Named Entity Recognition as a House of Cards: Classifier Stacking

Radu Florian
Department of Computer Science and Center for Language and Speech Processing
Johns Hopkins University
3400 N. Charles St., Baltimore, MD 21218, USA
rflorian@cs.jhu.edu

1 Introduction

This paper presents a classifier stacking-based approach to the named entity recognition task (NER henceforth). Transformation-based learning (Brill, 1995), Snow (sparse network of winnows (Muñoz et al., 1999)) and a forward-backward algorithm are stacked (the output of one classifier is passed as input to the next classifier), yielding considerable improvement in performance. In addition, in agreement with other studies on the same problem, the enhancement of the feature space (in the form of capitalization information) is shown to be especially beneficial to this task.

2 Computational Approaches

All approaches to the NER task presented in this paper, except the one presented in Section 3, use the IOB chunk tagging method (Tjong Kim Sang and Veenstra, 1999) for identifying the named entities.

2.1 Feature Space and Baselines

A careful selection of the feature space is a very important part of classifier design. The algorithms presented in this paper are using only information that can be extracted directly from the training data: the words, their capitalization information and the chunk tags. While they can definitely incorporate additional information (such as lists of countries/cities/regions, organizations, people names, etc.), due to the short exposition space, we decided to restrict them to this feature space.

Table 2 presents the results obtained by running off-the-shelf part-of-speech/text chunking classifiers; all of them use just word information, albeit in different ways. The leader of the pack is the MXPOST tagger (Ratnaparkhi, 1996). The measure of choice for the NER task is F-measure, the harmonic mean of precision and recall: $F_\beta = 2\frac{\beta^2 P \cdot R}{\beta^2 P + R}$, usually computed with $\beta = 1$.

As observed by participants in the MUC-6 and -7 tasks (Bikel et al., 1997; Borthwick, 1999; Miller et al., 1998), an important feature for the NER task is information relative to word capitalization. In an approach similar to Zhou and Su (2002), we extracted for each word a 2-byte code, as summarized in Table 1. The first byte specifies the capitalization of the word (first letter capital, etc), while the second specifies whether the word is present in the dictionary in lower case, upper case, both or neither forms. These two codes are extracted in order to offer both a way of backing-off in sparse data cases (unknown words) and a way of encouraging generalization. Table 2 shows the performance of the fnTBL (Ngai and Florian, 2001) and Snow systems when using the capitalization information, both systems displaying considerably better performance.

1: Capitalization information	2: Presence in dictionary
first_cap, all_caps, all_lower, number, punct, other	upper, lower, both, none

Table 1: Capitalization information

2.2 Transformation-Based Learning

Transformation-based learning (TBL henceforth) is an error-driven machine learning technique which works by first assigning an initial classification to the data, and then automatically proposing, evaluating and selecting the *transformations* that maximally decrease the number of errors. Each such transformation, or rule, consists of a *predicate* and a *target*. In our implementation of TBL – fnTBL – predicates consist of a conjunction of atomic predicates, such as feature identity (e.g. $word_0 = Barcelona$), membership in a set (e.g. B – ORG $\in \{chunk_{-3} \ldots chunk_{-1}\}$), etc.

TBL has some attractive qualities that make it suitable for the language-related tasks: it can automatically integrate heterogenous types of knowledge, without the need for explicit modeling (similar to Snow, Maximum Entropy, decision trees, etc); it is error–driven, therefore directly minimizes the

Method	Accuracy	$F_{\beta=1}$
without capitalization information		
TnT	94.78%	66.72
MXPOST	**95.02%**	**69.04**
Snow	94.27%	65.94
fnTBL	94.92%	68.06
with capitalization information		
Snow (extended templates)	95.15%	71.36
fnTBL	**95.57%**	71.88
fnTBL+Snow	95.36%	**73.49**

Table 2: Comparative results for different methods on the Spanish development data

ultimate evaluation measure: the error rate; and it has an inherently dynamic behavior[1]. TBL has been previously applied to the English NER task (Aberdeen et al., 1995), with good results.

