

The TALP participation at TAC-KBP 2013

A. Ageno, P.R. Comas, A. Naderi, H. Rodríguez, J. Turmo

TALP Research Center, UPC, Spain.

{ageno, pcomas, anaderi, horacio, turmo}@lsi.upc.edu

Abstract

This document describes the work performed by the Universitat Politècnica de Catalunya (UPC) in its second participation at TAC-KBP 2013 in both the Entity Linking and the Slot Filling tasks.

1 Introduction

Both Entity Linking (EL) and Slot Filling (SF) tasks aim at extracting useful information in order to enrich a knowledge base. This document describes the work carried out by the TALP research group of UPC in its second participation at TAC-KBP 2013 in both the Entity Linking and the Slot Filling tasks. Following our 2012 participation, we have assessed the obtained results and partially modified our approaches in order to try and improve our performance. In the case of EL, this year's approach is completely different. Nevertheless, the basis underlying the SF approaches remain the same as those of 2012.

EL is the task of referring a Named Entity mention to the unique entry within a reference knowledge base (KB). TAC-KBP track defines the task of EL as follows: having a set of queries, each one consisting of a target name string along with a background document in which the target name string can be found and a source document collection from which systems can learn, the EL system is required to select the relevant KB entry. Queries generally consist of the same name string from different docs. The system is expected to distinguish the ambiguous names (e.g., Barcelona could refer to the

sport team, the university, city, state, or person). In TAC-KBP 2013, we have participated with an approach completely different to the one used for the 2012 edition (c.f. Section 3). We have sent one run and evaluated our EL system just for Mono-lingual Entity Linking. The run did not access the web and also did not use query offsets during the evaluation.

In the SF task, the given set of queries is a set of entity KB nodes that must be augmented by extracting all the new learnable slot values for the entity as found in a large corpus of documents. SF involves mining information from the documents and therefore applies Information Extraction (IE) techniques. We have only participated in the English Mono-lingual Slot Filling task, submitting three runs. The first two runs differ from the latter in the IE approach employed to detect possible query slot fillers in the candidate documents. The approach used for the first two runs is completely unsupervised (based on minority clustering), whereas the approach used in the last run is supervised (based on distant learning).

The rest of the document is structured as follows. Section 2 describes the query preprocessing step, shared by all the systems. Section 3 is devoted to our Entity Linking approach and its results in KBP 2013. Finally, in section 4 we describe our Slot Filling approaches, including the shared document preprocessing step, the two different IE approaches applied, the shared integration phase, and the analysis of the results obtained by both of them.

2 Query preprocessing

A query preprocessing is carried out for both SF and EL with minor differences. We follow this year the

same approach than in our 2012 participation with some improvements for facing problems detected last year. In what follows we focus on the changes made, including some descriptions for the sake of readability but omitting details that can be found in the description of 2012 system, (Gonzalez et al., 2012).

Query preprocessing consists of the following tasks:

- For EL, classifying the query entity into the appropriate query type: Person (PER), Organization (ORG) or GeoPolitical (GPE).
- For SF, obtaining, when existing, the corresponding node in KB.
- For SF, looking up at WP for the possible existence of the corresponding page. Disambiguation pages are discarded.
- For both SF and EL, Generating the set of alternate names for the query entity. We describe next with some detail the last task.

A crucial point for both SF and EL tasks is to generate an accurate set of alternate names, also named variation lists, to be used in both the IR step and the extraction step in order to improve the recall. We call A this set of alternate names. For generating A we use the query name, its type and, when available, textual information coming from the background document, the KB entry and the WP page.

The query name can be a single word or a multiword. In SF, most of the queries are precise (first name and family name for PER, full name for ORG), while EL queries are much less precise (an acronym for ORG, just a single word for PER).

For query classification and alternate names generation it is important to locate accurately all the mentions of entities, specially the query, occurring in the reference document. For doing so we have used another NERC system and modified the way of selecting, for EL, the appropriate query type, as described in next section.

The structured information we use is the following:

- For SF, when a KBP node is included in the query, the facts associated to this node are selected.

- For both SF and EL we look at WP. When an unambiguous page is found, we select the included infoboxes, if any. In the case of reaching a disambiguation page, as trying to disambiguate the query using the short caption texts usually attached to the different options resulted in a degradation of the performance, we simply decided discarding the use of WP in this case.
- For EL, if the type is GPE we look at geographic gazetteers (GNIS¹ and GEONAMES²) and select the entries. In this case we decided to use all the variants included in the gazetteer's entry with no attempt to disambiguate the toponym. disappointingly this decision resulted in the introduction of a huge amount of noise.

The documents we use as knowledge sources (KS) are:

- The reference (background) document attached to the query.
- For SF, when a KBP node is included in the query, the attached description document and the facts associated to this node when containing free text.
- For both SF and EL we look at WP. When an unambiguous page is found, we select the textual content of the page.

