Improving a State-of-the-art Named Entity Recognition System Using the World Wide Web

György Szarvas[1], Richárd Farkas[1,2], Róbert Ormándi[2]

[1] University of Szeged, Department of Informatics
6721 Szeged, Hungary
[2] Research Group on Artificial Intelligence
of the Hungarian Academy of Sciences and University of Szeged
6721 Szeged, Hungary
{szarvas, rfarkas@inf.u-szeged.hu}
{ormandi.robert@stud.u-szeged.hu}

Abstract. The development of highly accurate Named Entity Recognition (NER) systems can be beneficial to a wide range of Human Language Technology applications. In this paper we introduce three heuristics that exploit a variety of knowledge sources (the World Wide Web, Wikipedia and WordNet) and are capable of improving further a state-of-the-art multilingual and domain independent NER system. Moreover we describe our investigations on entity recognition in simulated speech-to-text output. Our web-based heuristics attained a slight improvement over the best results published on a standard NER task, and proved to be particularly effective in the speech-to-text scenario.

Keywords: World Wide Web, web based techniques, named entity recognition, machine learning

1 Introduction

The identification and classification of Named Entities (NE) in plain text is of key importance in numerous natural language processing applications. In Information Extraction systems NEs generally carry important information about the text itself, and thus are targets for extraction. In machine translation, Named Entities and other sorts of words have to be handled in a different way due to the specific translation rules that apply to them.

We applied the NE Recognition and Classification (NER) system described in [10] which was designed for English language, and also worked with minor changes for Hungarian and domains different from newswire texts (medical records) [11]. To our best knowledge, this system gives the best results on the standard CoNLL-2003 task.

In this paper we investigate three heuristics that utilize online information (the World Wide Web and the Wikipedia online encyclopedia) to improve the performance of this state-of-the-art Named Entity Classification system.

As we plan to integrate our entity recognizer and classifier module into a multi-modal Information Extraction system, we tested the NER system in an artificial

scenario simulating speech-to-text output. As regards the problem of NER on the output of a general purpose speech-to-text application, it assumes that neither capitalization nor punctuation marks are available in the text. These restrictions make entity recognition a more challenging task. Experiments showed that the NER problem can be handled in such circumstances, without a serious loss of classification performance, while our web-based heuristics are particularly useful here.

In this paper we performed experiments for the English newswire NER task only but our heuristics should be portable across languages as long as the appropriate knowledge sources are available for the target language, with sufficient coverage[1].

1.1 Related work

The NER task was introduced during the nineties as a part of the shared tasks in the Message Understanding Conferences (MUC) [4]. The goal of these conferences was the recognition of proper nouns (*person*, *organization*, *location* names), and other phrases denoting dates, time intervals, and measures in texts from English newspaper articles. The best systems [1] following the MUC task definition achieved outstanding accuracies (nearly 95% F measure).

Later, as a part of the Computational Natural Language Learning (CoNLL) conferences [12], a shared task dealt with the development of systems like this that work for multiple languages and were able to correctly identify *person*, *organization* and *location* names, along with other proper nouns treated as *miscellaneous* entities.

There are some important differences between the CoNLL style task definition and the MUC approach that made NER a much harder problem. The most important is that CoNNL considers only whole phrases classified correctly (which is more suitable for real world applications). The F measure of the best performing systems [7] dropped below 89% for English.

There are several papers in the literature that investigate the usability of online resources for various NE-related tasks. The available systems seek to collect lists of Named Entities belonging to pre-specified classes from the WWW [5][6] or use online information for Named Entity Disambiguation [2], which differ from the problem addressed in this paper. We found no articles on using Web-searches to improve a NER system.

1.2 Structure of the Paper

In the following section we will introduce the NER problem in general, along with the details of the CONLL-2003 English task and the evaluation methodology. We also discuss the learning methods and other main characteristics of the NER system we applied. In section 3 we describe our web-based heuristics designed to improve the classification performance of a state-of-the-art NER system, followed by the description of our experiments on artificial speech-to-text data (Section 4).

[1] German, the language with the second largest Wiki encyclopedia has one third entries compared to English.

Experimental results are summarized in the last section along with some concluding remarks.

2 Description of the NER system applied

In this section we introduce the domain- and language independent NER system we used for our experiments. An NER system in English was trained and tested on a sub-corpus of the Reuters Corpus[2] (the CoNLL 2003 shared task database), consisting of newswire articles from 1996 provided by Reuters Inc. The data is available free of charge for research purposes and contains texts from diverse domains ranging from sports news to politics and the economy. The best result published in the CoNLL 2003 conference was an F measure of 88.76% obtained from the best individual model [7].

