Context-Based Person Identification for News Collections

Laurent Mertens, Thomas Demeester, Johannes Deleu,
Chris Develder, Piet Demeester
INTEC - IBCN
Ghent University, Ghent, Belgium
firstname.lastname@intec.ugent.be

ABSTRACT

In modern automated information extraction systems, Named Entity Disambiguation (NED) techniques are becoming increasingly important. The ambiguity of person names leads to a decrease in the output quality of search engines. This paper presents a two-stage rule-based NED model, based on a local and global context of the mentioned persons. A number of experiments with different scoring functions are reported, as well as a specific evaluation method to estimate the efficiency of the model on a real-life data collection in an unsupervised way.

Categories and Subject Descriptors

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

General Terms

Named Entity Disambiguation, Clustering

1. INTRODUCTION AND BACKGROUND

In the research areas of Natural Language Processing and Information Extraction, Named Entities are entities that are referred to by a proper name, such as persons and organisations. These entities are particularly relevant to search engines, as they often correspond to query keywords. This keyword property also leads to other important application areas, such as the development of the Semantic Web, and Information Extraction techniques in general. A lot of research has been done on automatic Named Entity Recognition (NER), see e.g. [6] for a good overview of the early work on NER.

The goal of NED is to resolve all Named Entities found in a text to a unique "real" instance, hereby correctly clustering the different forms as which an entity appears (e.g., only the first or last name of a person), also called "surface forms", and taking into account the fact that one mention can refer to several entities (e.g., distinct people sharing the same name). Several NED techniques have been proposed in literature, and many researchers have explored the potential of Wikipedia as a means of evaluating such a system. For example, Bunesco and Pasca [1] used the Wikipedia content and structure to train an SVM-based NED engine. A vector space model for NED was proposed by Cucerzan [2], strongly based on Wikipedia as well. He also provides a nice overview of earlier work on coreference resolution and NED. Further contributions along the same line, i.e., also based on Wikipedia, are [7] and [5]. A different approach is presented by Habib and Van Keulen [4], who explore the possible interplay between extraction and disambiguation of Named Entities as a means to improve accuracy.

This paper presents a number of experiments involving a rule-based NED technique tailored towards a collection of news articles. We focus our attention on person names, and the NED process, henceforth called the **IDDiscriminator**, boils down to mapping each mention of a person in the collection to the best matching entry in a gazetteer, an external reference repository, denoted as the **RefIDBank**. Note that this external list determines the resolution of the IDDiscriminator, in the sense that distinct persons sharing the same name can, at present, not correctly be distinguished. In order to determine the best match, the "context" of the article in which a surface form occurs, is compared to a typical context of a particular candidate. This context is considered on two different levels. First, the Named Entities most often co-occuring with the considered surface form and candidate match are taken into account. Secondly, the textual context of the article is compared to the typical local textual context of the candidate matches. The general workflow of the IDDiscriminator is depicted in Fig. 1, the different components of which are explained in Section 2. Section 3 presents some experiments, in particular focusing on a new non-supervised evaluation method. Finally, Section 4 summarises the main results.

2. NED MODEL

2.1 Data Preparation

2.1.1 Data Collection

The raw data consists of a dataset of news articles from Dutch newspapers, spanning a time range from March 1st 2011 untill August 30th 2011, and totalling 353798 items. This data has been provided by Flemisch media consortium Mediargus (`http://www.mediargus.be`).

2.1.2 The Named Entity Recogniser

First, the articles are parsed by a Named Entity Recogniser [3]. Four categories are distinguished by the NER engine: persons (PER), organisations (ORG), locations (LOC) and miscellaneous (MISC). The total number of distinct PER surface forms in the entire dataset amounts to 264823.

