{\beta=1}$
LOC	63.36%	64.83%	64.09
MISC	75.60%	32.84%	45.79
ORG	62.90%	48.90%	55.02
PER	83.47%	67.62%	74.71
Overall	71.15%	56.55%	63.02

Table 5: Results on the test sets of the variant combining all three extensions to the initial classifier.

References

R. H. Baayen, R. Piepenbrock, and H. van Rijn. 1993. *The CELEX lexical data base on CD-ROM*. Linguistic Data Consortium, Philadelphia, PA.

S. Buchholz and A. Van den Bosch. 2000. Integrating seed names and n-grams for a named entity list and classifier. In *LREC-2000 (Second International Conference on Language Resources and Evaluation) Proceedings. Vol. II*, pages 1215–1221.

X. Carreras, L. Marques, and L. Padro. 2002. Named entity extraction using AdaBoost. In *Proceedings of CoNLL-2002*, pages 167–170.

S. Cucerzan and D. Yarowsky. 1999. Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of 1999 Joint SIGDAT Conference on EMNLP and VLC*.

W. Daelemans, J. Zavrel, K. Van der Sloot, and A. Van den Bosch. 2002. TiMBL: Tilburg Memory Based Learner, version 4.3, reference guide. Technical Report ILK-0210, ILK, Tilburg University.

R. Kohavi and G. John. 1997. Wrappers for feature subset selection. *Artificial Intelligence Journal*, 97(1–2):273–324.

F. Provost, D. Jensen, and T. Oates. 1999. Efficient progressive sampling. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pages 23–32.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*. Edmonton, Canada.

A. Van den Bosch and W. Daelemans. 1999. Memory-based morphological analysis. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 285–292, New Brunswick, NJ. ACL.

J. Veenstra. 1998. Fast np chunking using memory-based learning techniques. In *Proceedings of Benelearn 1998*, pages 71–79.

D. H. Wolpert. 1992. On overfitting avoidance as bias. Technical Report SFI TR 92-03-5001, The Santa Fe Institute.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL33*, pages 189–196, Cambridge, MA.