Using all these KS our way of building A is the following: A is a set of pairs (*alternate_name*, *score*). Score ranges from 0 (minimum confidence) to 1 (maximum confidence). This set is initialized with the query name scored with 1. Then a set of enrichment procedures are iteratively applied until no more alternate names are found. There are two types of procedures: generic and type-specific. Generic procedures are the following:

1. We select a set of pairs (*WP infobox*, *slot*) where slot refers to an alternate name (e.g. *formal name*, *alias*, *nickname*, *also_known_as*, etc.). If we have a WP page we extract these

¹<http://geonames.usgs.gov/geonames/stategaz>

²<http://www.geonames.org>

| Query | Name | Alternate names | # |
|-------|----------------------|--|----|
| SF558 | Barbara Boxer | 1.0 Barbara Levy Boxer 0.8 Barbara L. Boxer 0.64 B. L. Boxer 0.56 B. Boxer 0.49 Boxer ... | 37 |
| SF520 | Hong Kong Disneyland | 1.0 Hong Kong Disneyland 1.0 HKDL 0.8 H. Kong Disneyland 0.8 Hong K. Disneyland 0.7 Hong Disneyland 0.7 Kong Disneyland 0.64 Disneyland ... | 30 |

Table 1: Examples of alternate names

values and insert them into A also with the maximum score.

2. We proceed in the same way with the KBP nodes.
3. We apply the SF corresponding to the generic slot *alternate_name* existing for both PER and ORG.

Specific procedures, applied iteratively over all the current members of A :

1. For PER. We use a DCG grammar of English person names for trying to extract the structure of a complex name. For instance, from *Paul Auster* our aim is to detect that the first name is *Paul* and the family (main) name is *Auster*. We then generate valid variants of the original name always preserving the family name. These variants are scored accordingly with the generalization degree, in our example: (*P. Auster*, 0.8), (*Auster*, 0.6). Compared to our 2012 system we have enriched the gazetteer of first names including Spanish names and very frequent names in other languages. We have included, too, diminutives³ (e.g. Robert → Bob, Bobby).
2. For ORG. We have developed a set of 12 acronym/expansion weighted mapping functions:

- Starting from an acronym we look up in the textual KS for the occurrence of valid expansions applying our mapping functions. We score the valid variants with the weight of the applied function.
- Starting from a complete form we perform acronym detection equally scored.
- New forms of ORG names can be found removing common company suffixes (e.g. *Inc.*, *Company*, etc.).

3. For GPE we extract all the variants existing in the geographic gazetteers and score them with the edit distance between the original form and the variant.

Some examples of alternate names generated with this procedure are shown in Table 1.

3 Entity Linking

Our approach follows the typical architecture in the state of the art (Figure 1). Briefly, given a query, consisting of an entity name and a background document, we start by expanding and enriching the background document (query expansion and enrichment step). Then, we select those KB nodes which are candidate to be the correct entity for the query (candidate generation step). Finally, we rank KB candidates and select the candidate with the highest rank, and all queries belonging to the same Not-In-KB entity are clustered together assigning a same NIL id (candidate Ranking and NIL clustering step).

Details of each step are provided next.

3.1 Query Expansion and Enrichment

Expanding the query from its context can effectively reduce the ambiguities of the mention, under the assumption that two name variants in the same document refer to the same entity. For example, “Roth” in Wikipedia refers to seventy-six entries, but its expansion “John D. Roth” only refers to two entries. This step also includes enriching the background document integrating information retrieved from knowledge resources. For expanding queries, the following steps are:

Query Classification. Queries are classified into 3 entity types: PER (e.g., “George Washington”),

³Obtained from <http://www.allwords.com>

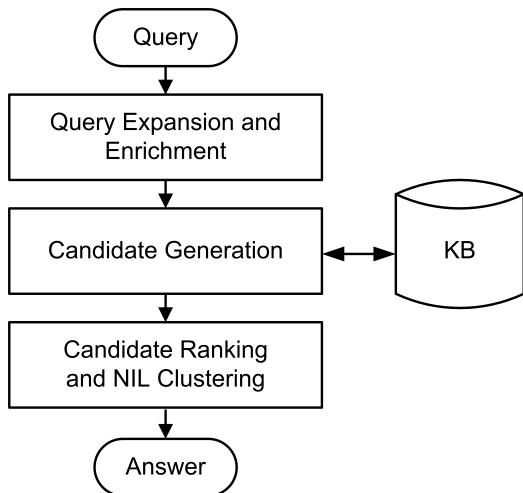


Figure 1: General architecture of the EL systems

ORG (e.g., “Microsoft”), GPE (GeoPolitical Entity, e.g., “Heidelberg city”). We have used Illinois Named Entity Recognizer and Classifier (NERC) (Ratinov and Roth, 2009)⁴ for this task. In our system, NERC is used for classifying all entity mentions in the background document. Thus, considering all mentions with their type, we select those ones related to the query name. Afterwards, we choose the longest mention (e.g., selecting “George W. Bush” rather than “George Bush” for the query name “Bush”), and assign its type as query type.