DIR 2012 23–24 February 2012, Ghent, Belgium

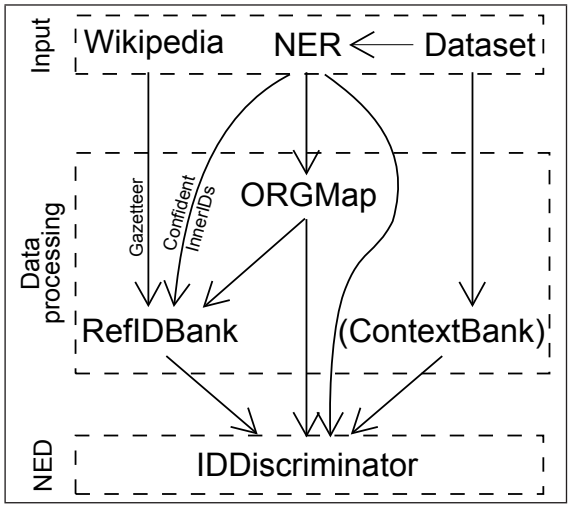

Figure 1: Graphical workflow overview.

2.1.3 The InnerIDs

Before resolving names over the entire dataset, the PERs are clustered per article. The longest item per such cluster is referred to as an **InnerID**, and the cluster will contain all substrings of this InnerID that appear in the article. Suppose, e.g., an article contains the PERs "Bart", "De Wever", "Bart De Wever" and "Di Rupo", then the first three will define one cluster, while the last one will form a cluster on its own. The two InnerIDs of this article are then "Bart De Wever" and "Di Rupo". Note that this leads to a reduction in the number of distinct original surface forms with only 5%.

2.1.4 The RefIDBank

The aim of the IDDiscriminator is to look in an external repository for the best match to a certain mention of a person. This repository is called the RefIDBank, and its entries are the **RefIDs**.

The candidate RefIDs are gathered from two sources, namely Wikipedia, and a list of "confident entities", i.e., entities for which the NER engine label assignment confidence exceeds a certain threshold, taken from the NER output [3] for the news dataset. For each candidate that appears as an InnerID in at least one article, the application will gather a list of the x most frequent PERs and ORGs that co-occur with this particular name in the dataset, by analysing the articles in which it appears as an InnerID, and where x is a user-defined parameter. As far as the PERs are concerned, only "full names" are considered, i.e., names that are composed of more than 1 word, and PERs that are also RefIDs are given more weight. For the ORGs, the "ORGMap" idea as described in 2.1.5 is applied. The resulting most commonly co-occuring x entities (or less if there aren't enough), are then listed in descending order of co-occurence frequency in the "RefIDBank of length x", one entry of which is shown in Fig. 2.

2.1.5 The ORGMap

Often, several surface forms of the same organisation appear in the data. For example, Coca-Cola might appear as "Coca-Cola", "Coca Cola", "Coke", etc. When one simply determines the organisations that co-occur most frequently with a certain person, all of these different variations will be considered to be different entities as well, while in fact, they all refer to the same organisation. In order to deal with this issue, the ORGs are converted to a normalised writing, then clustered, and finally mapped to the shortest common part they share.

```
Elio Di Rupo:
   ORGs:   N VA     PS     CD&V   VLD   [...]
   PERs:   Bart De Wever   Wouter Beke  [...]
```

Figure 2: One entry from the `RefIDBank`.

The "normalising" step consists of converting every ORG to a standard form by applying a number of formatting rules. Subsequently, the normalised organisations are grouped into clusters that start or end with a same common shortest ORG, which is used to identify the cluster. For example, the cluster identified by "KIWANIS" contains items like "KIWANIS ANTWERPEN" and "JONG KIWANIS". Finally, each normalized ORG is mapped onto the identifier of the cluster in which it is contained. This mapping is called the ORGMap, and in the RefIDBank the ORGs are represented by these identifiers.

