NERD workshop

Luca Foppiano @ ALMAnaCH - Inria Paris

Berlin, 18/09/2017

The HIRMEOS project has received funding from European Union's Horizon 2020 research and innovation programme under grant agreement 731102

Agenda

- Introducing the (N)ERD service
- NERD REST API
- Usages and use cases

Entities

Rigid textual expressions corresponding to certain pre-defined categories

Taking advantage of semi-normalized, nomenclature-based expressions

Examples:

- **person**: individual, function
- **locations**: country, place, etc.
- **time**: dates, time range, period, ...
- **organization**: institution
- **events**: military, political, …

but also measures, substance, biotech entities, websites, etc.

Example

Identification

Unfortunately for the **Allies**, the pro-**German King Constantine I** dismissed the pro-**Allied** government of **E. Venizelos** before the **Allied** expeditionary force could arrive.

Categorization

Normalization

King	Constantine	I
title	forename	suffix

E	Venizelos
forename	last name

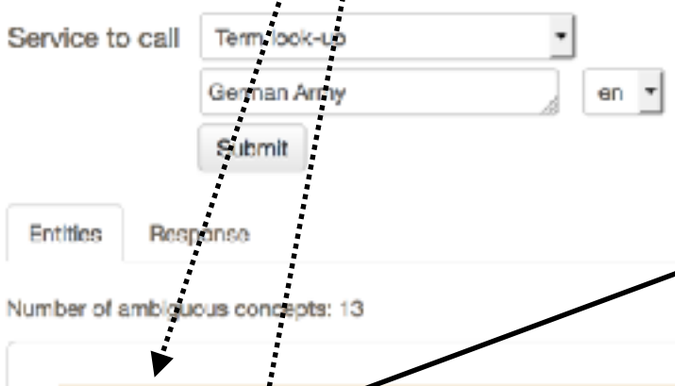
Resolution

King Constantine 1 —> Wikidata: Q152099, Wikipedia: 160204

E. Venizelos: Eleftherios Venizelos: Wikidata: Q205545, Wikipedia: 305256

Example (2)

*[..] After marching through Belgium, Luxembourg and the Ardennes, the **German Army** advanced, in the latter half of August, into northern France where they met both the French army, under Joseph Joffre, and the initial six divisions of the British Expeditionary Force, under Sir John French. [..]*



entity-fishing!

- NERD is a standard name
- this specific (N)ERD has a name: entity-fishing
- … as we fish entities from the large ocean of knowledge within wikipedia/Wikidata and more…
- but don’t worry, in this presentation I will keep calling it (N)ERD

(N)ERD service

Full entity processing: the service is not limited to the recognition of entity mentions but also covers the **disambiguation and the resolution** of entities against standard knowledge bases (Wikipedia, Wikidata).

Independence from a particular framework and usage scenario for maximum reuse

State-of-the-art: we target state-of-the-art performances in term of accuracy, coverage (entity variety), speed and exploitation of the context.

REST API service: the service is completely decoupled from the implementation details, thus independent from a particular platform, infrastructure and architecture, easier to integrate, to scale and to monitor.