Background Document Enrichment. As VSM components are extracted from the background document of each query, we need as most disambiguated entities as possible. For doing so, AIDA system (Hoffart et al., 2011)⁵ is applied. AIDA is a framework for entity detection and disambiguation. Given a natural-language text or a Web table, it maps mentions of ambiguous names onto canonical entities (e.g., individual people or places) registered in YAGO2 (Hoffart et al., 2013)⁶ a huge semantic KB derived from WP, WordNet (Fellbaum, 1998)⁷ and Geonames,⁸ and containing more than 10 million entities (like persons, organizations, cities, etc.) and more than 120 million facts about these enti-

⁴cogcomp.cs.illinois.edu

⁵www.mpi-inf.mpg.de/yago-naga/aida

⁶www.mpi-inf.mpg.de/yago-naga/yago

⁷wordnet.princeton.edu

⁸www.geonames.org

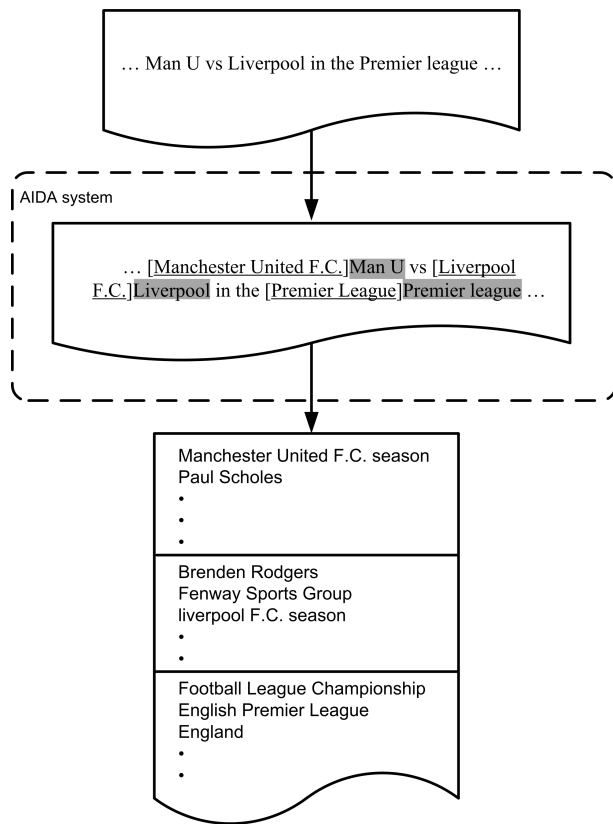


Figure 2: A sample of generating keyphrases by AIDA system

ties. Each entity in YAGO2 contains some kind of information, including weighted keyphrases. A keyphrase is contextual information extracted from link anchor, in-link, title and WP category sources of the corresponding entity page that can be used for entity disambiguation. We use AIDA just to extract keyphrases from the entities in the background document. The number of keyphrases of each entity is high, thus, a threshold (set to 0.002) is used for filtering out the less reliable and getting a smaller and more focused set of keyphrases. In general, our system extracts ~300 keyphrases for each mention of the background document. Figure 2 shows an example for producing related keyphrases of background document mentions “Man U,” “Liverpool,” and “Premier league” using AIDA.

Alternate Name Generation. In this step, a set of Alternate Names (ANs) of each query is generated from the content of its corresponding background document. For instance, in Figure 3, the sys-

```

<DOCID> eng-NG-31-108519-8977045 </DOCID>
<DOCTYPE SOURCE="usenet"> USENET TEXT </
DOCTYPE>
<DATETIME> 2007-10-10T22:25:00 </DATETIME>
<BODY>
<HEADLINE>
Dollars for Death
</HEADLINE>
<TEXT>
<POST>
<POSTER> Anybody &lt;anybod...@canada.com&gt; </
POSTER>
<POSTDATE> 2007-10-10T22:25:00 </POSTDATE>
Dieticians

The American Dietetic Association (ADA) has 67,000
members. Their motto is "Everything in
moderation." That includes McDonald's, other fast food
restaurants, dairy products, NutraPoison, and sugar-rich soda.
Of course, the one concept that they do not limit is donations
by various industry groups who delight in seeing the ADA's
continuing ..... i4crob(at)earthlink.net
</POST></TEXT></BODY></DOC>

```

Figure 3: Example background document for the query name “ADA” from the TAC 2013 data set

tem used Acronym expansion for extracting “American Dietetic Association” from “ADA.” We also applied auxiliary gazetteers such as the US states, (e.g., the pair $\langle CA, California \rangle$ or $\langle MD, Maryland \rangle$), and country abbreviations, (e.g., the pairs $\langle UK, United Kingdom \rangle$, $\langle US, United States \rangle$ or $\langle UAE, United Arab Emirates \rangle$). Thus, a set of potential candidates is generated from each AN of each query, as described next.

