Two-stage named-entity recognition using averaged perceptrons

Lars Buitinck and Maarten Marx

Information and Language Processing Systems
Informatics Institute
University of Amsterdam
{`l.j.buitinck,maartenmarx`}`@uva.nl`

Abstract. We describe a simple approach to named-entity recognition (NER), aimed initially at the Dutch language, but potentially applicable to other languages. Our NER system employs a two-stage architecture, with handcrafted but dataset-independent features for both stages, and is on a par with state-of-the-art systems described in the literature. Notably, our approach does not depend on language-specific assets such as gazetteers. The resulting system is quite fast and is implemented in less than 500 lines of code.

1 Introduction

Named-entity recognition (NER) is the problem of detecting, in running text, all names or *definite descriptors*, including the names of persons, organizations, geographical entities, etc. We describe a machine-learned approach to this problem for the Dutch language, for which no publicly available NER software exists. Our approach, however, makes no language-specific assumptions and should be portable to similar languages. We intend to use the results of NER tagging to enhance searching in parliamentary hansards; thus, in addition to a full-text query, users will get the opportunity to look for all occurrences of a certain person's or organisation's name.

1.1 Problem description

In most machine-learned approaches to NER (Tjong Kim Sang, 2002), it is considered a sequence labeling problem,[1] where tokens are annotated with `BIO`-tags, indicating *beginning* of, *inside*, or *outside* a name. In addition, the names are annotated with a category. The following sentence from the SoNaR (Oostdijk et al., 2008) corpus exemplifies the NER problem.

> Bij de stad [LOC Falujah] is een [LOC Amerikaanse] [PRO Black Hawk]-helikopter neergestort.

[1] But see Sarawagi and Cohen (2004) for an argument against this view, and for a view of NER as a *segmentation* problem.

We note that the recognition of names and their classification are really two distinct problems, and tackle the two using separate classification algorithms, as described in more detail in section 2.

1.2 Related work

An two-stage approach to NER was tried and found effective by Wang and Patrick (2009). The difference between our approach and theirs is that Wang and Patrick use a conditional random field (CRF) to do a first attempt at solving the complete problem of detection and classification, followed by a voting procedure to re-classify the detected entity names.

Similarly, Desmet and Hoste (2010) use a combination of various classifiers (CRF, SVM, *k*-NN) combined into a voting ensemble with voting weights learned by a genetic algorithm; they obtain good results on a Dutch NER task, but conclude that "[t]he performance gain of the ensemble system over the best individual classifier [...] is not very large and comes at a high computational cost." We avoid this cost by using a single linear classifier in each stage of our system.

2 Architecture

We break up the NER problem into two stages and use a two-stage architecture to solve it. The first stage is the recognition stage, where we try to find the actual entity references in running text. The second stage is that of classification, where the entity references are divided into various classes; the exact classes depend on the training and evaluation sets in use (see section 3).

In both stages, we use the averaged perceptron algorithm, a simplification of the voted perceptron algorithm of Freund and Schapire (1999), to train a linear classifier for the problem at hand, using a handcrafted set of features detailed below. We train our perceptrons for 40 iterations.

Our implementation language is the Learning Based Java (LBJ) language and assorted libraries (Rizzolo and Roth, 2010). By leveraging these, we have managed to implement our full system in less than 500 lines of code.

In the rest of this section, we discuss the features used in both stages of our NER pipeline. It should be noted that none of these are particularly new (Bogers, 2004, pp. 29–31), although we are not aware of NER systems that use separate feature sets for recognition and classification of entity names.

2.1 Recognition stage

In the recognition stage, we use the following basic features to determine the BIO tag of a single token:

– The identity of tokens in a size-five window, both with and without their relative position to the word under scrutiny. E.g., the word "Obama" in the

window "all of Obama's plans are" would yield token features "all", "of", "Obama's", "plans" and "are", as well as token-at-position features "-2: all", "-1: of", "0: Obama's", "+1: plans" and "+2: are".
- The part-of-speech tags of the above (assumed part of the training and test data).
- The conjunction of the previous two features; i.e. word identity+POS as a combined feature with window size of five.
- Boolean features for several patterns (regular expressions) of the token under scrutiny: first letter capitalized, all capitals, contains digit, all digits, contains dash (–), contains dash followed by number and a limited matcher for Roman numerals, `^[IVX]+$`.
- Capitalization features, as in the previous bullet point, for a token context window.
- Prefixes and suffixes of the current and previous token, up to four characters in length.

We extend this feature set with the classifier's output to the two previous tokens to turn our token-level classifier into a sequence classifier. We also add the conjunction of the previous two predictions, the conjunction of capitalization pattern and the previous prediction, and the conjunction of word window features with the previous prediction. Preliminary experiments with smaller feature sets showed a strong increase in classifier accuracy when introducing these conjunctive features.

2.2 Classification stage

In the classification stage, we assign a class label to a complete entity name as discovered by the recognition stage. We therefore have a different type of inputs than in the previous stage: sequences of tokens in context instead of single tokens, e.g.

. . . de stad [Falujah] is een . . .

Hence, we use a different set of features, comprising:

- Token identity inside the sequence.
- Identity of the two tokens preceding the sequence and the tokens following it.
- Affixes, up to length four, of the tokens inside the sequence.
- The length of the sequence.
- The capitalization pattern of the sequence, expressed as a string on the alphabet $(L|U|O)*$ to indicate alternations of upper and lower case initial letters (with O denoting not a letter).
- The occurrence of digits, dashes and all-capital tokens inside the sequence.

Note that we do not employ any so-called "gazetteer" features (lookup of entity names in a pre-compiled list). We chose not to use a gazetteer since no high-quality gazetteer was available for the Dutch language and available time prohibited us from compiling a gazetteer ourselves.