The fnTBL-based NER system is designed in the same way as Brill's POS tagger (Brill, 1995), consisting of a morphological stage, where unknown words' chunks are guessed based on their morphological and capitalization representation, followed by a contextual stage, in which the full interaction between the words' features is leveraged for learning. The feature templates used are based on a combination of word, chunk and capitalization information of words in a 7-word window around the target word. The entire template list (133 templates) will be made available from the author's web page after the conclusion of the shared task.

2.3 Snow

Snow – Sparse Network of Winnows – is an architecture for error-driven machine learning, consisting of a sparse network of linear separator units over a common predefined or incrementally learned feature space. The system assigns weights to each feature, and iteratively updates these weights in such a way that the misclassification error is minimized. For more details on Snow's architecture, please refer to Muñoz et al. (1999).

Table 2 presents the results obtained by Snow on the NER task, when using the same methodology from Muñoz et al. (1999), with the their templates[2] and with the same templates as fnTBL.

[1]The quality of chunk tags evolves as the algorithm progresses; there is no mismatch between the quality of the surrounding chunks during training and testing.

[2]In this experiment, we used the feature patterns described in Muñoz et al. (1999): a combination of up to 2 words in a 3-word window around the target word and a combination of up to 4 chunks in a 7-word window around the target word. All throughout the paper, Snow's default parameters were used.

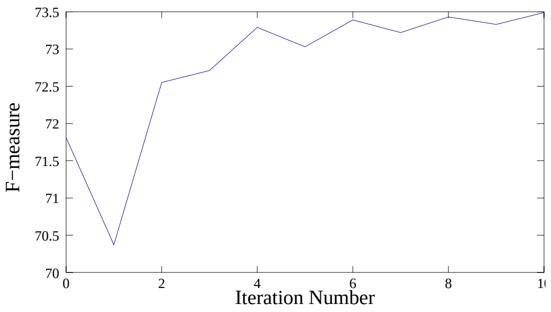

Figure 1: Performance of applying Snow to TBL's output, plotted against iteration number

2.4 Stacking Classifiers

Both the fnTBL and the Snow methods have strengths and weaknesses:

- fnTBL's strength is represented by its dynamic modeling of chunk tags – by starting in a simple state and using complex feature interactions, it is able to reach a reasonable end-state. Its weakness consists in its acute myopia: the optimization is done greedily for the local context, and the feature interaction is observed only in the order in which the rules are selected.
- Snow's strength consists in its ability to model interactions between the *all* features associated with a sample. However, in order to obtain good results, the system needs reliable contextual information. Since the approach is not dynamic by nature, good initial chunk classifications are needed.

One way to address both weaknesses is to combine the two approaches through stacking, by applying Snow on fnTBL's output. This allows Snow to have access to reasonably reliable contextual information, and also allows the output of fnTBL to be corrected for multiple feature interaction. This stacking approach has an intuitive interpretation: first, the corpus is dynamically labeled using the most important features through fnTBL rules (coarse-grained optimization), and then is fine-grained tuned through a few full-feature-interaction iterations of Snow.

Table 2 contrasts stacking Snow and fnTBL with running either fnTBL or Snow in isolation - an improvement of 1.6 F-measure points is obtained when stacking is applied. Interestingly, as shown in Figure 1, the relation between performance and Snow-iteration number is not linear: the system initially takes a hit as it moves out of the local fnTBL maximum, but then proceeds to increase its performance,

Method	Accuracy	$F_{\beta=1}$
Spanish	98.42%	90.26
Dutch	98.54%	88.03

Table 3: Unlabeled chunking results obtained by fnTBL on the development sets

finally converging after 10 iterations to a F-measure value of 73.49.

3 Breaking-Up the Task

Muñoz et al. (1999) examine a different method of chunking, called Open/Close (O/C) method: 2 classifiers are used, one predicting open brackets and one predicting closed brackets. A final optimization stage pairs open and closed brackets through a global search.