3.2 Candidate Generation

Given a particular query, q , a set of candidates, C , is found by retrieving those entries from the KB whose names are similar enough, using Dice coefficient, to one of the alternate names of q found with the query expansion. In our experiments we used a similarity threshold of 0.95, 0.85 or 1 for PER, ORG and GPE respectively. By comparing the candidate entity type extracted from corresponding KB page and query type obtained by NERC, we filter out those candidate having different types to attain more discriminative candidates.

In general, KB entity pages contain facts and an informative context about the entity. We enrich the

```

<entity wiki_title="Parker, Florida" type="GPE"
id="E0000012" name="Parker, Florida">

<facts class="Infobox Settlement">
<fact name="official_name">
Parker, Florida
</fact>

<fact name="subdivision_name">
<link entity_id="E0679687">United States</link>
</fact>

<fact name="subdivision_name1">
<link entity_id="E0373950">Florida</link>
</fact>
.....
</facts>

<wiki_text>
<![CDATA[Parker, Florida
Parker is a city in Bay County, Florida, United
States... ..... 4.6% of those age 65 or over.
]]>
</wiki_text>

</entity>

```

Figure 4: A sample KB candidate entity page containing a set of facts and its informative context

context information of each KB candidate entity by searching the corresponding facts as separate entities in the reference KB and then merging their related informative contexts with the current one. By applying this technique, the context of each candidate could be more discriminative and informative. Figure 4 shows a sample KB entity page corresponding to entity name “Parker, Florida.” The system collects the `wiki_text` information of its related entities “United States” and “Florida” to enrich the `wiki_text` of “Parker, Florida.”

3.3 Candidate Ranking and NIL Clustering

In EL, query expansion techniques are alike across systems, and KB node candidate generation methods normally achieve more than 95% recall. Therefore, the most crucial step is ranking the KB candidates and selecting the best node. This module sorts the retrieved candidates according to the likelihood of being the correct referent. Our approach is inspired by (Cucerzan, 2007). He employed a Vector Space Model (VSM), in which a vectorial representation of the processed background document is compared with the vectorial representation of the reference KB

candidates. In our case the vector space domain consist of the whole set of word within the keyphrases found in the enriched background document and the rank consists of their tf/idf computed against the set of candidate documents. We use cosine similarity. In addition, in order to reduce dimensionality we apply LSI.

We also apply a term clustering method to cluster NILs. For this purpose, each NIL query is searched in a generated NIL clustering list. The query name along with its name variations are added to the NIL clustering list by taking a new NIL ID if it does not exist in the list. Otherwise, the new NIL query takes the ID of found NIL query. A similarity threshold for searching is defined using Dice coefficient algorithm.

4 Slot Filling task

The UPC system used for Slot Filling is similar to the one used for the 2012 edition. It consists in three steps: 1) preprocessing the document collection in order to filter those documents relevant for each query, 2) applying Information Extraction (IE) patterns to the relevant documents to achieve possible fillers for the slots required for each query, and 3) integrating the resulting slot fillers into the KBP knowledge base by normalising extracted fillers.

As for the 2012 edition, two different IE pattern learning approaches have been performed for our participation in KBP 2013: the first approach based on distant learning and the second one based on unsupervised learning. The rest of this section describes the preprocessing of the document collection as well as both learning approaches and the integration process.

4.1 Document preprocessing

Prior to evaluation of KBP 2013, the document collection was indexed using the Lucene Information Retrieval engine⁹ using all the words occurring in the documents. During the evaluation, a set D of documents is retrieved for each query. This set consists of the top 300 documents retrieved from those documents containing at least one alternate name of the query expanded as described in Section 2 for the

⁹<http://lucene.apache.org/>

| | |
|---------------|--------------|
| full_name | birth name |
| othername | subject_name |
| burthname | alias |
| othername(s) | birth_name |
| native_name | othername(s) |
| birth_name | nickname |
| aliases | full name |
| other_names | name |
| birthname | nicknames |
| othernames | alias |
| also known as | fullname |
| nickname | stage/screen |
| full name | name |
| pseudonym | other names |
| name | realname |
| playername | names |

Table 2: Specific slots for the generic slot *per:alternate_names*

SF task. The document ranking used is Lucene’s default.