3 Evaluation

We evaluate our approach on two different datasets. The CoNLL'02 dataset (Tjong Kim Sang, 2002) consists of 309.686 tokens, containing 19901 names, split into 65% training, 22% validation and 12% test material (measured by the number of tokens). CoNLL'02 divides named entities into four categories: PER (person), LOC (location), ORG (organisation) and MISC (all other).

Table 1 shows our results on the CoNLL'02 test set. We note that the overall F_1 score is just below that of Wu et al. (2002), the second-best result reported in the CoNLL'02 shared task proceedings.

Table 1. Results on the CoNLL'02 test set

	Precision	Recall	F_1
LOC	80.11%	77.52%	78.79
MISC	77.09%	70.85%	73.84
ORG	73.68%	63.49%	68.21
PER	73.39%	85.15%	78.84
Overall	75.79%	74.50%	75.14

For reference, we show results for the Stanford Named Entity Recognizer (Finkel et al., 2005) on the CoNLL'02 test set (retrained on the CoNLL training set) in table 2;[2] re-using the Stanford NER package is a common choice for practical named entity recognition in the Dutch language area. We note that our system performs slightly better on this set, despite being smaller and simpler than Stanford's.

Table 2. Stanford NER results on the CoNLL'02 test set

	Precision	Recall	F_1
LOC	84.26%	74.68%	79.18
MISC	73.27%	65.12%	68.96
ORG	76.63%	66.55%	71.24
PER	73.56%	87.16%	79.78
Overall	76.02%	73.46%	74.72

Our second dataset is that used by Desmet and Hoste (2010, 2011), a selection from the SoNaR corpus (Oostdijk et al., 2008): 206421 tokens containing 13818 names, taken from Dutch newscast autocue text. SoNaR divides named entities into six categories: PER, LOC, ORG, PRO (product), EVE (event) and MISC.

[2] Courtesy Andrei Vishneuski, U. Amsterdam.

Table 3 shows our results on the SoNaR set, obtained by a stratified three-fold cross validation. We use the exact same division of the dataset into three folds as was used by Desmet and Hoste (2010). We note that the overall F_1 score is slightly lower than the 84.44 reported by Desmet and Hoste (2011) for their classifier ensemble trained by a genetic algorithm, but on a par with the 83.77 for their CRF classifier.

From table 3, it is obvious that our system's performance is somewhat lacking with respect to the classes PRO, EVE and MISC; we attribute this to two factors. First is the fact that our features have been optimized for the CoNLL'02 dataset, where two of these categories are largely subsumed by the MISC class. Second is the fact that these categories are relatively less well-represented in the data; each occurs less than a thousand times, while the categories LOC, PER and ORG occur 6624, 3290 and 2461 times, respectively.

Table 3. Results on the SoNaR set (macro-averaged over 3-fold cross validation)

	Precision	Recall	F_1
EVE	86.80%	60.08%	70.99
LOC	87.22%	92.14%	89.61
MISC	73.20%	53.91%	62.07
ORG	80.38%	77.89%	79.09
PER	82.43%	82.19%	82.30
PRO	70.51%	48.49%	57.46
Overall	83.95%	83.17%	83.56

4 Conclusion

We have shown that a two-stage approach with careful feature engineering, using a standard classification algorithm, can result in a named-entity recognizer that is competitive with state-of-the-art systems for the Dutch language, even without the use of gazetteers. In particular, our "poor man's approach" to sequence classification is on a par with specialized models such as conditional random fields.

Our system, however, is at present weaker than it could be when evaluated on the new SoNaR NER-tagged corpus. Further feature engineering could be used to improve its performance on this data set with its fine-grained classification.

5 Acknowledgments

The research for this paper was funded by Clarin-NL (`http://www.clarin.nl`) under project number 10-009 (WiP).

Bibliography

Toine Bogers. Dutch named entity recognition: Optimizing features, algorithms, and output. Master's thesis, Tilburg University, 2004.

Bart Desmet and Véronique Hoste. Dutch named entity recognition using classifier ensembles. In *Proc. 20th Meeting of CLIN*, pages 29–41, 2010.

Bart Desmet and Véronique Hoste. Dutch named entity recognition using classifier ensembles. In *Proc. 23rd Benelux Conference on Artificial Intelligence*, 2011.

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. 43rd Annual Meeting of the ACL*, pages 363–370, 2005.

Yoav Freund and Robert E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

N. Oostdijk, M. Reynaart, P. Monachesi, G. Van Noord, R. Ordelman, I. Schuurman, and V. Vandeghinste. From D-Coi to SoNaR: A reference corpus for Dutch. In *Proc. Int'l Conf. on Language Resources and Evaluation (LREC)*, 2008.

Nicholas Rizzolo and Dan Roth. Learning Based Java for rapid development of NLP systems. In *Proc. Int'l Conf. on Language Resources and Evaluation (LREC)*, 2010.

S. Sarawagi and W.W. Cohen. Semi-Markov conditional random fields for information extraction. *Advances in Neural Information Processing Systems*, 17: 1185–1192, 2004.

Erik F. Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In Dan Roth and Antal van den Bosch, editors, *Proc. 6th Conf. on Computational Natural Language Learning (CoNLL)*, pages 155–158, 2002.

Yefeng Wang and Jon Patrick. Cascading classifiers for named entity recognition in clinical notes. In *Proc. Workshop on Biomedical Information Extraction (WBIE)*, pages 42–49, 2009.

Dekai Wu, Grace Ngai, Marine Carpuat, Jeppe Larsen, and Yongsheng Yang. Boosting for named entity recognition. In Dan Roth and Antal van den Bosch, editors, *Proc. 6th Conf. on Computational Natural Language Learning (CoNLL)*, 2002.