We propose here a method that is similar in spirit to the O/C method, and also to Carreras and Màrquez (2001), Arévalo et al. (2002):

1. In the first stage, detect only the entity boundaries, without identifying their type, using the fnTBL system[3];
2. Using a forward-backward type algorithm (FB henceforth), determine the most probable type of each entity detected in the first step.

This method has some enticing properties:

- Detecting only the entity boundaries is a simpler problem, as different entity types share common features; Table 3 shows the performance obtained by the fnTBL system – the performance is sensibly higher than the one shown in Table 2;
- The FB algorithm allows for a global search for the optimum, which is beneficial since both fnTBL and Snow perform only local optimizations;
- The FB algorithm has access to both entity-internal and external contextual features (as first described in McDonald (1996)); furthermore, since the chunks are collapsed, the local area is also larger in span.

The input to the FB algorithm consists of a series of chunks $C_1, \ldots, C_n$, each spanning a sequence of words

$$w_1 \ldots w_{b_1-1} \underbrace{w_{b_1} \ldots w_{e_1}}_{C_1} \ldots \underbrace{w_{b_n} \ldots w_{e_n}}_{C_n} \ldots w_m$$

[3]For this task, Snow does not bring any improvement to the fnTBL's output.

Method	Spanish	Dutch
FB performance	76.49	73.30
FB on perfect chunk breaks	83.52	81.30

Table 4: Forward-Backward results (F-measure) on the development sets

For each marked entity C_j, the goal is to determine its most likely type:[4]

$$\begin{aligned}\hat{E}_j = \; & \arg\max_{E_j} \sum_{E_1^{j-1}, E_{j+1}^n} P\left(E_1^n | w_1^m\right) = \\ & \arg\max_{E_j} \sum_{E_1^{j-1}, E_{j+1}^n} P\left(w_1^{b_1-1} E_1 \ldots E_n w_{e_n+1}^m\right) \cdot \\ & \qquad P\left(len\left(w_{b_j}^{e_j}\right)|E_j\right) \cdot P\left(w_{b_j}^{e_j}|E_j\right)\end{aligned} \tag{1}$$

where $P\left(w_1^{b_1-1} E_1 \ldots E_n w_{e_n+1}^m\right)$ represents the entity-external/contextual probability, and $P\left(len\left(w_{b_j}^{e_j}\right)|E_j\right) P\left(w_{b_j}^{e_j}|E_j\right)$ is the entity-internal probability. These probabilities are computed using the standard Markov assumption of independence, and the forward-backward algorithm[5]. Both internal and external models are using 5-gram language models, smoothed using the modified discount method of Chen and Goodman (1998). In the case of unseen words, backoff to the capitalization tag is performed: if w_k is unknown, $P\left(w_k|E_j\right) = P\left(capit\left(w_k\right)|E_j\right)$. Finally, the probability $P\left(len\left(w_{b_j}^{e_j}\right)|E_j\right)$ is assumed to be exponentially distributed.

Table 4 shows the results obtained by stacking the FB algorithm on top of fnTBL. Comparing the results with the ones in Table 2, one can observe that the global search does improve the performance by 3 F-measure points when compared with fnTBL+Snow and 5 points when compared with the fnTBL system. Also presented in Table 4 is the performance of the algorithm on perfect boundaries; more than 6 F-measure points can be gained by improving the boundary detection alone. Table 5 presents the detailed performance of the FB algorithm on all four data sets, broken by entity type.

A quick analysis of the results revealed that most errors were made on the unknown words, both in

[4]We use the notation $w_1^m = w_1 \ldots w_m$.

[5]It is notable here that the best entity type for a chunk is computed by selecting the best entity in all combinations of the other entity assignments in the sentence. This choice is made because it reflects better the scoring method, and makes the algorithm more similar to the HMM's forward-backward algorithm (Jelinek, 1997, chapter 13) rather than the Viterbi algorithm.