4.2 Distant-Learning Approach

Our third run in SF task of KBP 2013 follows the distant learning (DL) paradigm for Relation Extraction (RE). DL was initially proposed as a RE approach by (Mintz et al., 2009) and applied to SF task in preceeding KBP contests by several groups such as (Agirre et al., 2009; Surdeanu et al., 2010; Garrido et al., 2011). DL uses supervised learning but the supervision is not provided by manual annotation but from the occurrence of positive training instances in a KS or reference corpus. In the first proposal, (Mintz et al., 2009) used Freebase, an on-line database of structured semantic data, as KB. In subsequent applications, Wikipedia (WP) infoboxes have been preferred due to its better precision, at a cost of a drop in recall. In our case we have chosen WP too. Our distant learning approach to the task is basically the same we followed in our 2012 participation (Run1) and consists on the following steps:

1. From a local copy of the English WP,¹⁰ we try

¹⁰http://en.wikipedia.org/wiki/English_Wikipedia. We use for this purpose the JWPK software by Irina Gurevich: <http://www.ukp.tu-darmstadt.de/software/jwpl>


| | |
|---|--|
| <p>Antoni Gaudí</p> <p>From Wikipedia, the free encyclopedia</p> <p>"Gaudí" redirects here. For other uses, see Gaudí (disambiguation). This is a Catalan name. The first family name is Gaudí and the second is Cornet.</p> <p>Antoni Gaudí i Cornet (Catalan pronunciation: [ənˈtoni ɣəwˈði]; 25 June 1852–10 June 1926) was a Spanish Catalan architect and the best-known representative of Catalan Modernism. Gaudí's works are marked by a highly individual style and the vast majority of them are situated in the Catalan capital of Barcelona, including his magnum opus, the Sagrada Família.</p> | <p>Antoni Gaudí</p>  <p>Antoni Gaudí by Pau Audouard</p> <p>Born 25 June 1852 Reus, Catalonia, Spain^{[1][2]}</p> <p>Died 10 June 1926 (aged 73) Barcelona, Catalonia, Spain</p> <p>Work</p> <p>Buildings Sagrada Família, Casa Milà, Casa Batlló</p> <p>Projects Park Güell, Colònia Güell</p> |
|---|--|

Figure 5: Example of WP page

| Occured Value | Extracted Value |
|----------------------------|------------------|
| [October 16] , [1952] | October 16, 1952 |
| [March 7] [322 BC] | March 7 322 BC |
| [748]([Arabian Peninsula]) | 748 |
| [1368] or [1377] | ? |
| [406 AH] (1015 AD) | 1015 AD |
| 25 June , 1274 | 25 June , 1274 |
| (still alive in 1974) | ? |
| alive | ? |
| ‘circa’ 1126 | 1126 circa |
| [1663] (age 23) | 1663 |

Table 3: Examples of values for the generic slot *per:date_of_death*

to automatically locate the set of pages corresponding to PER and the corresponding to ORG. For doing so we used the links between WP pages and WP categories as well as the graph structure of WP categories. Let *Pages_{PER}* and *Pages_{ORG}* be these sets. For *PER* this process is straightforward because there is a category *People* in *WP* and navigating top-down the *WP category graph* from this category (*top category*) we can collect the set of categories that likely correspond to people. From this set, following the category-page links the set of pages corresponding to people can be easily collected. In this way we collected 20,741 Cat-

egories and 418,352 source pages. For *ORG* the process is not so easy. First, although *Organizations* exists in *WP*, its coverage is rather reduced and other categories should be added to the *top category* set (as *Legal entities* or *Business*). The confidence on the set of categories derived from this *top category* set and on the set of pages obtained from it is not so high, although the figures are comparable (17,646 categories and 198,946 pages). We favour recall over precision in this task because false positives are not specially harmful as they do not contain the relevant infoboxes.

2. We used the mapping between the generic slots and the specific slots occurring in WP infoboxes provided by KBP organization. Table 2 shows, as an example, the set of specific slots corresponding to the generic slot *per:alternate_names*. As shown in Figure 5, WP pages can include both structured (infoboxes, itemized lists, ...) and unstructured material (text). We took profit of page infoboxes and page textual content. For all the pages in both *Pages_{PER}* and *Pages_{ORG}* we collected all the occurring infoboxes, slots and values. This process reduced drastically the available *WP* pages (for *PER* only 142,452 pages (34%) contained infoboxes and only 36,958 (8.8%) with target infobox attributes, for *ORG* the reduction is more heavy). This process resulted in a set of tuples: $\langle page\ name, generic\ slot, infobox\ name, specific\ slot, slot\ value \rangle$. Let *PagesSlotsValues_{PER}* and *PagesSlotsValues_{ORG}* be these sets. Extracting the values of an specific slot is in some cases easy (e.g. for single-valued slots with a precise type, as *per:date_of_birth*) but in many others it is difficult. In Table 3 some examples of values for the generic slot *per:date_of_death* are shown. Using Alergia system, (Carrasco and Oncina, 1994), we learned regular grammars of the slots' values for allowing their extraction. In fact, the number of learned grammars is smaller than the number of slots because some of the values are of the same type, for example the DATE grammar can be used for the slots *date_of_birth* and *date_of_death*.