2.1 Evaluation Methodology

To make our results easier to compare with those given in the literature, we employed the same evaluation script that was used during the CoNLL conference shared tasks for entity recognition. This script calculates Precision, Recall and $F_{\beta=1}$[3] value scores by analyzing the text at the phrase level. This way evaluation is very strict as it can penalize single mistakes in longer entity phrases doubly.

It is worth mentioning that this kind of evaluation places a burden on the learning algorithms as they usually optimize their models based on a different accuracy measure. Fitting this evaluation into the learning phase is not straightforward because of some undesired properties of the formula that can adversely affect the optimization process.

2.2 Complex NER Model

The NER system we use here treats the NER problem as the classification of separate tokens. Following Szarvas et al. [10], we apply decision tree classifiers (with boosting). This way our model is fast to train and evaluate, and incorporates a very rich feature set (described in detail in [10]). The model also takes into account the relationship between consecutive words as well through a window with appropriate window size. The rich feature set enables to split the set, build models on each subset and then recombine their results. Figure 1 sketches the structure of the complex model.

[2] http://www.reuters.com/researchandstandards/

[3] In this paper we always mean $F_{\beta=1}$ under F measure.

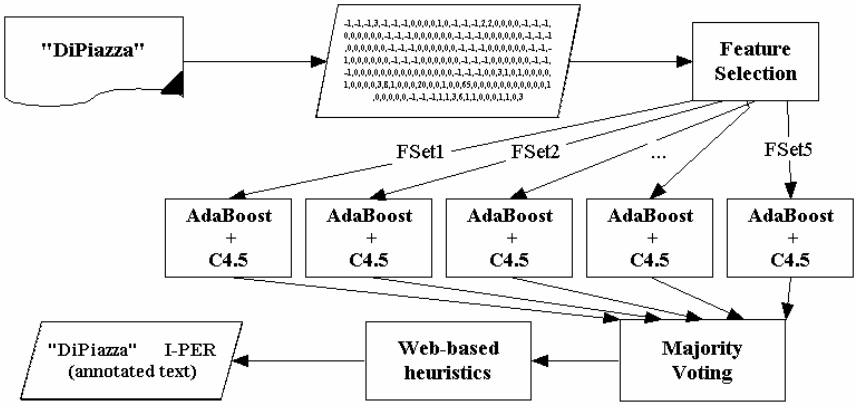

Figure 1.: The structure of our NER system

2.3 Feature Set

Initial features. We employed a very rich feature set for our word-level classification model, describing the characteristics of the word itself along with its actual context (a moving window of size four). Our features fell into the following major categories:

- gazetteers of *unambiguous NEs* from the train data: we used the NE phrases which occur more than five in the train texts and got the same label more than 90 percent of the cases,
- *dictionaries* of first names, company types, sport teams, denominators of locations (mountains, city) and so on: we collected 12 English specific lists from the Internet,
- *orthographical features*: capitalization, word length, common bit information about the word form (contains a digit or not, has uppercase character inside the word, regular expressions and so on). We collected the most characteristic character level bi/trigrams from the train texts assigned to each NE class,
- *frequency information*: frequency of the token, the ratio of the token's capitalized and lowercase occurrences, the ratio of capitalized and sentence beginning frequencies of the token,
- *phrasal information*: chunk codes and forecasted class of few preceding words (we used online evaluation),
- *contextual information*: POS codes, sentence position, document zone (title or body), topic code, trigger words (the most frequent and unambiguous tokens in a window around the NEs) from the train text, is the word between quotes and so on.

In our experiments we used a similar feature set splitting strategy as described in [10] to obtain 5 different (but not necessarily disjunctive) sets of features from the categories described above. We used these five sets for bagging similar classifiers to

obtain better results than in case of using all features together. The 5 similar AdaBoost+C4.5 boxes and Majority Voting illustrates this in Figure 1.

2.3 Classifiers and combination strategies

Boosting [9] and C4.5 [8] are well known algorithms for those who are acquainted with pattern recognition. Boosting has been applied successfully to improve the performance of decision trees in several NLP tasks. A system that made use of AdaBoost and fixed depth decision trees [3] came first on the CoNLL-2002 conference shared task for Dutch and Spanish, but gave somewhat worse results for English and German (it was ranked fifth, and had an F measure of 85.0% for English) in 2003.