Spanish and Dutch: the accuracy on known words is 97.4%/98.9% (Spanish/Dutch), while the accuracy on unknown words is 83.4%/85.1%. This suggests that lists of entities have the potential of being extremely beneficial for the algorithm.

4 Conclusion

In conclusion, we have presented a classifier stacking method which uses transformation-based learning to obtain a course-grained initial entity annotation, then applies Snow to improve the classification on samples where there is strong feature interaction and, finally, uses a forward-backward algorithm to compute a global-best entity type assignment. By using the pipelined processing, this method improves the performance substantially when compared with the original algorithms (fnTBL, Snow+fnTBL).

5 Acknowledgements

The author would like to thank Richard Wicentowski for providing additional language resources (such as lemmatization information), even if they were ultimately not used in the research, David Yarowsky for his support and advice during this research, and Cristina Nita-Rotaru for useful comments. This work was supported by NSF grant IIS-9985033 and ONR/MURI contract N00014-01-1-0685.

Spanish devel	precision	recall	$F_{\beta=1}$
LOC	70.44%	83.45%	76.39
MISC	53.20%	63.60%	57.93
ORG	78.35%	73.00%	75.58
PER	86.28%	84.37%	85.31
overall	75.41%	77.60%	**76.49**

Spanish test	precision	recall	$F_{\beta=1}$
LOC	82.06%	79.34%	80.68
MISC	59.71%	61.47%	60.58
ORG	78.51%	78.29%	78.40
PER	82.94%	89.93%	86.29
overall	78.70%	79.40%	**79.05**

Dutch deve	precision	recall	$F_{\beta=1}$
LOC	81.15%	74.16%	77.50
MISC	72.02%	74.53%	73.25
ORG	79.92%	60.97%	69.17
PER	66.18%	84.04%	74.05
overall	73.09%	73.51%	**73.30**

Dutch test	precision	recall	$F_{\beta=1}$
LOC	86.69%	77.69%	81.94
MISC	75.21%	68.80%	71.86
ORG	74.68%	66.59%	70.40
PER	69.39%	86.05%	76.83
overall	75.10%	74.89%	**74.99**

Table 5: Results on the development and test sets in Spanish and Dutch

References

J. Aberdeen, D. Day, L. Hirschman, P. Robinson, and M. Vilain. 1995. Mitre: Description of the Alembic system used for MUC-6. In *Proceedings of MUC-6*, pages 141–155.

M. Arévalo, X. Carreras, L. Màrquez, M. A. Martí, L. Padró, and M. J. Simón. 2002. A proposal for wide-coverage Spanish named entity recognition. Technical Report LSI-02-30-R, Universitat Politècnica de Catalunya.

D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97*, pages 194–201.

A. Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University.

E. Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

X. Carreras and L. Màrquez. 2001. Boosting trees for clause splitting. In *Proceedings of CoNNL'01*.

S. Chen and J. Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University.

F. Jelinek, 1997. *Information Extraction From Speech And Text*. MIT Press.

D. McDonald, 1996. *Corpus Processing for Lexical Aquisition*, chapter Internal and External Evidence in the Identification and Semantic Categorization of Proper Names, pages 21–39. MIT Press.

S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schwarz, R. Stone, and R. Weischedel. 1998. Bbn: Description of the SIFT system as used for MUC-7. In *MUC-7*.

M. Muñoz, V. Punyakanok, D. Roth, and D. Zimak. 1999. A learning approach to shallow parsing. Technical Report 2087, Urbana, Illinois.

G. Ngai and R. Florian. 2001. Transformation-based learning in the fast lane. In *Proceedings of NAACL'01*, pages 40–47.

A. Ratnaparkhi. 1996. A maximum entropy model for part of speech tagging. In *Proceedings EMNLP'96*, Philadelphia.

E. F. Tjong Kim Sang and J. Veenstra. 1999. Representing text chunks. In *Proceedings of EACL'99*.

G. D. Zhou and J. Su. 2002. Named entity recognition using a HMM-based chunk tagger. In *Proceedings of ACL'02*.