View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by University of Szeged

Acta Cybernetica **18** (2007) 293–301.

Hungarian named entity recognition with a maximum entropy approach

Dániel Varga* and Eszter Simon†

Abstract

In the analysis of natural language text a key step is *named entity recognition,* finding all complex noun phrases that denote persons, organizations, locations, and other entities designated by a name. In this paper we introduce the `hunner` open source language-independent named entity recognition system, and present results for Hungarian. When the input to `hunner` is already morphologically analyzed, we apply the system together with the `hunpos` morphological disambiguator, but `hunner` is also capable of working on raw (morphologically unanalyzed) text.

Keywords: natural language processing, computational linguistics, named entity recognition

1 Introduction

In the machine analysis of natural language documents we often seek to answer questions in terms similar to those used by humans: *who* is this document about, *where* is the action taking place, *how much money* is involved, and so on. By *named entity recognition* (NER) we mean an algorithm that takes natural language text (typically, in document-sized chunks rather than word by word or sentence by sentence) as input, and identifies all persons, locations, organizations and similar entities that are designated by a name. The name can be a single word (proper noun) such as *Budapest* or a complex phrase such as *Budapesti Műszaki és Gazdaságtudományi Egyetem Média Oktató és Kutató Központ.* Even when the eventual goal is more remote (e.g. machine translation, information extraction, or information retrieval), NER is a useful intermediate stage of processing.

The trivial algorithm that identifies those phrases as named entities which are written capitalized works well for English, Hungarian, and many other languages that capitalize proper names, but of course it fails both for languages like German

*Budapesti Műszaki és Gazdaságtudományi Egyetem – Média Oktató és Kutató Központ, E-mail: `daniel@mokk.bme.hu`

†Budapesti Műszaki és Gazdaságtudományi Egyetem – Kognitív Tudományi Tanszék, E-mail: `esimon@cogsci.bme.hu`

where capitalization conventions are different, for languages like Arabic or Chinese that do not have a separate upper- and lowercase, and for text that lacks casing (e.g. the output of speech recognition). Also, because the trivial algorithm is prone to false positives sentence-initially and to false negatives in text written in anything but the most carefully edited prose, it is important to develop methods that are less error prone.

The NER task, as posed by the MUC and CoNLL competitions [8], is to identify disjoint chunks of the input token sequence as named entities and to annotate these chunks with a small set on named entity categories such as PERson, ORGanization, LOCation, and MISC (all other named entities). The evaluation of an automatic NER algorithm is through comparing its output to a manual annotation. Typically, the algorithm itself learns its parameters from a manually annotated corpus (supervised learning).

For major languages, many dozens of papers were published on NER algorithms. Almost every currently known supervised machine learning technique was used. Some examples: BBN IdentiFinder [7], and Zhou and Su [14] apply Hidden Markov Modeling. Borthwick [2] [1], and Chieu and Ng [3] apply the maximum entropy approach. Sekine et al. [9] use decision trees. There are not too many language-dependent components of these, and other similar systems. Still, for Hungarian, we are only aware of one quantitative study of a NER system which is based on machine learning methods: Szarvas et al. [11] published results on their NER system based on C4.5 decision trees with Boosting. Their system achieved state-of-the-art accuracy for English, and for Hungarian it reached an accuracy of 94.77% CoNLL F-score on the Szeged NER Corpus [10]. (See Subsection 5.1. for the definition of CoNLL F-score.)

In Section 2 we describe the corpus, and in the Section 3 the external resources we used. We detail the architecture and implementation of our system in Section 4, and describe the methodology and the results of our experiments in the concluding Section 5.

2 The corpus

For the training and evaluation of our system, we used the Szeged NER Corpus [10]. At the time of writing, this is the only named entity annotated Hungarian corpus with a size suitable for supervised learning.

The Szeged NER Corpus is a more than 220 thousand token subset of the Szeged Corpus [4], manually annotated for named entities. A distinct characteristics of the Szeged NER text is its thematic homogenity: it only contains various subgenres of business news. This means that organization names are very highly represented. This category dominates the others in freqency.

The annotation of the corpus follows the tagset and annotation conventions of CoNLL [8]. This means that we used the following tagset: person names (`PER`), organization names (`ORG`), location names (`LOC`) and miscellaneous other named entities (`MISC`). In the Szeged NER Corpus, the `MISC` category mostly contains

brand names and financial acronyms.

