Named Entity Recognition on Flemish audio-visual and newspaper archives

Johannes Deleu, An De Moor, Thomas Demeester
Brecht Vermeulen, Piet Demeester
INTEC - IBCN - IBBT
Ghent University, Ghent, Belgium
firstname.lastname@intec.ugent.be

ABSTRACT

This paper describes a number of specific issues that we needed to deal with, in order to compose an accurate Named Entity Recognition tool on multimedia archives in Dutch. The considered data consists of archivation metadata from video collections, and large newspaper collections. For the video collections, the main challenge is to cope with a lack of capitalization in the metadata. To this end, specific capitalization features are calculated from Wikipedia. For the newspaper collections, the main concern is to create a system that maintains its performance over the course of many years. For that goal, special clustering features allow dealing with words that have not been encountered in training data. Results for the different components of the tool are reported on the target data, as well as on publicly available test data.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

1. INTRODUCTION

We present a Named Entity Recognition (NER) system that was initially developed to run on the data of the "Vlaanderen in Beeld" (VLIB) project [4], involving several Flemish broadcaster archives, and was further developed for an extensive newspaper collection, in the framework of the "TExSIS" [3] project. Our main goal was to obtain a system with the highest possible performance, given the (considerable) impediments of the data, the Dutch target language, and with a wide variety of possible techniques to tweak the NER engine at our disposal. Therefore, we describe the main design and a number of particular properties of our NER engine. We present the results according to the evolution of the system. For reference, we also show its performance on a publicly available test collection.

The Flemish research project VLIB was funded by the IWT (agency for Innovation by Science and Technology), and focused on the creation of a prototype multimedia archive for the Flemish broadcasters. The prototype archive contains roughly about 10.000 hours of video material, including audio data, and a large amount of textual metadata. About half of this material relates to news and contains a fair amount of textual metadata. The full text search functionality for the archive was provided by indexing these metadata (i.e., textual descriptions and keywords).

As the content of the archive was provided by different partners, an important step was to bring together the metadata into a unified model. Due to the individual procedures of adding metadata, there was an important need for data normalization, such as merging keywords from different thesauruses together (e.g., Kongo / Zaire), correcting typographical errors, and so on. Moreover, it appeared essential to fill in missing data as a considerable amount of documents did not contain any keywords at all. We therefore started looking for ways to automaticaly extract information from the full text descriptions, and use these to fill in and correct the metadata fields.

Named Entity Recognition (NER) appeared to be an attractive means for that goal. It would enable the automatic extraction of names such as people, locations and organizations from the full text. Unfortunately, existing Named Entity Recognizers did not perform well on our data. Most systems are trained on a collection of news articles. The quality of metadata written for archivation purposes is typically less critical. The text fragments in the multimedia archive are often written in a very condensed form, using domain-specific terminology, and with a lot of editorial issues (such as the lack of capitalization). The new NER system was initially built to circumvent these problems with a number of special measures.

In a further stage, during the TExSIS project, the NER engine was used on a large data set from Mediargus [2], initially containing the Flemish newspapers' content of 2011. Although the quality of this collection was clearly superior, the main challenge is to build a system that works reliably over a longer period of time, as the number of new words is expected increase quickly over time.

The paper is outlined as follows. The next section describes the architecture of the NER engine and highlights where it can be most clearly distinguished from other systems. Section 3 describes a number of experiments and overal performance results. Section 4 describes possible future work directions. The last section concludes the paper.

2. NAMED ENTITY RECOGNITION

The standard approach to NER is to use a machine learning technique to learn a classifier from a manually annotated data set.

In this setting, a classifier is built that labels each token in the sentence with a category and segmention label. The category label indicates the entity type of a word: PER (person), LOC (location), ORG (organization) or MISC (miscellaneous). The segmention label specifies whether a token begins (B), is inside (I) or outside (O) an entity (hence called BIO-encoding).

Learning algorithms are unable to operate directly on text. First, sentences have to be encoded into a numerical representation, by defining features that extract signal from text. Features are typically binary predicates that test simple conditions (e.g., word at position i equals "postbode", word at position $i+1$ contains a digit, and so on).

