Entity Extraction Without Language-Specific Resources

Paul McNamee **James Mayfield**
The Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road, Laurel, Maryland 20723-6099 USA
{mcnamee,mayfield}@jhuapl.edu

Abstract

We describe a named-entity tagging system that requires minimal linguistic knowledge and thus may be applied to new target languages without significant adaptation. To maintain a language-neutral posture, the system is linguistically naïve, and in fact, reduces the tagging problem to supervised machine learning. A large number of binary features are extracted from labeled data to train classifiers and computationally expensive features are eschewed. We have initially focused our attention on linear support vectors machines (SVMs); SVMs are known to work well when a large number of features is used as long as the individual vectors are sparse. We call our system SNOOD (Hopkins APL Inductive Retargetable Named Entity Tagger).

1 Introduction

Text processing research at the Johns Hopkins University Applied Physics Laboratory (JHU/APL) has focused on information retrieval tasks. We have built a language-independent retrieval engine, HAIRCUT, to explore effective, knowledge-light approaches to the multilingual text retrieval problem (McNamee et al., 2001). Through participation in cross-language evaluation conferences like TREC, CLEF, and NTCIR, we have evaluated retrieval effectiveness over document collections containing Arabic, Chinese, Dutch, English, Finnish, French, German, Italian, Japanese, Korean, Spanish, and Swedish. We have found that a system based on weak-methods can achieve high retrieval effectiveness, both monolingually and translingually, in all of these languages. HAIRCUT uses a statistical language model of retrieval and simple tokenization methods such as unnormalized word forms and character n-grams. We approached the CoNLL-2002 shared task in Language-Independent Named Entity Recognition with the same philosophy of attempting to maximize performance with judicious use of simple, language-neutral methods; the promise of such approaches has previously been demonstrated for this task (Cucerzan and Yarowsky, 1999).

2 Approach

Our initial approach was to cast the labeling problem as a binary decision problem of determining whether the current token belongs to one of the eight classes (e.g., B-LOC). Thus we built eight individual classifiers. The algorithm, features, and learning parameters were identical for each classifier. For each, we used a linear kernel support vector machine that was trained using all positive and negative exemplars from the training set. The labels for this problem are mutually exclusive of one another, so we applied a rudimentary deterministic method to select a single label when multiple classifiers predicted membership in multiple categories.

We computed features pertaining to each token, and used a window of w tokens, with x tokens before and y tokens following the current token. Our submitted results used $w=7$ and $x=y=3$; we examined other window sizes briefly, but did not find significant differences. Binary features were used and were of two kinds: those derived from orthography and punctuation and those related to the language under question. For each token, the basic features were:

- Token length equal to 0-9 (10)
- Token length between 10 and 15, or greater than 15 (2)
- Whether the token was a comma, a full stop, a question mark, an exclamation mark, or other punctuation (5)
- If the token contained all digits, letters and

digits, all capitals, any internal capitalization, and an initial capital (5)

- Whether the 1st/2nd/nth-1/nth character exists (4 x 1)
- Whether the 1st/2nd/nth-1/nth character is ASCII code 33, ..., 90 (4 x 58)

for a total of 258 features. The use of leading and trailing characters was motivated by a desire to use morphological clues, such as if a word ended in 'ly' or 'ing'. In a non-alphabetic language, different features would be required, for example, in Chinese the stroke count of a character might be used, or perhaps a transliterated form such as that produced by the Pinyin system could be applied first. In Japanese, the kana (hiragani and katakana) would be useful. This rudimentary approach would less applicable with infix morphology, but consonant triples might be useful for a Semitic language like Arabic.

One thousand features that were specific to the language under consideration were also used:

- If the token is the ith most frequent word in the language

Very common words were used as an approximation to more grammatically rich features, such as whether the token under consideration is a closed class word or a particular part of speech. In total, 1258 features were used for each token, and a window of 7 tokens was used, so each classifier used exemplars with 8806 binary features. The Spanish training set contained roughly 270k tokens, and about 33k labels (see Table 1).

	LOC	PER	ORG	MISC
B	4913	4321	7390	2173
I	1891	3903	4992	3212
ALL	6804	8224	12382	5385

Table 1: Spanish training exemplars by class.

The Dutch training data consists of roughly 220k tokens, and about 19k labels (see Table 2).