An example sentence from the corpus: „[Sam DiPiazza]$_{PER}$, a [PWC]$_{ORG}$ vezérigazgatója szerint az [EU]$_{ORG}$ által már kötelezővé tett úgynevezett [GAS]$_{MISC}$ az eddiginél nagyobb betekintést ad az adott cég pénzügyeibe." (According to [Sam DiPiazza]$_{PER}$, chief executive of [PWC]$_{ORG}$, the so-called [GAS]$_{MISC}$, already enforced by the [EU]$_{ORG}$, gives more insights into the finances of the given firm.)

3 Gazetteers

Though 'gazetteer' originally means geographical directory, in the context of the NER task the phrase is simply used as a list of names. We assembled various gazetteers to be incorporated into our system.

- Hungarian and common non-Hungarian last names
- Hungarian and common non-Hungarian first names
- names of Hungarian cities
- country names in Hungarian
- Hungarian street names
- Hungarian organization names
- international organization names
- suffices for company names
- suffices for street names
- financial acronyms

In the first six cases, our source was an aggregated version of a Hungarian phone book, and a web database. The lists of Hungarian organization names and street names were cleaned of suffices with automatic methods. The common suffices (e.g. Inc., Ltd. for organizations, Street, Sq. for places, or in Hungarian 'Kft.', 'Rt.', 'utca', 'tér', respectively) were extracted, and moved into separate lists. The international organization list was kindly provided to us by György Szarvas and Richárd Farkas.

There was just one case when analyzing the development corpus lead to the inclusion of a new dictionary: the lexicon of financial acronyms. The development corpus contained several stock market index names (DAX, Libor, Nasdaq), which were sometimes marked as `ORG` instead of `MISC` by the algorithm. To solve this problem, we extracted such stock market terms from a web-based financial knowledge base. We note that using this lexicon did not improve the performance on the test corpus, and even decreased it slightly. The reason for this is that most of

these terms occured in the development part of the corpus, i.e. the split between the test and the development corpus was not a random one (see Subsection 5.1 for more details).

The gazetteers incorporated into our final system (except the case of financial terms) were finalized before the inspection of the train and development corpora. During the tuning of the system to the development corpus, we have found serious cases of over- and undergeneration in the gazetteers. Since correcting these errors did not improve accuracy significantly, we reverted to the original, uncorrected, automatically collected versions of the gazetteers especially as using these results in a cleaner methodology (less manual labor).

Similarly to the source code of the system, we publish the gazetteers under a free document license.

4 Architecture

Our system roughly follows the architecture described by [1] and [3], incorporating some ideas introduced by [6].

When building a supervised machine learning system, a major step is feature extraction, that is, collecting information from the raw data that can be relevant for the classification task. In the case of the NER task, an obvious approach to feature extraction is to collect such information from the neighbourhood of the inspected token. Some possible examples of features are capitalization, part of speech, occurrence in some dictionary. The task of the supervised machine learning algorithm is then to find in this large amount of information regularities that are relevant to the classification task.

We used maximum entropy learning as our supervised machine learning approach. During classification, the output of the maximum entropy algorithm was post-processed in a so-called smoothing phase we will describe in Subsection 4.4.

4.1 Feature extraction

Most of our features deal with very easily computable syntactic properties of tokens. On the other hand, we exploited the fact that we have the `hunpos` morphological disambiguator [5] at our disposal.

The feature set was composed manually. Below is the complete list of the features used by our system:

1. Is some neighbourhood of the token contained in a gazetteer? If yes, is the token at beginning, ending or middle position of the phrase? (To deal with morphology, when determining the matching of multi-word phrases, we treated the last word of the phrase differently: matching on a suitably chosen prefix was enough. This corresponds to the way Hungarian multi-word phrases are inflected.)

2. Sentence start, sentence end position.

3. Boolean valued syntactic properties of the word form: upper case, all upper case, contains capitalized letter after noncapitalized (e.g. *iPod*), is a number, contains a number, contains a dash, contains a period.

4. String-valued surface properties of the word form: the word form itself, the five-letter prefix, and most notably all consecutive three-letter character sequences (trigrams) of the word form. Note that we use gazetteers crafted beforehand, but in practice, these features have an effect similar to gazetteers directly extracted from the train corpus. Similar application of character n-grams for named entity recognition was first proposed by Klein et al. [6].

5. Information provided by the `hunpos` morphological disambiguator: part of speech (NOUN, ART, NUM, ADJ, VERB, etc.). The lemma of the token. Is the word form recognized by `hunpos`? Is the identified lemma differently capitalized than the token itself?