2.1.6 The ContextBank

The ContextBank is a document that lists, per RefID, those words that frequently appear within a certain "context window" around mentions of that RefID. The way this document is obtained, goes as follows: for all mentions of the RefIDs in the dataset, gather all words within the context window, and count how many times each word is observed. Once these data have been gathered, each RefID is scored by normalising these counts by the total number of times it appears in the dataset. Finally, retain those words with a score > 0.001, a well-chosen cutoff value.

The ContextBank used for the runs described in this article were obtained with a context window of 3 words preceding and following the mentions of the RefIDs. Note however that its use can be turned off, which we use to evaluate its significance.

2.2 Person Identification Algorithm

Now that the data preparation steps for the IDDiscriminator have been described, the main algorithm, mapping InnerIDs on RefIDs, is succinctly described.

2.2.1 Algorithm Overview

The pseudocode for the algorithm is depicted in Figure 3. First, the InnerIDs for all documents are computed, i.e., all PERs are clustered per article, thereby taking into account possible spelling mistakes.

In a second step, the algorithm proceeds to finding the best match in the RefIDBank for every InnerID in every document. For this, it is first verified if the InnerID is also a RefID, in which case it is assumed that it is its own best match. If this is not the case, the set of all RefIDs that contain the InnerID as a subset will be gathered, and the context of the document will be compared to the context

```
IDDiscriminator(D, R):

1. for doc ∈ D
     InnerIDs[doc] = ComputeInnerIDs(doc);

2. for doc ∈ D:
     for ID ∈ InnerIDs[doc]:
       if ID ∈ R:
         BestMatch = ID;
       else:
         RefIDMatches = GatherMatches(ID);
         BestMatch = null;
         if !RefIDMatches.isEmpty():
           for RefID ∈ RefIDMatches:
             Scores[RefID] = ComputeScore(RefID);
             if max Scores[RefID] > 0:
               BestMatch = arg max Scores[RefID]
       return BestMatch;
```

Figure 3: Pseudocode for IDDiscriminator. The arguments are the dataset D and the RefIDBank R.

of these RefIDs, as stored in the RefIDBank, and optionally also the ContextBank. This comparison will result in a score, and the RefID with the highest score will be taken as the best match for the considered InnerID.

2.2.2 Scoring Function

The scoring proceeds as follows. For a given InnerID in a given document d, and for every candidate RefID match, the (other) InnerIDs and ORGs contained in that document will be compared with the PERs and ORGs stored in the RefIDBank for that RefID. Starting from 0, the score will be incremented for every match between both lists. Define a "full match" as one where an InnerID appears *as such* in the RefIDBank, as opposed to a "partial match" where it appears "contained within" a PER listed in the RefIDBank (e.g., "Bart" is a partial match for "Bart De Wever"). Partial matches only count once, e.g., suppose an InnerID appears as a subset of 3 different PERs, only the first one (the one with the lowest position) is accounted for. For ORGs, only full matches are considered.

Five different scoring functions were experimented with. These are depicted in Table 1. Note that positions are counted starting from 0, up till $N-1$, where N is the length of the RefIDBank, and that PERs are compared before ORGs. The used abbreviations correspond to: normalised (Norm), One-over-X (OOX), One-over-square-root (OOSR), Linear Down (LinDn) and Linear Up (LinUp). Note that the idea was not to design an optimised scoring function, as the weights are still chosen arbitrarily, but rather to evaluate different tendencies.

If the system is used with the ContextBank, the score will be additionally incremented with the score for every word that appears both in the entire document and the ContextBank entry for the candidate RefID, and this regardless of the scoring function used.

3. EXPERIMENTS & RESULTS

	Full PER Match	Partial PER Match	ORG Match
Norm	$+2$	$+1$	$\times 2$
OOX	$+2\,(X+1)^{-1}$	$+\,(X+1)^{-1}$	$\times 2$
OOSR	$+2\,(X+1)^{-1/2}$	$+\,(X+1)^{-1/2}$	$\times 2$
LinDn	$+2\,(1-X/N)$	$+\,(1-X/N)$	$\times 2$
LinUp	$+2(X+1)/N$	$+(X+1)/N$	$\times 2$

Table 1: Used scoring functions, with X the position of the match in the RefIDBank vector, and N the length of the RefIDBank.