Most of our work went into designing good features. Our baseline model mirrors publicly available systems (in particular [1]). It includes word features, shape features, character n-grams (prefixes and suffixes) and a limited set of conjunctions. Further on, we will describe some extra features, namely, capitalization and clustering features, that appeared essential to increase the performance of the baseline system.

2.1 Conditional Random Fields

As a classification algorithm we use linear-chain Conditional Random Fields (CRFs) [7], a very popular yet higly competitive algorithm. CRFs are discriminative models; they learn a direct mapping from the input (feature) space to the output (class label) space, without putting any effort in modeling the input variables' distribution (as a Hidden Markov Model would do). This results in a great flexibility in handling large numbers of arbitrary features, even if they overlap and are inter-dependent. As such, virtually any useful feature of the input observation can be included into the model.

CRFs are structured classifiers, and are therefore also able to deal with dependencies between output labels. In contrast to many models (such as SVMs and plain logistic regression), CRFs are able to correctly model likely and unlikely labeling transitions in sentences. Examples of such transitions are: 'successive words are more likely to have the same entity type', or 'new entity types always begin with a B segmentation attribute' (i.e., <O> <I-PER> is not allowed).

2.2 Capitalization Features

For the older documents in the VLIB archive, the textual metadata originate from computer systems that only supported full capitalized input. Capitalization is however a very strong source of information to determine where an entity begins, in Dutch and English text. In order to realize a decent performance on such documents, we first tried to restore capitalization with a sequence tagger. In the end, simple capitalization statistics gathered from Wikipedia proved much easier and performed better. For this we define 5 capitalization classes c: lowercased, capitalized, all uppercased, mixed case and no case. As input features to the CRF we give the numerical value of the expectations $p(c|w)$

$$p(c|w) = \frac{N(w,c)+\lambda}{\sum_c (N(w,c)+\lambda)}$$

in which $N(w,c)$ is the number of occurrences of word w in capitalization class c and λ is a smoothing constant.

2.3 Word Clusters

One of the key challenges of any Natural Language Processing (NLP) system is to make robust decisions on new texts, which possibly may deviate from the trainig data. Because it will never be possible to gather all words in a training set, the way in which words are represented is crucial.

In its most basic form, NLP systems represent words as binary features. For each position in the sentence, only one of those features is active. This representation fails when a new, out of vocabulary, word has to be encoded. One approach is to introduce a smaller vocabulary of word clusters, grouping similar words into clusters. To obtain such a clustering, an unsupervized algorithm is run on as much data as available (i.e., the whole corpus instead of only the annotated data). By optimizing an objective function, each word is assigned to one cluster. Later, when a word is encountered that has never been seen in the training data, the classifier can make decisions based on the cluster that it belongs to.

One of the oldest clustering methods, which is still very competitive, is that of brown clusters or class-based n-grams [5]. We use predictive exchange clustering (PEC), which is a very fast variant for which also a distributed version exists [8]. PEC works by searching for the optimal clustering that maximizes the log likelihood of the input data

$$\begin{aligned} L(\mathbf{w};\mathbf{c}) &= \sum_i \log p\left(w_i|w_{i-1}\right), \\ &= \sum_i \log p\left(w_i|c\left(w_i\right)\right)\, p\left(c\left(w_i\right)|w_{i-1}\right). \quad (1) \end{aligned}$$

Each term in the above summation consists of a part that predicts the current cluster given the previous word, and a component that predicts the current word given the current cluster. Using word frequencies, (1) can be rewritten as follows

$$L(\mathbf{w};\mathbf{c}) = \sum_{w,c} N(w,c)\log N(w,c) - \sum_c N(c)\log N(c), \quad (2)$$

with $N(w,v)$ the frequency of bigram (w,v) and $N(w,c)$ the frequency of word w followed by cluster c.

$$\begin{aligned} N(w,c) &= \sum_{v\in V(c)} N(w,v) \\ N(c) &= \sum_w \sum_{v\in V(c)} N(w,v) \end{aligned}$$

When moving a single word from one cluster to another, only a limited number of terms in (2) are affected, enabling one to evaluate the gain in an efficient manner.

