# NERD: A Framework for Evaluating Named Entity Recognition Tools in the Web of Data

Giuseppe Rizzo[1,2] and Raphaël Troncy[1]

[1] EURECOM, Sophia Antipolis, France, <raphael.troncy@eurecom.fr>
[2] Politecnico di Torino, Torino, Italy, <giuseppe.rizzo@polito.it>

**Abstract.** In this paper, we present NERD, an evaluation framework we have developed that records and analyzes ratings of Named Entity (NE) extraction and disambiguation tools working on English plain text articles performed by human beings. NERD enables the comparison of different popular Linked Data entity extractors which expose APIs such as AlchemyAPI, DBPedia Spotlight, Extractiv, OpenCalais and Zemanta. Given an article and a particular tool, a user can assess the precision of the named entities extracted, their typing and linked data URI provided for disambiguation and their subjective relevance for the text. All user interactions are stored in a database. We propose the NERD ontology that defines mappings between the types detected by the different NE extractors. The NERD framework enables then to visualize the comparative performance of these tools with respect to human assessment.

**Key words:** Entity extraction, Linked Data, Natural Language Processing, Evaluation of Linked Data entity extraction tools

## 1 Introduction

The Web has become a large data space, where millions of semi-structured texts such as scientific, medical or news articles as well as forum and archived mailing list threads or (micro-)blog posts are available. These documents often contain rich semantics which is hidden to computing machinery. Natural Language Processing (NLP) and information extractors play a key role to extract information from unstructured or semi-structured text. Recently, Linked Data entity extractors have emerged for providing a URI that uniquely identifies named entities in the Web of Data in addition to their type.

Tools such as AlchemyAPI[3], DBpedia Spotlight[4], Extractiv[5], OpenCalais[6] and Zemanta[7] offer a clear opportunity for the Semantic Web community to increase the volume of interconnected data. Although these tools share the same

[3] http://www.alchemyapi.com
[4] http://dbpedia.org/spotlight
[5] http://extractiv.com
[6] http://www.opencalais.com
[7] http://www.zemanta.com

purpose – extracting semantic units from text – they make use of different algorithms and training data. They generally provide a similar output composed of a set of extracted named entities, their type and potentially a URI disambiguating each named entities ($o = (NE, type, URI)$). These services have their own strengths and shortcomings but to the best of our knowledge, no thorough comparison of them have ever been published in the scientific literature. This demonstration is our attempt to provide an actionable framework for performing such a comparison. In the following, we briefly describe the architecture of the NERD framework and we present an evaluation test scenario.

## 2 NERD Framework

NERD (Named Entity Recognition and Disambiguation)[8] is a web application plugged on top of various NE extractors. It allows a user to analyze any textual resource published on the web and accessible with a URI, and to extract from the text the named entities detected, typed and disambiguated by various NE extractor APIs. It provides a user interface for assessing the performance of each of those tools according to the pattern (NE, type, URI). All user interactions are collected and stored in a database. The framework can finally generate analysis reports and comparison of tools using the NERD ontology.

The NERD system architecture is composed of a front-end web interface and a back-end coupled with a SQL database. The user interface is developed in HTML/Javascript and enables a user to analyze and assess the extraction results of several NE extractors. Through asynchronous calls, the user evaluation is sent to the back-end when the client asks to store the data. The application contains a help page that provides guidance and details about the whole evaluation process. The landing page of the application includes doodles that remind the user the three important steps for conducting an evaluation.

The back-end is developed in Java and runs on an Apache-Tomcat application server. We adopted a modular implementation that is easily extensible composed of seven modules: authentication, scraping, extraction, ontology matching, store, statistics, web. The *authentication* takes as input the FOAF profile of a user and links the ratings with the user who performs them. We consider implementing WebID in the future. The *scraping* module takes as input the URI of an article and extracts all its raw text. *Extraction* is the module designed to invoke the external service APIs and collect the results according to the pattern (NE, type, URI). Each services provide its own taxonomy of type of named entity it can recognize. We therefore designed the NERD ontology[9] which provides a set of mappings between these various classifications. The *ontology matcher* is the module in charge of the NE type comparison using the ontology. The *store* module saves all ratings according to the ER model in a MySQL database. The *statistic* module enables to extract data patterns form the user interactions stored in the database, and to compute statistical scores such as the Fleiss'

[8] `http://semantics.eurecom.fr/nerd`

[9] `http://semantics.eurecom.fr/nerd/ontology/`

Kappa score and precision/recall analysis. Finally, the *web* module manages the client request and generates the HTML pages. We are currently developing a RESTful API exposing all the data.