As the results of [10] show, the combination of AdaBoost and C4.5 can bring some improvement in classification accuracy and preserves the superiority of decision tree learning in term of CPU time used for training and evaluating a model. In our experiments we used the implementations available in the WEKA [13] library, an open-source data mining software written in Java.

Combination of classifiers. There are several well known meta-learning algorithms in the literature that can lead to a 'better' model (in terms of classification accuracy) than those serving as a basis for it, or can significantly decrease the CPU time of the learning phase without loss of accuracy. The decision function used to integrate the five hypotheses (learnt on different subsets of features) was the following: *if any three of the five learners' outputs coincided we accepted it as a joint prediction, with a forecasted 'O' label referring to a non-named entity class otherwise*. This cautious voting scheme is beneficial to system performance as a high rate of disagreement often means a poor prediction rate. For a CoNLL type evaluation it is better to make such mistakes that classifies an NE as non-named entity than to place an NE in a wrong entity class (the latter detrimentally affects precision and recall, while the former only affects the recall of the system).

Here we used the same voting strategy for the baseline system, and tested other alternative voting schemes that exploit online information to assign NE labels in case of disagreement of the learnt models. This will be discussed in detail in the next section.

3 WWW based improvement of the NER system

Using online knowledge sources in Human Language Technology (HLT) and Data Mining problems has been an emerging field of research in the past few years. This trend is boosted by several special and interesting characteristics of the World Wide Web. First of all, it provides a practically limitless source of (unlabeled) data to exploit, and, more important it can bring some dynamism to applications. As online data changes and rapidly expands with time, a system can remain up-to-date and extending its knowledge without the need of fine tuning, or any human intervention (like retraining on up-to-date data for example). These features make the Web a very useful source of knowledge for HLT applications as well. On the other hand, the

usage of WWW is a new challenge to overcome for language processing applications as data cannot be accessed directly (only via a search engine) and might prove to be time consuming as task-specific pre-processing and collection of data is not feasible.

3.1 Fine-tuning phrase boundaries

A significant part of system errors in NER taggers is caused by the erroneous identification of the beginning (or end) of a longer phrase. Token-level classifiers (like the one we applied here) are especially prone to this as they classify each token of a phrase separately.

We considered a tagged entity as a candidate long-phrase NE if it was followed or preceded by a non-tagged uppercase word, or one/two stop words and an uppercase word. The underlying hypothesis of this heuristic is that if the boundaries were marked correctly and the surrounding words are not part of the entity, then the number of web-search results for the longer query should be significantly lower (the NE is followed by the particular word in just certain contexts). But in the case of a dislocated phrase boundary, the number of search results for the extended form must be comparable to the results for the shorter phrase (over 0.1%[4] of it). This means that every time when we found a tagged phrase that received more than 0.1% web query hits in an extended form, we extended the phrase with its neighboring word (or words). This decision function was fine tuned and found to be optimal on the training and development sets of the CoNLL task; the following evaluations have been performed on the CoNLL evaluation set.

This web-based post-processing heuristic improved the performance of the applied NER model from 89.02% to 89.15% F measure. The relatively small improvement is due to the classification error of some extended phrases (this heuristic extended the phrase boundaries precisely in several cases where the class label was assigned incorrectly by the classifier, and those left the system performance unchanged).

3.2 Using the most frequent role in uncertain cases

Some examples are easier to classify for a given model than others. In our applied NER system, the final decision was obtained by applying the majority voting procedure of 5 classifiers (which were all trained on different sets of features). A simple way of interpreting the uncertainty of a decision is to measure the level of disagreement among the individual models. We considered a token as a difficult or uncertain example if no more than 2 models gave coinciding decisions (we should mention here that each models chose the most probable of 5 different possible answers, so this indeed meant a high level of uncertainty).

Our hypothesis here was that the most frequent role of a named entity can be statistically useful information. Thus we did the following: if the system was unable to decide the class label of a phrase (it could not find evidence in the context of the

[4] We sought to keep the evaluation set blind. All the heuristics were fine-tuned on CoNLL-2003 developement set.

certain phrase) then we mined the most frequent usage of the corresponding NE using the WWW and took that as prediction.

The most frequent role searching method we applied here was inspired by the category extraction methods of Etzioni et al. [6]. This approach works by invoking several special Google queries in order to find such noun phrases following or preceding the pattern that is a category name for a particular class. The following queries were used to obtain category names from web search results:

NP such as NE
NP including NE
NP especially NE
NE is a NP
NE is the NP
NE and other NP
NE or other NP

Category names from the training data. We used the lists of unambiguous NEs collected from the training data to acquire common NE category names. We sent Google queries for NEs in the training data and all the patterns shown above. The heads of the corresponding NPs were extracted from the snippets of the best ten Google responses.

