Identifying Named Entities in Text Databases from the Natural History Domain

Caroline Sporleder, Marieke van Erp, Tijn Porcelijn, Antal van den Bosch, Pim Arntzen*

ILK/Language and Information Science
Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands
{C.Sporleder,M.G.J.vanErp,M.Porcelijn,Antal.vdnBosch}@uvt.nl

* Naturalis, National Museum of Natural History
Darwinweg 2, 2333 CR Leiden, The Netherlands
Arntzen@naturalis.nl

Abstract

In this paper, we investigate whether it is possible to bootstrap a named entity tagger for textual databases by exploiting the database structure to automatically generate domain and database-specific gazetteer lists. We compare three tagging strategies: (i) using the extracted gazetteers in a look-up tagger, (ii) using the gazetteers to automatically extract training data to train a database-specific tagger, and (iii) using a generic named entity tagger. Our results suggest that automatically built gazetteers in combination with a look-up tagger lead to a relatively good performance and that generic taggers do not perform particularly well on this type of data.

1. Introduction

Named entity tagging refers to the task of identifying named entities (such as person names) in a text and assigning them to the appropriate category, e.g. PERSON, ORGANISATION or LOCATION. It is an important subtask for information extraction and retrieval. Named entity tagging is a well studied area, with many implemented systems obtaining performance levels of 80% or more average F-Score (e.g., Florian et al. (2003)). However, most previous research in this area has focused on data that is more or less unstructured, i.e., raw or typeset texts. Arguably, named entity recognition is equally important for information extraction from semi-structured sources, such as textual databases. Databases with a large amount of textual content are frequently used to store information about collections and archives. Examples are the University of St. Andrews Photographic Collection,[1] the CANMORE database of historical monuments in Scotland[2], and the Nederlands Soortenregister[3], but also Amazon's book database[4] or the Internet Movie Database.[5] These databases all have in common that some of their fields contain more or less "free text", for example the *title* field in the University of St. Andrews Photographic Collection contains multi-word text strings like:

(1) Cathedral ruins, West Gate, St Andrews. View of West Entrance and East Gable from north west

For such free-text columns, named entity tagging would be beneficial. However, generic named entity taggers may not work very well on this type of data. First, these taggers are typically trained and tested on texts from the news domain. Porting them to other domains may not provide good results, not only because there may be domain-specific differences in language use, but also because new entity classes may be needed for new domains (cf. Bontcheva et al. (2002)). For example, for archaeological data it would be useful to have categories like ARTEFACT or SITE, in addition to, or even instead of the entity categories normally defined for the news domain (typically PERSON, LOCATION, and ORGANISATION). Also, even if an entity category occurs in both domains, the linguistic cues for that category may be different. For example, an ORGANISATION in the news domain is frequently signalled by abbreviations such as *Ltd.* or *Corp.* (e.g., *Intel Corp.*), whereas organisations in the cultural heritage domain are more likely to be signalled by words such as *Museum* or *Trust* (e.g., *Perth and Kinross Heritage Trust*). Second, language use in databases differs from language use in "normal", unstructured text; for example, text fields in databases often contain incomplete sentences, as can be seen in example (1) above. The degree and type of incompleteness of sentences in cells of such databases is often triggered by the type of column the cell is in; for instance, a column header *location of archaeological site* may trigger locative phrases in cells of that column (*100 miles north of Mexico City*).

While porting a generic tagger may prove difficult, it might alternatively be possible to exploit the semi-structured nature of a database to bootstrap a database-specific entity tagger. Usually, the database structure provides implicit information about at least some entities. For example, the St. Andrews Photographic Collection database has a *location* field, which should contain entities of type LOCATION. Similarly the *originator* (i.e., photographer) field should contain entities of type PERSON. This can be exploited to automatically create gazetteer lists, which can then form the basis for a simple look-up tagger. Alternatively, the gazetteers can be used to automatically extract examples to train a database-specific tagger. Similar bootstrapping approaches to named entity recognition have been proposed before, though to our knowledge not for textual databases. This paper investigates how successful such an approach could be, and how the performance of such a bootstrapped tagger compares to that of a generic tagger.