3. For each of the tuples in *PagesSlotsValues-PER* and *PagesSlotsValues-ORG* we extracted the patterns occurring in the text corresponding to the page. For doing so we obtained the possible variants of the page name, for instance if *page_name = Paul Auster*, also *P. Auster* and *Auster* are considered variants of the name. Acronym expansion or extraction is performed in the case of *ORG*. For doing this task we used the same processors described in section 2) for generating *A*. A similar process is carried out for the slot value, so we have functions for generating alternate names for people, for cities, for organizations, for dates, and so on. For instance for the slot *per:date_of_birth* if the value is *27 April 1945*, also *27-04-1945*, *April 1945*, and *1945* are considered as valid variants (the same grammars used for extraction are used here for generation). Two sets of alternate names, *alternateNamesX*, for variants of the query name, and *alternateNamesY*, for the variants of the slot value, were obtained. Then We looked on the text for all the occurrences of *alternateNamesX* (X_0, \dots, X_n) and of *alternateNamesY* (Y_0, \dots, Y_m). For each pair of occurrences (X_i, Y_j) (in this order) we collected the sequence of words occurring between them and we grouped together all the patterns corresponding to each generic slot. We built in this way the bag *PatternsGenericSlot*. This process resulted in collecting 9,064 patterns for *ORG* (ranging from 70 for *org:city_of_headquarters*, up to 2,573 for *org:political_religious_affiliation*) and 6,982 patterns for *PER* (from 23 for *per:cause_of_death* to 588 for *per:title*) with very variable accuracy. In Table 6 some examples of the 57 patterns for the generic slot *per:date_of_birth* are shown. For the sake of illustration we include next the figures corresponding to this slot 355 patterns were collected with a global precision of 0.95 and recall of 0.96. We found 25,624 occurrences of these patterns. The top scored pattern was *X born Y* with 19,447 occurrences (0.95 precision and 0.70 recall). The top 5 patterns occur 23,341 times with 0.96 precision and 0.90 recall.

Once the set of patterns for each generic slot was built (only the most frequent patterns are selected) the process of extraction can be performed as shown in the following steps.

1. For each query we expanded the name onto *alternateNamesX*.
2. We looked into Lucene indexes for the occurrences of documents containing any of the variants in *alternateNamesX*. Some filtering processes were performed in order to maintaining this set in a manageable size, namely a maximum of 1,000 documents per query.
3. For each query we tried to apply all the patterns corresponding to each generic slot to all the retrieved documents. So if (X_0, \dots, X_n) are the variants of the query name and *PatternsGenericSlot* contains the patterns of a generic slot we look for the occurrences of an X_i followed by a pattern. The text following this pattern is thus a candidate to be the value of such slot. For locating the right limit of this text we used the same grammars used for extraction in step 2.

4.3 Unsupervised learning approach

Our second approach for learning IE patterns is completely unsupervised from the point of view of using annotated slot-filler examples. Our goal is to explore the appropriateness of using clustering techniques to discover patterns useful to detect relevant relations between pairs of named entity types occurring in text, and then, classifying the relevant relations into the set of possible slots in an unsupervised manner. Following, we describe both the relation detection pattern learning approach and the relation classification approach. These techniques are an enhancement of our KBP 2012 unsupervised relation detector (Run2).

4.3.1 Relation Detection

For each slot in a template of the KBP scenario of extraction, we can define the pair (t_1, t_2) as the pair of entity types associated to the template itself (t_1 can be *ORG* or *PER*) and to the slot (t_2 can be *AGE*, *PER*, *ORG*, *CITY*, *COUNTRY*, *RELIGION*, *CHARGE*, and so on). For each (t_1, t_2) , the procedure starts by gathering the set X of entity pairs,