We found 173 reliable category names by performing a limited number of Google queries. Using these category lists as a disambiguator (we assigned the class sharing the most words in common with those extracted for the given NE) when the NER system was unable to give a reliable prediction was beneficial to overall system performance. The system F Measure improved from 89.15% to 89.28%. We should mention here that the baseline NER system labeled these examples as non-entities, whose prediction was incorrect in the majority of the cases.

Enriching category lists using WordNet. We enlisted the help of a linguist expert to determine the WordNet synset corresponding to each category name we found and give its most common substituting synset (the one highest in hypo/hypernym hierarchy) that was still usable as a category name for the particular NE class. Using these WordNet synsets we extended our category lists (to a size of 19537) with all literals that appear in their hyponym subtree (with sense #1). This additional knowledge further improved the F measure of the NER system to 89.35%.

4 Experiments on speech-to-text data

Named Entity Recognition on the output of a speech-to-text system has to handle the problem of several missing features (like capitalization) that are particularly useful for entity recognition.

We used the same data as for the experiments described above, but modified the text so it looked as if it had been obtained from a speech-to-text system. First we converted all tokens to lowercase, thus the feature that is undoubtedly the most important for NER became unavailable. Second, we removed all punctuation marks

from the original corpus (they do not appear explicitly in the audio stream, only in the accent hence it is doubtful that any punctuation can be retrieved efficiently). This means we assumed that all word forms were recognized correctly.

In the majority of cases, consecutive Named Entities either follow each other with a separating punctuation mark (enumerations), or belong to different classes. In the first case, a non-labeled token separates the two phrases, while in the second case the different class labels identify the boundaries. Rarely do two or more NEs of the same type appear consecutively in a sentence. In such cases the phrasal boundaries must be marked with a tag ('B-' instead of the common 'I-' prefix). We changed 'I-' tags to 'B-' where it was necessary in the simulated speech-to-text data to retain the correct phrase boundaries. This conversion resulted in over ten times more consecutive NEs (those separated with 'B-' tag), and hence the separation of such phrases became no longer negligible.[5]

We should add here that this simulation of the output of a speech-to-text system seemed obvious for two reasons. First, we wanted to test how a NER system behaves in significantly different circumstances, not a speech-to-text system itself. Second, by doing this we could avoid the need for a NE-labeled real speech database and also have better grounds for comparison between written text and speech-to-text output as we used a standard database. The performance of the baseline NER system on this converted text decreased to 81.1% $F_{\beta=1}$. Even though this simplification does not take into account that real speech-to-text data would certainly contain word errors, it fits to our purposes well (it is capable of demonstrating the usability of online knowledge sources to improve NER in speech-to-text data).

4.1 Identifying consecutive NEs

As we stated above 'B-' tags are even more common in texts obtained from a speech-to-text system due to the absence of punctuation marks. We exploited the encyclopedic knowledge of Wikipedia to enable our system to distinguish between long phrases and consecutive entities.

The B-tag heuristic. We queried the Wikipedia site for all entities that had two or more tokens. If we found an article sharing the same title as the whole query, or the majority of the occurrences of the phrase in the Google snippets occurred without punctuation marks inside, we treated the query phrase as a single entity. If a punctuation mark was inside the phrase in the majority of the cases, we separated the phrase at the position of the punctuation mark. This method allowed us to separate phrases like 'Golan Heights | Israel'. If there was no hit for the query in the Wikipedia, but we were able to find a specific article for two or more parts of the query, we put phrase boundaries following the Wiki entries. This way we identified successfully phrases like 'Taleban | MiG-19' and many enumerations that lacked the separating commas due to the removal of punctuation marks from the data. We made use of a first names list here containing 3217 first names which allowed us to avoid

[5] Most of the best performing NER systems deliberately ignore the separation of consecutive phrases as they are too sparse to handle efficiently in written text data. This problem has no significant effect on performance either (there are only 20 'B-' tokens out of 50,000 in the CoNLL-2003 test dataset).

the erroneous separation of full names (First name, Last name pairs). Of course a more comprehensive first names list would be beneficial. Our system suffered from the lack of Romanian or Arabic First names. This heuristic improved the overall performance of the NER tagger on speech-to-text data by a significant 1.42% (8,1% error reduction). The heuristic itself managed to recognize the 'B-' tags with an $F_{\beta=1}$-measure of 75.19% (precision 71.7%; recall 79.03%).