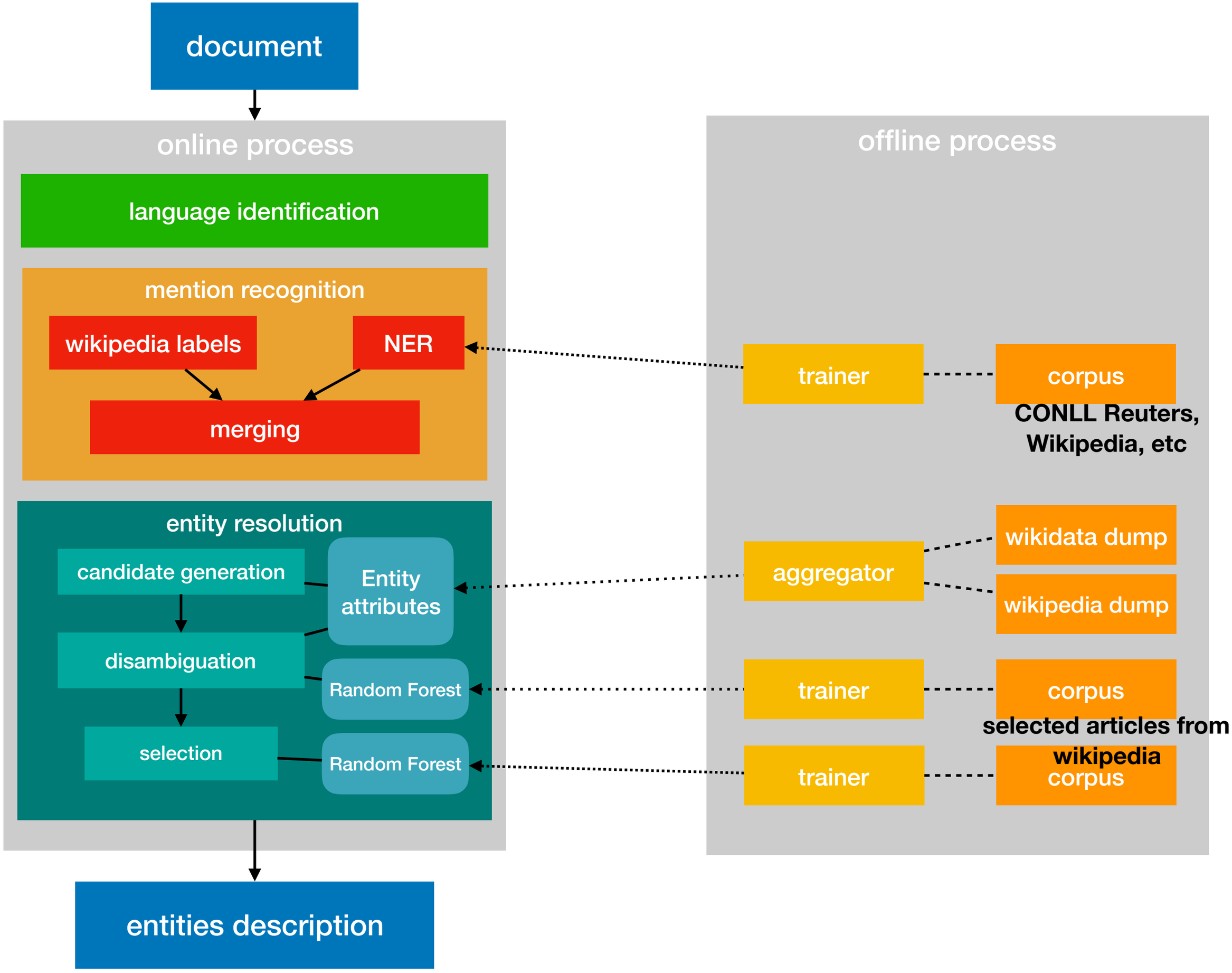
(N)ERD service

No requirement for expertise in knowledge engineering. Knowledge workers are not knowledge engineers and we cannot expect a good adhesion to a tool which would require sophisticated preliminary human effort (rules, lexicon, ontologies, etc.).