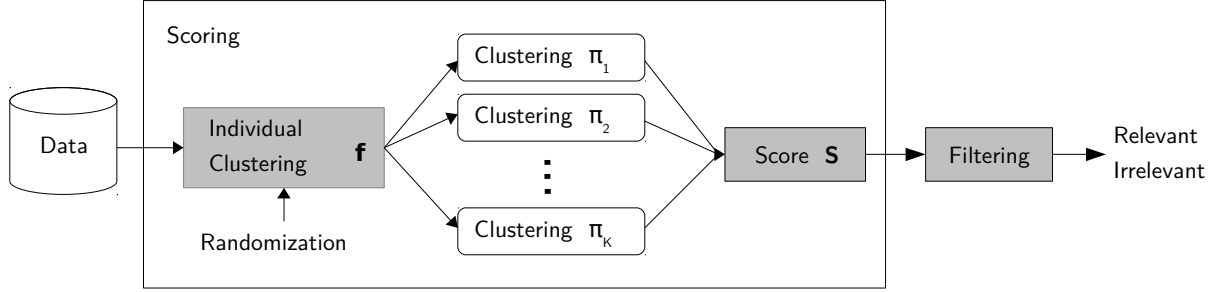


Figure 6: RD-EWOCs approach for relation detection

$x_i = (e_1, e_2)$, being t_1 and t_2 the entity types of e_1 and e_2 , respectively, and co-occurring in sentences of the document collection. Most of the pairs x_i will not be linked by any particular relation. In fact, a minority of them will be effectively related. In this context, minority clustering can be used to detect groups of related entity pairs as foreground clusters and discard non-related ones as background noise.

Based on these assumptions, our goal in KBP 2013 is to extend our previous year experiments using the Ensemble Weak minority Cluster Scoring (EWOCs) algorithm (González and Turmo, 2012). Concretely, we have used the default configuration to deal with the relation detection task (González and Turmo, 2009; González, 2012), RD-EWOCs up to now, which is briefly described below.

Figure 6 depicts the RD-EWOCs general algorithm. It requires to represent each example as a binary feature vector. The default features used to represent each entity pair $x_i \in X$ are described in Table 4. The algorithm consists in two main steps: the *scoring* of the set of entity pairs related to a particular (t_1, t_2) and the *filtering* of the relevant pairs.

Scoring. Briefly, using an individual weak clustering algorithm f , we randomly produce R clustering models, $\pi = \pi_1, \dots, \pi_R$ where $R = 100$ by default, from X . The default f for RD-EWOCs is a *Random Bregman Clustering* algorithm, a partition clustering algorithm which consists of the following steps:

- For each clustering model $\pi_c = \{\pi_1^c, \dots, \pi_k^c\}$ randomly select both the number of clusters $k \in [2, k_{max}]$, where $k_{max} = 50$ by default, and the k seeds, $\{x_1^c, \dots, x_k^c\}$.
- For each entity pair $x_i \in X$ and cluster $\pi_j^c \in \pi_c$

compute membership grades using a Gaussian-kernel distance as Bregman divergence as follows:

$$grade(x_i, \pi_j^c) = \frac{e^{-D(x_j^c, x_i)}}{\sum_{q=1}^k e^{-D(x_q^c, x_i)}}$$

$$D(x, y) = 2\alpha(1 - e^{-\gamma\|x-y\|^2})$$

where, parameters α and γ are automatically tuned in an unsupervised manner with the SOFTBBC-EM algorithm (Gupta and Ghosh, 2006).

- For each cluster $\pi_j^c \in \pi_c$ compute normalized sizes, $size^*$, as the product of the number of non-empty clusters,¹¹ K_c , with the sum of membership grades of all pairs $x_i \in X$:

$$size^*(\pi_j^c) = K_c \cdot size(\pi_j^c)$$

$$size(\pi_j^c) = \sum_{x_i \in X} grade(x_i, \pi_j^c)$$

$$K_c = |\{\pi_j^c | size(\pi_j^c) \geq 1\}|$$

Once π has been computed, each pair x_i is scored as the average of scores s_i^c achieved with each clustering model $\pi_c \in \pi$:

$$s_i^* = \frac{\sum_{\pi_c \in \pi} s_i^c}{R}$$

$$s_i^c = \sum_{\pi_j^c \in \pi_c} grade(x_i, \pi_j^c) \cdot size^*(\pi_j^c)$$

¹¹A cluster is non-empty if its size is greater or equal than a threshold. By default, this threshold is 1.