## 3 How to Perform an Evaluation?

### 3.1 Type the URI of an article and enter your FOAF identifier

In the landing page, there are two input boxes: one for a news article URI and one for a FOAF profile. English is the only language fully supported by all NE extractors although some of them support other languages. The user provides a FOAF identifier in order to attach all evaluations with a profile and visualize a history. After clicking on the "Analyze" button, NERD extracts the raw text contained in the web document and displays a preview page.

### 3.2 Skim over the article and choose one extractor

In the preview page, the user briefly skims over it and chooses one extractor in order to execute the NE extraction task. Once a particular service is clicked, the NERD framework invokes the corresponding NE extractor API and the user is directed to the evaluation page which shows the extraction results returned by this service.

### 3.3 Evaluate the extraction results and save the evaluations

When looking at the results returned by a NE extractor, the user can still read the content preview although the named entities are not localized in the text. The user rates the correctness of the extraction using a "+" and "-" button in a table composed of three columns.

- NE: assess whether the entity extracted is a named entity ("+") or not ("-"). By NE, we meant an entity that refers to a person, a location, an organization, a product/brand, a date or a currency as opposed to a general topic. By default, the "-" is selected.
- Type: assess whether the entity extracted is typed correctly in the context of the article ("+") or not ("-"). By default, the "-" is selected. If the type returned by NE extractor is NULL or belongs to a general miscellaneous category, the user is instructed to also assess it negatively.
- URI: assess whether the URI can be used as an identifier of the named entity ("+") or not ("-"). Some extractors, such as AlchemyAPI and Zemanta, provide general URI relevant to the named entity while others disambiguate the named entity with a Linked Data URI.
- Relevance: assess whether the entity extracted is important for the article (checked) or not (unchecked). This box is also used to counter balance and not penalize the tools that extract more than only named entities such as general topic that can well index the article being analyzed while not being considered as a named entity in NERD. The relevance of the NEs is later used to assign different weights in computing precision and recall.

Figure 1 shows two different evaluations performed by AlchemyAPI and Extractiv. Some NEs are comparable in terms of the taxonomy of types of named entity they can recognize while having some differences: e.g., AlchemyAPI reports an accurate detection of the general Person type and disambiguates the NE with a general resource URI while Extractiv provides a fine-grained taxonomy of this type according to the person profession and disambiguates the NE with a LOD URI. After reviewing all the rows in the table, the user saves the evalua-

(a) AlchemyAPI (b) Extractiv

**Fig. 1.** NE extraction results performed by AlchemyAPI and Extractiv given the same input news article. Fields in green have been rated positively while fields in red have been rated negatively.

tion results by clicking the "Save" button and completes the evaluation process. When neither "+" nor "-" have been clicked for some cells, the user is alerted with a pop up notice that the missing evaluations will be considered negatively and is prompted to confirm or not this choice before leaving the evaluation page. Comparison charts are then presented in the comparison page.

## 4 Conclusion

We presented a framework that compares popular Linked Data Named Entities extractors. The service is oriented towards people wishing to build gold standard annotations for comparing NLP services. We will invite ISWC guests to try the service with a set of pre-annotated articles in order to create and improve a gold standard and discuss the complexity of such an evaluation task.

## Acknowledgments

This paper was supported by the French Ministry of Industry (*Innovative Web* call) under contract 09.2.93.0966, "Collaborative Annotation for Video Accessibility" (ACAV). The authors would also like to thank Xueyi Yang and Mintao Ye from EURECOM and Thomas Steiner from Google for fruitful discussions on the design and development of the NERD interfaces.