The algorithm begins with a random assignment of words onto clusters. It then iterates over all words and tries to move them individually to other classes. Words are moved

cluster 48	cluster 52	cluster 69	cluster 76	cluster 96
meer	*hebben*	twee	sint	gilber
beter	hadden	drie	brussel	laurent
minder	*hopen*	vier	antwerpen	cancellara
langer	wisten	vijf	mechelen	kadhafi
hoger	vrezen	tien	leuven	vangheluwe
groter	staken	zes	hasselt	*kbc*
sneller	dachten	*verschillende*	oostende	boussoufa
sterker	*mikken*	zeven	kortrijk	albert
lager	verdienden	acht	aalst	bin
vaker	zochten	*vele*	turnhout	*toerisme*

Table 1: Word clusters

only if it leads to an increase of the objective function. This algorithm terminates when no new moves are found. At this point, a local optimum is reached.

The resulting clustering is syntactic in nature; words that fulfill a similar function within the sentence, are placed in the same cluster. An example is given in Table 1, displaying the ten most frequent words in five out of two hundred clusters. Although the table contains some questionable assignments, the NER classifier is supposed to deal with that.

2.4 Phrase Clusters

An alternative clustering algorithm was proposed in [6], for clustering phrases using the k-means algorithm. It was argued that the disambiguation power of phrases is stronger than that of words only. Word clusters have their limitations, because words out of their proper context are often ambiguous. For example, considering the indivual words of the phrase "Jan De Nul", one could easily mistake it for a person. Looking at the context of this phrase in a large corpus, it becomes clear that the phrase actually represents a company.

In a first step, by collecting word occurences in a small window around each phrase, a context vector $\mathbf{x}_{phr}(w)$ is created. The authors of [6] obtained a list of phrases from an anonymized query log. Because we do not have access to such resources, we first created a list of Named Entities as phrases, using an intermediate version of the NER engine. All occurrences of these phrases are then looked up over the entire corpus and their surrounding context is aggregated, including the occurrences that may have initially been missed by the NER engine. The collected frequency counts are then rescaled using point-wise mutual information

$$\mathbf{x}_{phr}(w) = \log\left(\frac{N(phr,w)}{\sum_{phr} N(phr,w)\sum_{w} N(phr,w)}\right).$$

The k-means algorithm creates the final clusters by maximizing the following objective function:

$$E(\mathbf{c}) = \sum_i \frac{\mathbf{x}_i \cdot \mathbf{c}_{k(i)}}{\left\|\mathbf{c}_{k(i)}\right\|}.$$

Initially, (i) a number of random phrases are taken as the cluster seeds, (ii) each phrase is assigned to the most similar center using cosine similarity, (iii) the centers are recomputed, and (ii) and (iii) are repeated until no reassignments occur. The centers are recomputed by averaging the feature vectors of the cluster elements (thus maximizing the above objective function, given the elements of a clustering).

cluster 13	cluster 64
ronde van catalonië	tc de meyl
tweedaagse van de gaverstreek	tc westerlo
ronde van de limousin	tc de zwalum
haspengouw tour	smash neeroeteren
ster zlm toer	voco tennisclub
cluster 94	**cluster 120**
red riding hood	dag van de open kerken
mr. popper's pengiuns	wings and wheels
the hangover 2	landjuweelfestival
rio-3d	week van de fair trade
sucker punch	deistelrock

Table 2: Phrase clusters

data set	documents	tokens	terms
conll-train	287	207066	25306
conll-testa	74	38413	7629
conll-testb	119	70071	10917
vlib-train	655	41784	8578
vlib-test	345	24948	5847
mediargus11-train	124	45113	8236
mediargus11-test	74	23845	5264

Table 3: Data sets to evaluate experiments

Table 2 lists the best scoring phrases of 4 clusters (out of 128 clusters) obtained from running it on the Mediargus data set of 2011, for which a window size of 3 was arbitrarily chosen.

3. DATA AND RESULTS

In this section we describe in chronological order the experiments we performed. The total number of tokens and the number of unique terms within human-annotated data sets are given in Table 3. Although the VLIB and Mediargus data are not publicly available, these numbers should give a good idea of the size of the annotated data set in comparison to the publicly available CoNLL data set. The labeled data set contains about 1000 documents from the VLIB archive, but is equal in size to the second CoNLL test set. The Mediargus data set is also about the same size.