Example: The *Gyula* token, in a sentence starting position gets the following features: Built-in Boolean features: `sentencestart caps`. Character n-gram features: `tri.Gyu tri.yul tri.ula prefix.Gyula`. Gazetteer features: `firstname.lone city.lone familyname.lone corp.start`. (`.lone` here means that the gazetteer contains the token itself, as opposed to the case of `corp.start`, which means that the `corp` gazetteer contains a phrase starting with this token.) Morphological features: `postag.noun lemma.Gyula`.

The system collects these data for each individual token of a sentence. To incorporate context, we simply add the features of neighbouring tokens, recording the relative positions of the tokens. For example, if a token gets the feature `caps.pre2`, it means that the token two positions before is capitalized. A parameter of our system is the size of the context window for a given feature. For simplicity's sake, we didn't optimize this parameter for each feature separately. According to our experiments, in the case of character trigrams and prefixes, using context radius 3 (that is, a 7-token interval) leads to optimal results. In the case of the rest of the features, context radius 5 (11 tokens) was used.

4.2 Tag sets

The NER task in its original form deals with the classification of unknown contiguous token sequences, and it is not immediately obvious how to phrase this as a token classification task. Roughly following [3], we chose the following solution: Every token must be classified into one of 17 different classes: { `O, LOC.single, LOC.start, LOC.middle, LOC.end, ORG.single, ..., MISC.end` }. There are two major advantages of this approach: First, the machine learner can more easily recognize correlations that are specific to the start or end of the NE. Second, the tag set has implicit built-in consistency requirements: e.g. `*.start` can not follow `*.middle`. As we will see in Subsection 4.4, we can use this fact to improve the output of the machine learner in a post-processing step.

4.3 Maximum entropy method

We used the maximum entropy method as our classification methodology. This was already successfully used in the weighted morphological analyzer component of the `hunpos` morphological disambiguator. We also experimented with C4.5 decision trees and support vector machines, but the maximum entropy method gave consistently superior results.

We have chosen Zhang Le's [13] maximum entropy implementation because of its high performance. This implementation uses the L-BFGS algorithm [15] for model building.

On our data sets, the L-BFGS iterative learning algorithm starts to converge after approximately 100 iterations. The accuracy of the model fluctuates wildly and randomly before this iteration number is reached. Our published numbers are based on 300 iterations. On the other hand, because of the relatively high (approx. 1 hour) running time of 300 iterations, some of our elementary feature engineering decisions were based on measurements with lower (30 or 100) iterations. This may have led to suboptimal decisions.

We note that using prefixes, character trigrams and very wide context windows led to a very high total number of features. On the 200,000 token corpus, 250,000 different kinds of features occur, for a total of 10 million feature instances. The maximum entropy approach is capable of dealing with such a high number of features without the feature selection phase needed by some other machine learning methods.

4.4 Smoothing

One important characteristics of maximum entropy learning is that during classification it can emit a full probability distribution on tags, instead of just a single tag with maximum likelihood. The advantage of this is that we can override local decisions if they prove to be inconsistent with each other. The method described below is common in the machine learning literature [3]. First we query the maximum entopy algorithm for tag emission probabilities for each token. We then define transition probabilities between tags as follows: Transition probabilities for illegal transitions (e.g. `ORG.start` after `LOC.start`) are set to be zero. Every legal transition (e.g. `ORG.start` after `LOC.end`) is set to be equiprobable. Treating the maximum entropy algorithm's outputs as independent distributions, we can apply the Viterbi algorithm to calculate the tag sequence that maximizes the joint transition-emission probability for a whole sentence. This tag sequence will be necessarily well-formed. According to our measurements, this parameterless post-processing step improves the system's F-score by approximately 0.5% in typical measurement setups.

5 Measurements

5.1 Methodology

To measure the accuracy of a machine learning algorithm, its output has to be compared to a gold standard test dataset. The standard method to quantify the similarity between two named entity labelings is the CoNLL F-score. According to this, a gold standard named entity is correctly labeled if the automatic labeling gives the same start- and end-position, and the same named entity class. Based on this, precision and recall values can be calculated for the corpus, and the F-score is, as usual, the harmonic mean of these two values.

We started the early development of our system with an ad hoc train-test split of the Szeged NER Corpus. But it quickly became apparent that if we intend our results to be comparable to the only existing quantitative study on Hungarian named entity recognition, then we will have to switch to the train-development-test split used by [11]. Szarvas et al. were kind to provide this split, and from this point, we followed standard methodology: We optimized the parameters of the system guided by the F-score on the development corpus, and only measured the F-score on the test corpus once, when this optimization was finished.