[1] http://special.st-andrews.ac.uk/saspecial/
[2] http://www.rcahms.gov.uk/index.html
[3] http://www.nederlandsesoorten.nl
[4] http://www.amazon.com
[5] http://www.imdb.com/

2. Related Work

Over the past few years there has been a lot of research into named entity recognition strategies that do not require a large set of manually annotated training examples. A number of approaches exploit the redundancy between word internal and contextual cues (Collins and Singer, 1999; Cucerzan and Yarowsky, 1999). An example of a word internal cue is the fact that a capitalised word starting with the letters *Mc* is likely to belong to the PERSON class, as in *McLeod*. An example of a contextual cue is that a word which is followed by *Ltd.* is likely to be of type ORGANISATION. These two types of cues can be combined in a co-training loop (Blum and Mitchell, 1998). The basic algorithm starts with a small seed list, which is used to label examples in an unannotated corpus. These then form the basis for extracting contextual classification rules which are again applied to the corpus to label additional examples from which word internal rules can be generated, and so on. Phillips and Riloff (2002) also proposed a co-training approach but instead of relying on word internal and contextual cues they exploit the fact that common and proper nouns of the same semantic class tend to co-occur in certain syntactic constructions, such as appositives. For example, if the expression *John Seng, the financial analyst* is encountered and it is known that *John Seng* belongs to the PERSON class then it can be inferred that *financial analyst* also belongs to that class. This in turn allows one to infer that *Peter Smith* in *Peter Smith, the financial analyst* is also of type PERSON. Lin et al. (2003) present a bootstrapping approach in which multiple semantic classes are learnt simultaneously on the basis of positive and negative examples. Finally, Buchholz and van den Bosch (2000) collect sizable seed lists from the internet and use these to label training examples in a corpus, which are then used to train a named entity tagger, which can in turn be applied to unlabelled data to gather more names to add to the seed lists. Our approach is similar. However, instead of generating seed lists from the web, we exploit the database structure to create gazetteer lists.

3. Data

We tested our bootstrapping approach on a zoological database containing information about reptile and amphibian specimens, collected by researchers at the Dutch Natural History Museum Naturalis in Leiden. Each database record encodes information about one or more specimens, describing the circumstances of its collection, its position in the zoological taxonomy and so on. The database currently contains about 16,870 records and 35 columns. Some fields (like *special remarks*) contain more or less free text; others contain entities of a given type (e.g., *place* contains an expression of type LOCATION).

4. Bootstrapping a Named Entity Tagger

We used three different named entity classes: location (LOC), person (PER), and taxonomic (TAX). The gazetteers for the three classes were obtained from 14 entity specific fields. Table 1 shows the fields from which we extracted, together with the entity type and an example entry for each field. Direct extraction from these fields leads to some noise; for example, person fields often contain more than one entity or additional, non-entity material. For instance the *author* field, which encodes the author of the publication which forms the basis for a species classification, may contain the string *Cole & Dessauer, 1993*, where *1993* refers to the year in which the publication appeared. We implemented a few simple rules to split multiple expressions and remove additional material. We also used a small list of stop words to identify and remove expressions that are not of the required type. For example, the *donator* field sometimes contains entities that are organisations (e.g., *Museum of Vertebrate Zoology*); we identified these by looking for words like *museum* or *collection*. Finally, we generalised person names, e.g., if *M.S. Hoogmoed* was found, we also added *Hoogmoed, M.S.* and *Hoogmoed* to the list. Overall we extracted 1,009 expressions of type PER, 3,568 expressions of type LOC and 2,167 expressions of type TAX.

Field	Example	Type
place	Bigisanti beach	LOC
province/state	Marowijne	LOC
land	Indonesia	LOC
author	Daudin, 1802	PER
collector	Hoogmoed, M.S.	PER
donator	P.J.M. Maas	PER
determinator	M.S. Hoogmoed	PER
recorder	Grouw, H.J. van	PER
sub-species	marmoratus	TAX
species	pseudolemniscatus	TAX
genus	Anolis	TAX
family	Polychrotidae	TAX
order	Sauria	TAX
class	Reptilia	TAX

Table 1: Fields from which gazetteers were extracted.

While the gazetteers can be used to identify entities directly via look-up, they can also be used to automatically extract and label training examples (i.e., named entities in context) for a named entity tagger (see Section 2.). Once such a tagger has been trained, it should be able to exploit contextual and word internal cues to identify named entities even when they are not in the original gazetteer lists.

We used one of the free text fields (*special remarks*) to obtain training and test data for the experiments. When we carried out the experiments, the *special remarks* field contained around 2200 filled cells. We randomly selected 10% of these for testing and the remaining 90% for training. The text in the test and training sets was then tokenised[6] and the test set was manually annotated with named entity information. In the training set, all expressions for which an exact match was found in one of the gazetteer lists were automatically labelled with the corresponding tag; all other tokens were labelled as OTHER. Table 2 shows the number of tokens in the data sets and the number of tokens for each entity type and for non-entities (OTHER).