	LOC	PER	ORG	MISC
B	3192	4734	2091	3319
I	466	2880	1183	1381
ALL	3658	7614	3274	4700

Table 2: Dutch training exemplars by class.

3 Learning Algorithm

Many empirical approaches to information extraction have examined, including hidden Markov Models (Seymore et al., 1999), Maximum Entropy Models (McCallum et al., 2000), and Support Vector Machines (Kudo and Matsumoto, 2001). Support Vector Machines (SVMs) are known to be good choices for sparse, high-dimensional problems. A typical approach is to minimize error by maximizing the margin between two classes. This is done by identifying a hyperplane that separates the classes; support vectors of examples close to the other class are used to define the separating hyperplane.

Linear kernels appear to be applicable to text classification problems and we investigate their use in this study; polynomials and radial-basis functions are other popular choices. We used a Java Native Interface (JNI) API with the SVM-Light toolkit (Joachims, 2002). Various learning parameters can significantly affect the performance of the resulting classifiers. With these data, there are many more negative then positive examples, so it is appropriate to give higher importance to learning the positive class. This can be achieved by increasing the cost ratio parameter, J (we used J=*{1,3}* for our models). Multiclass SVMs are being developed, and these might be better suited for this problem. Combining Multiple Binary Classifiers To produce a single label for each token, we first identified the set, S, of possible labels predicted by the classifiers. If S was empty, the "O" label was assigned. Otherwise, the most frequent category was determined (ORG, PER, LOC, or MISC) and selected; if both a beginning and inside tag were in S, the beginning tag was chosen. More sophisticated methods are almost certainly warranted; this scheme can produce invalid sequences, for example, where an inside tag might precede an initial one, such as: "Estados I-ORG" followed by "Unidos B-ORG".

4 Results

We developed our methods based on the Spanish training and test data files (esp.train and esp.testa) and then utilized the same methods and parameter settings when building classifiers for the Dutch data. For reasons of symmetry we did not wish to use the part-of-speech indicators in the Dutch text; these were removed with a simple *awk* script.

5 Influence of Learning Parameters

When using both types of features and a linear kernel SVM with J set to 1.0, the evaluation script reported:

Spanish dev.	Precision	Recall	$F_{\beta=1}$
LOC	78.68%	43.45%	55.98
MISC	7.50%	0.67%	1.24
ORG	65.47%	48.41%	55.66
PER	77.43%	55.32%	64.53
Overall	71.11%	44.35%	54.63

Table 3: Results with J=1 for esp.testa

Clearly the MISC class has not been learned well at all. Since there are so many more negative examples than positive ones, the machine can focus on identifying only the easy positive cases and still achieve low error. We have begun examining the use of different learning parameters, in particular, the effects of changing the cost ratio. Using J = 3.0, we observed some improvement (see Table 4).

Spanish dev.	Precision	Recall	$F_{\beta=1}$
LOC	58.36%	59.90%	59.12
MISC	20.00%	9.44%	12.82
ORG	52.90%	65.53%	58.54
PER	68.13%	67.51%	67.82
Overall	56.65%	59.08%	57.84

Table 4: Improvement when J=3 was used

In this case, the MISC category is still difficult to learn. On the whole, recall is enhanced in the other categories, but at a cost in precision.

Many of the mistakes that the system makes are due to mis-tagging an entity; that is, an entity is labeled with one tag when another was appropriate. For example, distinctions were made in the test set where the country Brazil could refer to either an organization (e.g., the government of Brazil) or a location, depending on context; these can be difficult to learn. When the problem is reduced to simply tagging a word or not, we see much better performance. This suggests that additional effort to establish proper tags would be beneficial.

6 Contribution of Common Word Features

To assess the benefit from using common words as features, we attempted the task using only the basic orthographic and punctuation features. As expected, performance was worse when these features were not included (Table 5). We selected 1000 features expecting that this number would be sufficient to include most closed class words.

Spanish dev.	Precision	Recall	$F_{\beta=1}$
LOC	47.80%	61.73%	53.88
MISC	11.52%	14.83%	12.97
ORG	48.35%	62.06%	54.35
PER	60.43%	71.36%	65.44
Overall	47.55%	59.77%	52.96

Table 5: Baseline performance when common words were not used as features. Compare with the results in Table 4.