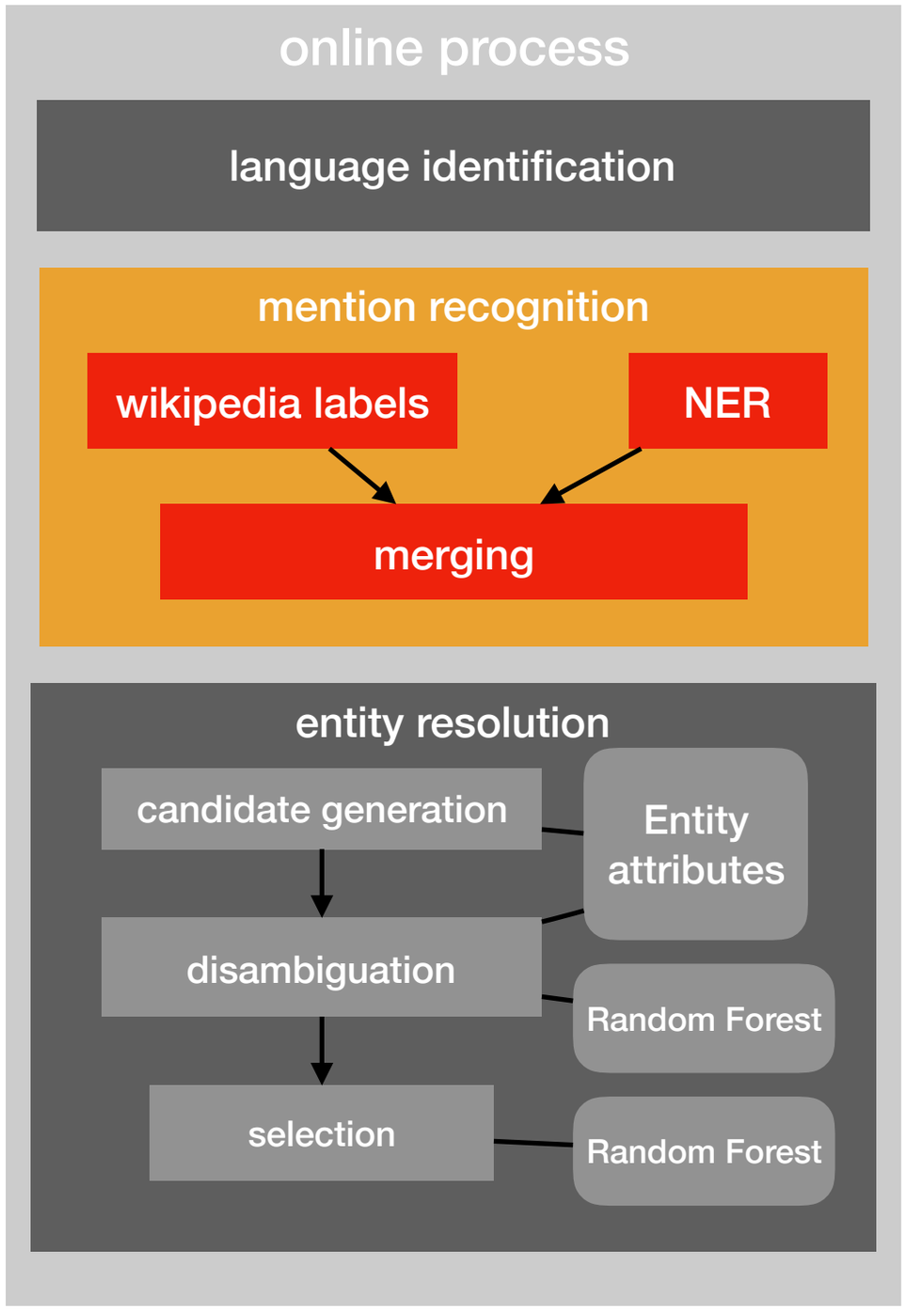
The default usage mode of the service is thus automatic, relying on **existing large scale knowledge bases** and the ability to identified and normalise **entities never seen before**.

Customisable: Offering to the user the possibility to customise **easily** the generic NERD to **different** historical topics without the need of knowledge engineering skills and the dependence to a particular framework (e.g. rules, formal ontology/logics, semantic web, etc.)