The tagged examples were then used to train a memory-

[6]We used a rule-based tokeniser for Dutch developed by Sabine Buchholz.

	PER	LOC	TAX	OTHER	overall
training set	555	265	253	22,972	23,545
test set	90	42	68	2,590	2,790

Table 2: Distribution of token labels over the training and test sets.

based classifier (TiMBL, (Daelemans et al., 2004)) to identify named entities in free text fields. We employed 65 features, encoding the following information for the target token, the two tokens to the left and the two tokens to the right:

- the token (t) itself
- whether t is capitalised
- whether t contains one or more digits
- whether t contains one or more letters
- the length of t in characters
- whether t contains punctuation
- whether t contains a hyphen
- whether t is utterance/sentence-initial
- whether t is utterance/sentence-final
- the first 4 letters of t
- the last 4 letters of t
- whether t looks like an initial (e.g. E., EM etc.)
- whether t occurred in one of the gazetteers

TiMBL's parameters (similarity metric, number of nearest neighbours etc.) were set by running a heuristic search algorithm on the training set (wrapped progressive sampling, see van den Bosch (2004)).

5. Comparing Three Tagging Strategies

We tested the database-trained tagger on the test set. For comparison, we also applied a look-up tagger, which used our automatically created gazetteers, and a generic named entity tagger for Dutch (Bogers, 2004). The generic tagger was trained and tested on the CoNLL 2002 shared task data (Tjong Kim Sang, 2002). It makes use of a variety of features, such as orthographic information, morphological information, and part-of-speech tags. The tagger distinguishes five classes (PERSON, LOCATION, ORGANISATION, MISCELLANEOUS NAMES, and OTHER). It is reported to perform at around 70% average F-score.

We evaluated all three taggers on a token-by-token basis, calculating precision (P), recall (R) and F-score (F).[7] As the generic tagger does not have the entity category TAXONOMIC, we only evaluated this tagger on PERSON and LOCATION. Table 3 shows the results.

The look-up tagger performs surprisingly well, with an average (macro) F-score of 75.80%. This is in line with previous research which found that good gazetteers combined with a look-up strategy can be remarkably successful (Maynard et al., 2004; Mikheev et al., 1999; Palmer and Day, 1997). The relatively high recall values for the entity classes (in particular for PER) suggest that there is a lot of overlap between entities mentioned in the entity specific database fields and those in the free text fields. The precision is also very high, as one would expect for a look-up strategy. There are two reasons why precision is not perfect. First, some words are ambiguous between two entity classes. For example, *Virginia* is not only a person name but also a location and part of a taxonomic name (*Virginia striatula*). Similarly, *Iguana* usually falls in the category TAX, but in the expression *Reptilien Zoo Iguana* it should be tagged as OTHER. As the look-up tagger does not use context, it cannot disambiguate between these cases. Another reason for the imperfect precision is that our automatically built gazetteers are not entirely noise-free. For example, the gazetteer for LOC contains the word *vindplaats* (place of collection), due to the fact that one of the location columns contained the expression *vindplaats: Garoet, Java* and this was not filtered out during the gazetteer construction.

The database-trained classifier performs worse than the look-up tagger. It achieves an average (macro) F-score of 68.65%; for all three entity classes both recall and precision are lower than for the look-up tagger. This can be due to several factors. First, it is possible that our training set was simply too small, especially with respect to the three entity classes (with 555 tokens of type PER, 265 of type LOC and 253 of type TAX, see Table 2). It is also possible that differences between the test and training data have a negative effect. For instance, person names in the entity specific fields typically do not contain initials, hence initials are not learnt as being part of a name and are thus frequently misclassified as OTHER in the test set. We experimented with various strategies to improve the results obtained by the database trained tagger (e.g., classifier stacking, combining gazetteer look-up with our classifier etc.), but we were not able to obtain results that are significantly better than those obtained by the look-up tagger.

Finally, for the generic tagger, the results are quite low (around 33% average F-score) and in fact much lower than the results reported for this tagger when testing on newspaper texts (around 70% average F-score). This provides evidence for our hypothesis that a generic named entity tagger is not suited for this type of data. This seems to apply even to non-domain-specific categories, such as PERSON and LOCATION; probably due to domain-specific differences between the training set for this tagger, which came from a news domain, and the test set, which came from a natural history domain, with its fragmented full-text cell content. It is possible that these differences could be partially overcome by using a named entity tagger that was trained on *spoken* language, as spoken language may be more similar to the language used in textual databases with respect to the degree of fragmentation of sentences.