To test the accuracy of the IDDiscriminator, a number of experiments were carried out.[1] For these tests, several new datasets were created from the original dataset. The idea is to take names of which one is confident, i.e., the RefIDs, take all the articles in which these names appear (as InnerID), and replace the surface forms of these InnerIDs at random with either only the first or only the last name of this RefID.[2] This way, one knows in advance what RefID these particular mentions should be resolved to by the IDDiscriminator, and furthermore creates a worst case scenario, in which no RefIDs appear as such anymore, and hence all surface forms need to be resolved using the context information.

Five different sets were created, using the x RefIDs that appear most frequently, with $x \in \{50, 100, 150, 200, 250\}$, and keeping only the articles in which they appear. These datasets will hitherto collectively be referred to as the "Top X datasets". How many documents each of these datasets comprise, can be seen in Table 2.

#Top RefIDs	# Docs	% Total Docs
50	25047	7.1
100	34647	9.8
150	40080	11.3
200	43493	12.3
250	46742	13.2

Table 2: Size of the Top X datasets.

3.1 Evaluation Method

The output of the clustering tool is a document that specifies for each surface form to which InnerID, which possibly is also a RefID, it gets mapped. But for the Top X datasets, we know in which documents these RefIDs originally appear, and have replaced each explicit RefID mention by only its first or last name. As such, after the cluster tool has done its work, it is possible to verify for each of these Top X RefIDs in how many of the documents in which they should appear, the tool correctly predicted the right mapping. By dividing the number of correctly resolved documents by the number of documents in which the RefID should appear, one obtains a score from 0 to 1, indicating the resolving efficiency of the tool for that particular RefID. The score for

[1]Note that NED techniques are often evaluated using the disambiguation pages of Wikipedia. However, this is not possible here, given the insufficient overlap between the used news archive and Wikipedia in terms of persons.

[2]By "first name" we understand the first word, i.e., everything preceding the first whitespace, of a "full name".

a Top X dataset is then the average of these scores taken over all Top X RefIDs. In the subsequent paragraphs, this score is referred to as "accuracy".

3.2 Scoring Functions

First of all, it was verified which of the scoring functions depicted in Table 1 performed best. Runs using these different functions were executed on the Top X datasets, for RefIDBank lengths 10, 20, 30, 40 and 50, without using the ContextBank. The results, including the running times, are shown in Table 3. It can be seen that the OOSR function performed best, although not by much, and hence this is the function that is used in the IDDiscriminator for all other runs discussed in this article.

	Norm	OOX	OOSR	LinDn	LinUp
Top 50	0.572	0.548	**0.583**	**0.583**	0.524
	1:49:35	1:52:12	1:49:20	2:05:05	1:59:33
Top100	0.590	0.566	**0.597**	0.595	0.543
	3:19:31	3:08:12	3:27:00	3:09:35	3:08:27
Top150	0.598	0.571	**0.602**	0.601	0.553
	3:53:08	3:08:45	4:18:02	3:55:03	4:24:59
Top200	0.612	0.584	**0.614**	0.613	0.567
	3:29:32	3:56:51	3:38:18	4:35:56	4:03:56
Top250	0.620	0.590	**0.621**	0.620	0.579
	4:06:25	4:08:08	3:45:08	3:47:37	4:09:50

Table 3: Accuracy and running time (in h:mm:ss) for Top X datasets for different scoring functions, all runs with ($N = 20$), and no context.