Multilingual support: designed for several languages, exploiting the well implemented translation in Wikidata


document
online process
language identification
mention recognition
wikipedia labels
NER
merging
entity resolution
candidate generation
Entity attributes
disambiguation
Random Forest
selection
Random Forest
entities description
offline process
trainer
corpus
CONLL Reuters, Wikipedia, etc
aggregator
wikidata dump
wikipedia dump
trainer
corpus
selected articles from wikipedia
trainer
corpus

Mention recognition



two steps:

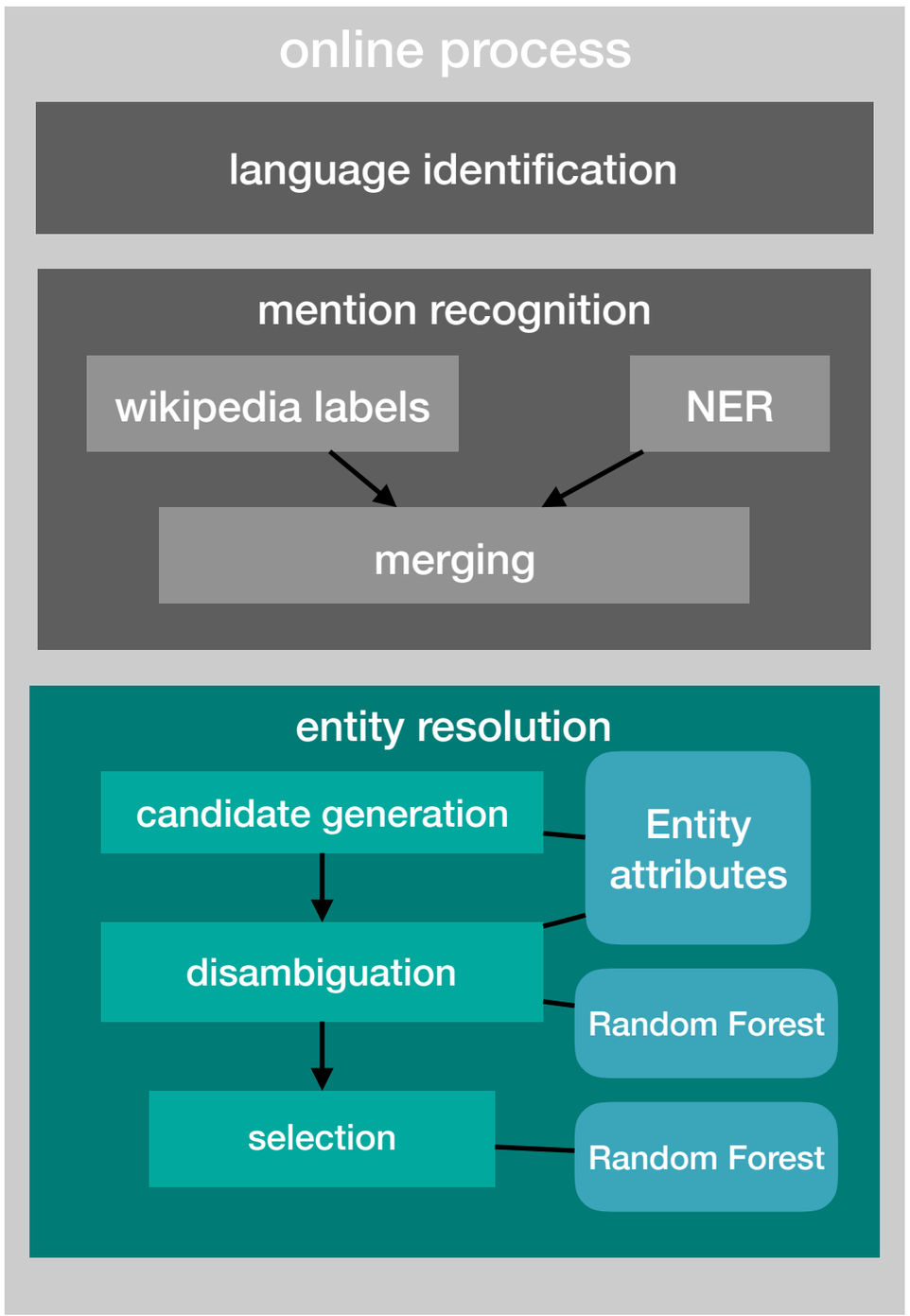
Mention class tagging

- Identification of each mention and tag using wikipedia labels
- Use another open source tool (Grobid-NER) based on CRF machine learning model - total 27 classes as of today, supporting FR and EN
- Additional domain specific Named entities recognition can be applied (Chemistry, biology, etc.)