6. Conclusion

We investigated whether it is possible to bootstrap a named entity tagger for textual databases by exploiting the fact that such databases typically contain several entity-specific fields. These fields can be used to automatically extract gazetteer lists, which can then be used by a look-up tag-

[7] *Precision* (P) is the number of correctly tagged tokens for a given entity category divided by all tokens which were tagged with this category. *Recall* (R) is the number of correctly tagged tokens for a given entity category divided by the number of all tokens with this category in the manually annotated data. The *F-Score* is defined as $\frac{2PR}{P+R}$.

	Look-up			Database Trained			Generic		
	P %	R %	F %	P %	R %	F %	P %	R %	F %
PER	84.42	72.22	77.84	79.01	71.11	74.85	34.12	32.22	33.14
LOC	65.51	45.23	53.52	56.00	33.33	41.79	75.00	21.43	33.33
TAX	97.56	58.82	73.39	86.11	45.59	59.62	—	—	—
OTHER	97.47	99.46	98.45	97.28	99.46	98.35	—	—	—

Table 3: Results for the three taggers.

ger or to bootstrap training material for a domain-specific entity tagger. We compared the performances achieved by (i) a database-trained named entity tagger, (ii) a look-up tagger which utilises the gazetteer lists extracted from the database, and (iii) a generic named entity tagger for Dutch. We found that the look-up tagger performed best, with an average F-Score of 76%. This indicates that there is a lot of overlap between named entities in the entity-specific fields and in the free-text fields. The database trained tagger performed less well. While the tagger still achieved a reasonable performance, its average F-Score (69%) was significantly below that obtained by the look-up tagger. We believe that the main problem lies in the relatively small training set. A better strategy for future work might thus be to train on external data from the natural history domain. This could be extracted from the web by using the gazetteer lists to automatically generate suitable queries, or it could be extracted from scientific papers published by Naturalis researchers or other experts in the field. Finally, the generic tagger only achieved F-Scores of around 33%, which suggests that generic named entity taggers may not be very suitable for our type of data.

Acknowledgments The research reported in this paper was funded by NWO (Netherlands Organisation for Scientific Research) and carried out at the Naturalis Research Labs in Leiden. We would like to thank the staff at Naturalis and the three anonymous reviewers for their comments and suggestions.

7. References

A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT-98)*, pages 92–100.

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

Kalina Bontcheva, Diana Maynard, Hamish Cunningham, and Horacio Saggion. 2002. Using human language technology for automatic annotation and indexing of digital library content. In *Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries (ECDL-02)*.

Sabine Buchholz and Antal van den Bosch. 2000. Integrating seed names and ngrams for a named entity list and classifier. In *Proceedings of the Conference on Language Resources and Evaluation (LREC-00)*.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*.

Silviu Cucerzan and David Yarowsky. 1999. Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of the Joint SIGDAT Conference on Empirical Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch, 2004. *TiMBL: Tilburg Memory Based Learner, version 5.1, Reference Guide*. ILK Research Group Technical Report Series no. 04-02.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the Conference on Natural Language Learning (CoNLL-03)*, pages 168–171.

Winston Lin, Roman Yangarber, and Ralph Grishman. 2003. Bootstrapped learning of semantic classes from positive and negative examples. In *Proceedings of the ICML-03 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*.

Diana Maynard, V. Tablan, and Hamish Cunningham. 2004. NE recognition without training data on a language you don't speak. In *Proceedings of the ACL Workshop on Multilingual and Mixed-language Named Entity Recognition: Combining Statistical and Symbolic Models*, Sapporo, Japan.

Andrei Mikheev, Marc Moens, and Claire Grover. 1999. Named entity recognition without gazetteers. In *Proceedings of Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL-99)*, Bergen, Norway.

D. Palmer and D. Day. 1997. A statistical profile of the named entity task. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97)*, Washington, DC.

William Phillips and Ellen Riloff. 2002. Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-02)*.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL-02)*, pages 155–158.

Antal van den Bosch. 2004. Wrapped progressive sampling search for optimizing learning algorithm parameters. In *Proceedings of the 16th Belgian-Dutch Conference on Artificial Intelligence*, pages 219–226.