3.3 Top RefID Datasets

A further set of runs was performed on the Top RefID datasets described earlier. For every dataset, 10 runs were performed, namely two runs per RefIDBank lenghts 10, 20, 30, 40, 50 and 100. One run did not use the ContextBank, whereas the other one did. The results of these runs are shown in Fig. 4. Taking into account the context clearly leads to better results. The accuracy also increases when taking into account the context of co-occuring persons and organizations, i.e., for an increasing vector size of the RefIDBank. This increase appears to plateau if one were to carry on experiments with still higher vector sizes. It can also be observed that the added benefit of the extra context becomes less important as the length of the RefIDBank increases, because the longer the RefIDBank, the less extra relevant information is contained in the remaining context. Note however, that there is still a 6% increase in accuracy for a RefIDBank length of 100, which remains very significant.

3.4 ORGMap vs. No ORGMap

Additional runs without ContextBank were executed that did not employ the ORGMap principle, for all Top X datasets, and RefIDBank lengths 10, 20, 30, 40 and 50. These runs indicated that the ORGMap barely had any positive effect for the Top 50 dataset, with an average increase in accuracy of 0.4%, but that for bigger datasets the difference does become significant, with an average accuracy increase of 2.0% over the remaining 4 datasets. The effect however strongly decreased from the smallest to the largest RefIDBank lengths, with respective average accuracy increases of $3.0 \pm 0.3\%$ vs. $1.3 \pm 0.1\%$.

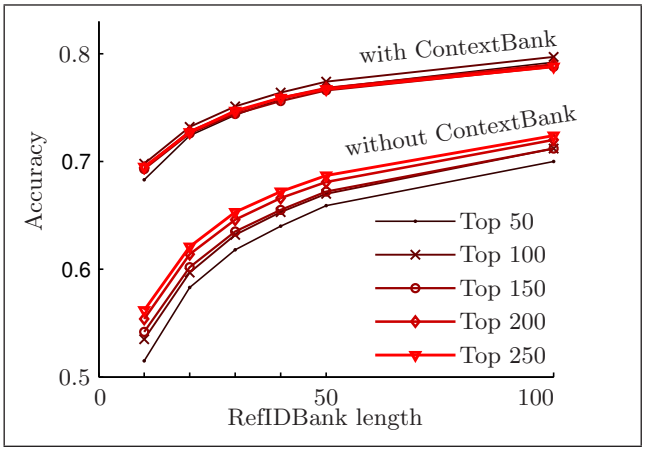

Figure 4: Influence of context on accuracy for 5 different dataset sizes

4. CONCLUSIONS

We presented a rule-based Named Entity Disambiguation model for persons, where the different occuring surface forms are mapped to the entries of a reference gazetteer, based on a local textual context and a global context of co-occuring persons and organisations. The influence of these contextual factors was demonstrated in a number of experiments, based on a basic but efficient evaluation method. Future work consists of investigating the effect of the size of the context window, as well as optimising the scoring function.

5. REFERENCES

[1] R. Bunescu and M. Pas. Using Encyclopedic Knowledge for Named Entity Disambiguation. *Computational Linguistics*, (April):9–16, 2006.

[2] S. Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. *Computational Linguistics*, (June):708–716, 2007.

[3] J. Deleu, B. Vermeulen, A. De Moor, T. Demeester, and B. Van Den Bossche. Building a dedicated Named Entity Recognizer for Dutch Multimedia Archives. In *Proceedings of the 12th Dutch-Belgian Information Retrieval Workshop*, 2012.

[4] M. Habib and M. van Keulen. Named Entity Extraction and Disambiguation: The Reinforcement Effect. In *Proceedings of the 5th International Workshop on Management of Uncertain Data*, volume WP11-02, pages 9–16. Centre for Telematics and Information Technology, University of Twente, Aug. 2011.

[5] X. Han and J. Zhao. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceeding of the 18th ACM conference on Information and knowledge management - CIKM '09*, page 215, New York, New York, USA, Nov. 2009. ACM Press.

[6] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[7] H. Nguyen and T. Cao. Named entity disambiguation: A hybrid statistical and rule-based incremental approach. *The Semantic Web*, pages 420–433, 2008.