Merging

- merging wikipedia label identified mentions and recognised Named Entities

Entity resolution



Knowledge bases: Wikipedia (4 M entities) and Wikidata (27 M entities)

Three steps:

Candidate generation: the selection in the knowledge base of the entity candidates for an entity mention identified previously in the text. We exploit large standard datasets to create mention/entity statistical profiles.

Disambiguation: the association of a score to each candidate entity estimating how well the entity describes the mention in the **context of the input text.** Considering the other entities appearing at the same moment.

Selection: decision to select or not an entity for a given mention

Entity resolution

Approach using Machine Learning

The statistical model is Random Forest and benefit from the several fine features generated by the rich information contained in the knowledge base

The training data are obtained from the knowledge base itself as each mention in wikipedia is already resolved to the correct real entity (e.g. the mention Washington is pointing already to the correct page)

Demo

REST API

REST API which is decoupled from the implementation

Currently input and output is driven by a Query Language in JSON format


```
"text": "The text to be processed.",
"shortText": "term1 term2 ...",
"termVector": [
    {
        "term": "term1",
        "score": 0.3
    },
    {
        "term": "term2",
        "score": 0.1
    }
],
"language": {
    "lang": "en"
},
"entities": [],
"resultLanguages": [
    "fr",
    "de"
],
"onlyNER": false,
"nbest": 0,
"sentence": false,
"customisation": "generic",
"processSentence": []
```

REST API

Text: disambiguation of the paragraph of plain text (as provided in the query) or PDF (provided separately)

short text: corresponding to several search terms used together and which can possibly be disambiguated when associated

weighted terms: each term will be disambiguated, when possible, in the context of the complete vector


```
"text": "The text to be processed.",
"shortText": "term1 term2 ...",
"termVector": [
    {
        "term": "term1",
        "score": 0.3
    },
    {
        "term": "term2",
        "score": 0.1
    }
],
"language": {
    "lang": "en"
},
"entities": [],
"resultLanguages": [
    "fr",
    "de"
],
"onlyNER": false,
"nbest": 0,
"sentence": false,
"customisation": "generic",
"processSentence": []
```


Input

Options and parameters

REST API

language: provided language by the client, if none provided the language will be recognised on the fly

onlyNER: return only the recognised named entities without disambiguate them

nbest: return the n disambiguated results (by default only the first one)

REST API

entities: if entities in the text are already known, they can be provided in the query

sentence: if used, it's segmenting the text using an internal segmenter

sentences: return the n disambiguated results (by default only the first one)


```
"text": "The text to be processed.",
"shortText": "term1 term2 ...",
"termVector": [
    {
        "term": "term1",
        "score": 0.3
    },
    {
        "term": "term2",
        "score": 0.1
    }
],
"language": {
    "lang": "en"
},
"entities": [],
"resultLanguages": [
    "fr",
    "de"
],
"onlyNER": false,
"nbest": 0,
"sentence": false,
"customisation": "generic",
"processSentence": []
```

REST API text processing

Short text (< 5 words), use the shortText option.

Long text (> 500 word): the service is designed, trained and scaled for paragraphs:

- process the PDF version of the text (if possible)

- process the text “by paragraphs” providing the disambiguated entities from the previous paragraph at each query (see later)

- process XML-TEI version of the text (soon available, the text will be handled server-side)

REST API

processSentence: trigger the NERD only on the selected sentence

sentences: provides the segmentation boundaries

entities: if entities in the text are already known, they can be provided in the query


```
{
    "text": "The army, led by general Paul von
        Hindenburg defeated Russia in a series of
        battles collectively known as the First Battle
        of Tannenberg. But the failed Russian invasion,
        causing the fresh German troops to move to the
        east, allowed the tactical Allied victory at
        the First Battle of the Marne.",
    "processSentence": [
        1
    ],
    "sentences": [
        {
            "offsetStart": 0,
            "offsetEnd": 138
        },
        {
            "offsetStart": 138,
            "offsetEnd": 293
        }
    ],
    "entities": [
        {
            "rawName": "Russian",
            "type": "NATIONAL",
            "offsetStart": 153,
            "offsetEnd": 160
        }
    ]
}
```

Process long text

We assume the client should know better their own text and how to segment it.