7 Chunk Detection

In an attempt to improve overall performance, we explored whether separate classifiers could be combined for different portions of the problem. We trained a classifier to detect when a given token should be labeled at all, and used all labeled tokens as positive examples and the unlabelled tokens as negative examples. We then used a second processing phase where the decision problem was to identify the correct label, given that some label should be assigned. The problem of detecting chunks is far simpler as is revealed in the following table:

Our hope was that separation of the two problems would lead to better overall performance. Using a J = 3.0 classifier to identify tokens which should be labeled, we then made a decision about the correct label type (LOC/MISC/ORG/PER), and then made assignments for beginning or internal tags. We also changed our method for combining multi-

Spanish dev.	Precision	Recall	$F_{\beta=1}$
J = 1	85.97%	85.75%	85.86
J = 3	84.91%	88.32%	86.58
Basic features	83.39%	86.90%	85.11

Table 6: Accuracy on the chunk detection subproblem (esp.testa). Common word features contribute very little.

ple binary classifiers. Here, when it was determined that a token should be labeled, we simply picked the class whose model produced the highest score.

Little change was seen using this new approach; recall was enhanced, but precision dropped slightly (see Table 7 below). This is the method that we then applied to the Dutch data set. Performance was very similar across the two languages; however, the MISC class was apparently much easier to learn in Dutch as can be seen in the tables on the right.

Spanish dev.	Precision	Recall	$F_{\beta=1}$
LOC	51.97%	67.11%	58.57
MISC	17.67%	23.15%	20.04
ORG	54.46%	64.94%	59.24
PER	65.73%	74.71%	69.93
Overall	52.75%	63.90%	57.80

Table 7: Results when classes were assigned subsequent to chunk detection

8 Conclusions

The approach we investigated relied on a minimum of language-specific support. Only some knowledge of the character encoding (e.g., ASCII features) and a list of common terms were utilized. We believe simple techniques continue to show promise, but that there is room for significant improvement. It may be possible to refine this approach and achieve higher performance while retaining a language-neutral focus. But it may be instead, that a large drop in accuracy will result from avoiding linguistic resources such as morphological analyzers, gazetteers, and POS-taggers.

We also found that the values of learning parameters are important; this agrees with results reported by Joachims on a text classification

Spanish dev.	Precision	Recall	$F_{\beta=1}$
LOC	51.97%	67.11%	58.57
MISC	17.67%	23.15%	20.04
ORG	54.46%	64.94%	59.24
PER	65.73%	74.71%	69.93
Overall	52.75%	63.90%	57.80

Spanish test	Precision	Recall	$F_{\beta=1}$
LOC	63.54%	63.19%	63.37
MISC	15.11%	16.18%	15.62
ORG	56.39%	72.79%	63.55
PER	63.53%	82.72%	71.87
Overall	56.28%	66.51%	60.97

Dutch devel.	Precision	Recall	$F_{\beta=1}$
LOC	62.86%	64.71%	63.77
MISC	58.55%	66.09%	62.09
ORG	55.26%	39.47%	46.05
PER	49.45%	75.99%	59.91
Overall	55.32%	61.59%	58.29

Dutch test	Precision	Recall	$F_{\beta=1}$
LOC	66.58%	66.67%	66.62
MISC	58.67%	63.27%	60.88
ORG	54.72%	44.24%	48.93
PER	50.18%	75.91%	60.42
Overall	56.22%	63.24%	59.52

Table 8: Our results for the four CoNLL-2002 Shared Task test sets.

problem (Joachims, 2000).

References

S. Cucerzan and D. Yarowsky. 1999. In *Joint SIGDAT Conference on EMNLP and VLC.*

T. Joachims. 2000. *Estimating the generalization performance of a SVM efficiently.*

T. Joachims. 2002. http://svmlight.joachims.org/.

T. Kudo and Y. Matsumoto. 2001. Chunking with support vector machines. In *NAACL-2001.*

A. McCallum, D. Freitag, and F. Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *ICML-2000.*

P. McNamee, J. Mayfield, and C. Piatko. 2001. A language-independent approach to european text retrieval. In *CLEF-2000 Workshop.*

K. Seymore, A. McCallum, and R. Rosenfeld. 1999. Learning hidden markov model structure for information extraction. In *Proceedings of the AAAI-99 Workshop on ML for IE.*