5.2 Results

The system described above reached an F-score of 96.35 on the development corpus, and 95.06 on the test corpus. This is a minor improvement on the numbers published by [11] (see Table 1). But we have to note that the [11] system was optimized in parellel for English and Hungarian. Our system obviously needs further work to give state-of-the-art results for several languages.

Table 1: Results

NE-type	**devel**	**test**	**Szarvas et al devel**	**Szarvas et al test**
`LOC`	92.06	96.36		95.07
`MISC`	93.58	85.12		85.96
`ORG`	97.62	96.20		95.84
`PER`	97.44	94.94		94.67
Global	**96.35**	**95.06**	**96.20**	**94.77**

We measured the effect of each major subsets of the features. As we noted, the global accuracy score of the system was 95.06 on the test corpus. Removing just the built-in Boolean features (`caps`, `dash`, etc.) decreased this score to 92.37. Removing just the character n-gram features decreased the score to 90.04. The gazetteer and morphological features had significantly less effect: removing these decreased the score to 94.69 and 94.70, respectively. Note that these two sets of

features were exactly the ones that required external resources. Removing both led to a resourceless system without seriously affecting the score: the resourceless system had an accuracy of 94.73.

6 Acknowledgements

We thank György Szarvas and Richárd Farkas for sharing their data sets and Péter Halácsy for help in the implementation of the algorithm.

References

[1] Borthwick, A.. *A Maximum Entropy Approach to Named Entity Recognition*. PhD thesis, New York University, 1999.

[2] Borthwick,A., Sterling, J., Agichtein, E., Grishman, R.. NYU: Description of the MENE Named Entity System as Used in MUC-7. In: *Proceedings of MUC-7*, 1998.

[3] Chieu, H. L., Ng, H. T.. Named Entity Recognition with a Maximum Entropy Approach. In: *Proceedings of CoNLL-2003*, pp. 160-163. Edmonton, Canada, 2003.

[4] Csendes, D., Csirik, J., Gyimóthy, T.. The Szeged Corpus: A POS tagged and Syntactically Annotated Hungarian Natural Language Corpus. in: *Proceedings of TSD 2004*, vol. 3206 (2004) 41-49.

[5] Halácsy, P., Kornai, A., Varga, D.. Morfológiai egyértelműsítés maximum entrópia módszerrel (Morphological disambiguation with maximum entropy method). In: *Proc. 3rd Hungarian Computational Linguistics Conf.*, 2005, Szegedi Tudományegyetem.

[6] Klein, D., Smarr, J., Nguyen, H., Manning, Ch. D.. Named Entity Recognition with Character-Level Models. In: *Proceedings the Seventh Conference on Natural Language Learning*, 2003.

[7] Miller, S., Crystal, M., Fox, H., Ramshaw, L., Schwartz, R., Stone, R., Weischedel, R., and the Annotation Group (BBN Technologies). BBN: Description of the SIFT System as Used for MUC-7. In: *Proceedings of MUC-7*, 1998.

[8] Sang, E. F. T. K., De Meulder, F.. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In: *Proceedings of CoNLL-2003*, 2003. http://www.cnts.ua.ac.be/conll2003/ner/.

[9] Sekine, S., Grishman, R., Shinou, H.. A decision tree method for finding and classifying names in japanese texts. In: *Proceedings of the Sixth Workshop on Very Large Corpora*, Montreal, Canada, 1998, 14(4):365-393, 1999.

[10] Szarvas, Gy., Farkas, R., Felföldi, L., Kocsor, A., Csirik, J.. A highly accurate Named Entity corpus for Hungarian. In: *Proceedings of International Conference on Language Resources and Evaluation*, 2006.

[11] Szarvas, Gy., Farkas, R., Kocsor, A.. A Multilingual Named Entity Recognition System Using Boosting and C4.5 Decision Tree Learning Algorithms. In: *Proceedings of Discovery Science 2006*, DS2006, LNAI 4265 pp. 267-278, 2006. Springer-Verlag, 2006.

[12] Trón, V., Gyepesi, Gy., Halácsy, P., Kornai, A., Németh, L., Varga, D.. Hunmorph: open source word analysis. In: *Proceeding of the ACL 2005 Workshop on Software*, 2005.

[13] Le, Zh.. Maximum Entropy Modeling Toolkit for Python and C++. http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

[14] Zhou, G. D., Su, J.. Named Entity Recognition using an HMM-based Chunk Tagger. In: *Proceedings of the 40th Annual Meeting of the ACL*, Philadelphia, pp. 473-480, July 2000.

[15] Zhu, C., Byrd, R. H., Lu, P., Nocedal, J.. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. In: *ACM Transactions on Mathematical Software (TOMS)*, 1997.