We also assume the client doesn’t have ‘natural paragraphs’:

- call the segmenter with the long text
- process the first n sentences (10 < n < 20)
- process the following n sentences by providing the previously recognised entities
- … and so on and so forth

Customisation API

- The customisation is a way to specialise the entity recognition, disambiguation and resolution for a particular domain.

- The context will be build based on Wikipedia articles and raw texts, which are all optional.

Example: WW2 customisation

In the context of WW2, term like German Army or Vichy have different related wikipedia articles.

wikipedia entity corresponding to the page Vichy France

```
{
  "wikipedia": [
    21376046,
    20599016
  ],
  "texts": [
    "The German Army entered Paris. They fought against
        Vichy."
  ],
  "description": "Customisation for World War 2 domain"
}
```

wikipedia entity corresponding to the page German army - Wehrmacht

Each customisation is saved using a name, which can be selected in the query. Propriety ‘customisation’.

Demo

Usage and use cases

- The (N)ERD, initially based on Wikipedia and Freebase (discontinued in favour of Wikidata) is now moving toward Wikidata

- Better, cleaner and more structured data than DBPedia

- Once obtained the disambiguated entities, the Wikidata attributes contains a huge amount of information that can be exploited

Wikidata

- Generic knowledge base: 35M entities, 154M statements
- Licence CC0 (freely usable, modifiable, redistributable, etc)
- Knowledge base collaborative human+machines (88% of the editing are automatic)
- Wikidata has recently reached the 561,378,122 edits operation (18000 distinct human have collaborated, **only 6 staff members**, it's the largest collaborative DB in the world by far)

Wikidata model

- The Wikidata data model provides a meta-model or ontology for describing real world entities. Such descriptions are concrete models for real world entities.
- Wikidata’s motto: **”verifiability, not truth”**: statements are supported by references. Contradictions can be verified.

  “*We do not say that Berlin has a population of 3,5 M, we say that there is this statement about Berlin's population being 3,5 M as of 2011 according to the German statistical office.”*

label

Douglas Adams (Q42)

item identifier

description

English writer and humorist

aliases

Douglas Noël Adams | Douglas Noel Adams

► In more languages

Statements

property

educated at

value

St John's College

qualifiers

end time	1974
academic major	English literature
academic degree	Bachelor of Arts
start time	1971

rank

▼ 2 references

opened references

stated in	Encyclopædia Britannica Online
reference URL	http://www.nndb.com/people/731/000023662/
original language of work	English
retrieved	7 December 2013
publisher	NNDB
title	Douglas Adams (English)

– add reference

statement group

Brentwood School

end time	1970
start time	1959

collapsed reference

► 0 references

– add (statement)

Wikidata model

Wikidata is becoming progressively a scientific knowledge base of reference (genetic, chemistry, ecology, animal species, etc)

e.g. "*Thanks to the amazing work of the Wikidata community, every human gene (according to the United States National Center for Biotechnology Information) now has a representative entity on Wikidata.*"
(Ref: *https://blog.wikimedia.de/2014/10/22/establishing-wikidata-as-the-central-hub-for-linked-open-life-science-data/)*

Example:

- https://www.wikidata.org/wiki/Q5090 (Rice)
- https://www.wikidata.org/wiki/Q405 (Moon)

Demo

Questions?

Thank you

The HIRMEOS project has received funding from European Union's Horizon 2020 research and innovation programme under grant agreement 731102