We should also mention here that some of the 'B-' phrases in the CoNLL database are arguably consecutive NEs, but are actually single entities (e.g. 'English Moslems' or 'City State' phrases like 'Rochester NY'). Our heuristic does not divide up such cases as they usually seem to be single NEs for the online encyclopedia – and they can be treated as single entities as well in an Information Extraction system. Without these cases the recall of our system would have been even higher.

5 Summary of the results

A brief summary of the heuristic improvements achieved on the various systems can be seen in Table 1. Here we show the system described in Section 2 (and in [10] in more details), ***Base NER***; its voting with the 2 best CoNLL systems, ***Voting***; the system described in Section 2 on the speech-to-text data, ***Speech-to-text***. The boundary heuristic is not applicable in the speech-to-text task, because it is dependant on the capitalization of the context. The ***Voting*** column adds a further voting level to the system (not showed in Figure 1.), it is obtained by the majority voting the best performing CoNLL systems and ***Base NER***. This hybrid method was also discussed in [10]; we show here that the improvement of our web based heuristics carries over to this hybrid model also.

Table 1.: Results of the three heuristics, $F_{\beta=1}$

	Base NER	Voting	Speech-to-text
Baseline	89.02%	91.40%	81.10%
B-tag	89.02%	91.40%	82.52%
Boundary id.	89.15%	91.51%	n/a
Most freq. role	89.35%	91.67%	82.64%

6 Conclusion

The aim of this paper was to show the potentials of the WWW in HLT problems like named entity recognition. Our heuristics are based on the assumption that, even though the World Wide Web contains much useless and incorrect information, regarding our simple features the correct usage of language dominates over misspellings and other sorts of noise. Our experiments confirmed this hypothesis. We showed experimentally that these heuristics could further improve a state-of-the-art

NER system on the standard text processing task and they proved to be particularly useful on a more challenging simulated speech-to-text task. We believe that our results are valuable due to two main reasons: first, we managed to give improvements on a top performing model for the task of NER that is of great importance even if the improvement is slight. Second, we showed that the WWW can be exploited with significant success to overcome the drawback caused by the lack of certain information that is extremely important and characteristic for certain HLT applications (like the absence of punctuation or capitalization is NER).

Acknowledgments. We would like to thank the anonymous reviewers for their valuable comments.

References

1. Bikel, D. M., Schwartz, R. L. and Weischedel, R. M. *An algorithm that learns what's in a name.* Machine Learning, 34-1-3, 211-231. (1999)
2. Bunescu, R. and Paşca, M. *Using Encyclopedic Knowledge for Named Entity Disambiguation.* In Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (2006)
3. Carreras X., Márques L. and Padró L., 2002. Named Entity Extraction using AdaBoost Proceedings of CoNLL-2002, Taipei, Taiwan, pp. 167-170.
4. Chinchor, N. *MUC-7 Named Entity Task Definition*, Proceedings of Seventh MUC. (1998)
5. Cimiano, P., Handschuh, S. and Staab, S. *Towards the self-annotating web.* In Proceedings of the 13th WWW Conference (2004)
6. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S. and Yates, A. *Unsupervised named-entity extraction from the web: an experimental study.* Artificial Intelligence Volume 165, Issue 1, 91-134. (2005)
7. Florian, R., Ittycheriah, A., Jing, H., and Zhang, T. *Named Entity Recognition through Classifier Combination.* Proceedings of CoNLL-2003 (2003)
8. Quinlan, R. *C4.5: Programs for machine learning.* Morgan Kaufmann (1993)
9. Shapire, R. E. *The Strength of Weak Learnability*. Machine Learnings, Vol. 5, 197-227 (1990)
10. Szarvas, Gy., Farkas, R. and Kocsor, A. *A multilingual named entity recognition system using boosting and c4.5 decision tree learning algorithms.* Lecture Notes on Artificial Intelligence Vol. 4265, 267-278. (2006)
11. Szarvas, Gy., Farkas, R., Iván, Sz., Kocsor, A. and Busa-Fekete, R. *An iterative method for the de-identification of structured medical text.* Workshop on Challenges in Natural Language Processing for Clinical Data (2006)
12. Tjong Kim Sang, E. F., and De Meulder, F. *Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition*. Proceedings of CoNLL-2003. (2003)
13. Witten I. H. and Frank E. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, second edition. (2005)