We start with a system trained on VLIB data only. We gain about 2.5% in F1 measure by adding the CoNLL-2002 data to the training set, which is still far below test accuracy on CoNLL test sets (which is about 78% and 79% respectively). Enabling capitalization features brings us to CoNLL performance, clearly confirming that capitalization was indeed the main issue. Another 3% gain in performance is obtained by enabling word clusters (from the unlabeled data of VLIB and a complete Wikipedia dump).

Testing this version of the NER engine on the Mediargus test set, without further training, resulted in a lower score. After including the Mediargus training set and performing word clustering on the complete collection of news articles of 2011, performance increased to above that of VLIB. The inclusion of phrase clusters resulted in our best and final model.

To summarize, we list the performance of the final model on all data sets in table 6. These results are obtained by training on CoNLL 2002, VLIB and Mediargus. Word clusters

	Pr	Re	F1
vlib only	83.2	64.3	72.6
+CoNLL data	83.9	68.0	75.1
+capitalization	82.2	77.5	79.8
+word clusters	85.3	80.3	82.7

Table 4: Precision, recall and F1-measure for a NER evaluated on the VLIB test set

	Pr	Re	F1
version 1	75.9	77.6	76.7
+mediargus11-train	82.8	81.4	82.1
+word clusters	85.2	84.0	84.6
+phrase clusters	87.0	86.1	86.6

Table 5: Precision, recall and F1-measure for a NER evaluated on the Mediargus test set

and phrase clusters are derived from Wikipedia, VLIB and Mediargus. Although we needed to include extra annotation data to get to these results (as compared to the publicly available training data), that annotation was needed to get the NER performing to the same level as publicly available systems. The largest increase in performance is due to the word clustering and phrase clustering.

4. FUTURE WORK

We were recently granted access to the large newspaper archive from Mediargus, with news articles from 2000 to 2008 (containing 6.3 million articles, as compared to the 2011 collection with only 0.5 million articles, used for this paper). We are hence extending the cluster features for this entire dataset, to evaluate to what extent these features are capable of allowing for accurate NER over a large time span. Considering the increase in NER performance based on word and phrase features for the current test collection, there will most likely be a significant influence.

In the near future we will investigate to what extent it is worthwhile to increase the complexity of the model. Like most other systems, we use a first order model CRF which is unable to model long range phenomena and this is visible in certain aspects of text (e.g., a concatenation of entities separated by commas or a conjunction). Also in the more general case, sequence models are unable to model multiple mentions of the same entity in a single document. Ideally, predicting the labels for related mentions should not be done independently. Modeling such interactions in a fundamentally correct way is a difficult problem and leads to approximate methods. Looking at the output of our system, it is however an important issue.

5. CONCLUSIONS

In this paper, we presented our Named Entity Recognition tool for Dutch, specifically tailored towards the metadata of a Dutch multimedia archive, but currently also trained and applied on other textual data such as news collections. Some features of the NER engine, required by the inherent limitations of the metadata, were explained, along with some of our internal evaluation results during the development phase.

	Pr	Re	F1
conll-testa	86.2	84.2	85.2
conll-testb	87.5	85.3	86.4
vlib-test	86.4	82.2	84.2
mediargus-test	87.0	86.1	86.6

Table 6: Performance of final model on all test sets

6. ACKNOWLEDGMENTS

This work was in part carried out in the framework of the projects VLIB and TExSIS, sponsered by the agency for Innovation by Science and Technology (IWT).

7. REFERENCES

[1] http://www-nlp.stanford.edu/ner/.
[2] http://www.mediargus.be/.
[3] Terminology Extraction for Semantic Interoperability and Standardization (TExSIS). http://www.vlaandereninbeeld.net.
[4] Vlaanderen In Beeld (VLIB). http://www.vlaandereninbeeld.net.
[5] C. Christodoulopoulos. Two Decades of Unsupervised POS induction: How far have we come? *Proceedings of the 2010 ...*, 2010.
[6] D. Lin and X. Wu. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, volume 2, page 1030, Morristown, NJ, USA, 2009. Association for Computational Linguistics.
[7] C. Sutton and A. McCallum. An introduction to conditional random fields. *Arxiv preprint arXiv:1011.4088*, 2010.
[8] J. Uszkoreit and T. Brants. Distributed word clustering for large scale class-based language modeling in machine translation. *Proceedings of ACL-08: HLT*, 2008.