| | Feature | Description |
|-------------|---|---|
| structural | rightly/lefty dist_X ch_dist_X | the first NE type t_1 occurs to the right/left of t_2 distance in tokens between the pair is X distance in chunks between the pair is X |
| word based | left_X_Y/right_X_Y lmid_X_Y/rmid_X_Y left_X/right_X/mid_X l_left_X_Y/l_right_X_Y l_lmid_X_Y/l_rmid_X_Y l_left_X/l_right_X/l_mid_X l_L2gr_X/r_L2gr_X/m_L2gr_X n_left_X/n_right_X n_lmid_X/n_rmid_X | token X positions before/after to the left/rightmost NE of the pair has POS Y token X positions after/before to the left/rightmost NE of the pair has POS Y a token to the left/right/middle of the NE pair has POS X token X positions before/after to the left/rightmost NE of the pair has lemma Y token X positions after/before to the left/rightmost NE of the pair has lemma Y a token to the left/right/middle of the NE pair has lemma X bigram of lemmas X to the left/right/middle of the NE pairs token X positions before/after to the left/rightmost NE is a negative word token X positions after/before to the left/rightmost NE is a negative word |
| chunk based | ch_left_X_Y/ch_right_X_Y ch_lmid_X_Y/ch_rmid_X_Y chl_left_X_Y/chl_right_X_Y chl_lmid_X_Y/chl_rmid_X_Y cht_left_X_Y/cht_right_X_Y cht_lmid_X_Y/cht_rmid_X_Y | chunk X positions before/after to that containing the left/rightmost NE of the pair has type Y chunk X positions after/before to that containing the left/rightmost NE of the pair has type Y chunk X positions before/after to that containing the left/rightmost NE of the pair has a head with lemma Y chunk X positions after/before to that containing the left/rightmost NE of the pair has a head with lemma Y chunk X positions before/after to that containing the left/rightmost NE of the pair has a head with POS Y chunk X positions after/before to that containing the left/rightmost NE of the pair has a head with POS Y |

Table 4: Default feature set for RD-EWOCs

Filtering. Using the same idea as for filtering the most relevant documents in the preprocess (see Section 4.1), the set \hat{X} of those pairs having greater or equal score than the one supporting the maximum convexity of the curve, x_{th} with score s_{th} , is considered as the set of relevant entity pairs:

$$\hat{X} = \{x_i \in X | s_i^* \geq s_{th}\}$$

$$s_{th} = \min_i \sqrt{s_i^{*2} - (i / \max i)^2}$$

4.3.2 Relation classification

The unsupervised pattern-detection we have described so far, produces a set of entity pairs (e_1, e_2) that are related. But the exact nature and meaning of this relation remains unknown. Thus, we implement an unsupervised classification method that assigns each entity pair to the most likely template slot defined in the KBP evaluation. Additionally, it is necessary to allow a new slot, called OTHER, for relations not present in the KBP definition.

We use an unsupervised similarity measure to map each relation (t_1, t_2) to one of the template slots available for this specific pair of types, including

OTHER. The slots are characterized by their lexical content as follows:

Slot characterization. We take the *description* field from the official slots definition document (Ellis, 2013) corresponding to the pair (t_1, t_2) . The definition is lemmatized and the tf/idf of each lemma is computed. Each slot is interpreted as a point in a vector space and is represented as a binary feature vector defined by the lemmas of nouns and verbs present in the definition.

We group the slots according to the entity types involved, such as $(person, date)$, $(person, location)$, $(organization, date)$. Then we define the OTHER relation as the complementary of the union of all slot vectors in each group, thus being the point that is more different from any other slot in the space.

Certain number of options and variations can be tested within this approach: using only the first sentence of the provided definition (the second is usually some technical clarification), using the examples provided with the descriptions, limiting the amount of context words that define the feature vector, filtering out specific words, use alternative representations instead of lemmas (e.g. forms, Word-

Net synsets). We report several experiments on this topic in Section 4.5.

Mapping. In this step we try to map each detected relation to one of the template’s suitable slots for that specific pair (e.g. the pair (*person,organization*) can correspond to the slots: *employee_or_member_of* and *schools_attended*). Human experts have selected what t_2 types are the most suitable for each slot.

Each relation is encoded in the same vector space as the slot definitions using the sentence where it has been found. Then, we calculate the cosine similarity between between this vector and all the slot vectors we have previously computed (including the OTHER slot).

Finally, the relation is mapped to the most similar slot. If this is the OTHER slot, we consider it is not a relation representable in the KBP definition and it is deleted as if it had never detected by our system.

4.4 Integration

Following the application of both the unsupervised and the supervised approaches, a common integration step is applied in which the extracted information is, on one hand, arranged according to the KBP submission guidelines, and on the other hand, it is filtered by means of a very basic common sense checking.

Firstly, the ignore slots for each query are removed and those slots not belonging to this list which have not been extracted by the IE system are added with NIL response. Dates, which have been extracted with the TARSQI toolkit, are normalised so that they follow KBP standard.

In the case of single-valued slots, only one slot filler must be returned. Therefore, if more than one filler had been extracted by the IE system, we first look for *compatible fillers*, those that may refer to the same contents. Our methodology is very basic and just detects names that subsume other more informative ones or dates that include other more specified ones (it might be easily extended by taking into account acronyms or demonyms). These *compatible fillers* are filtered, removing all but the more informative one. If the slot keeps on having more than one filler, the one with the higher confidence score will be selected. This general criterion is applied to all slots but dates, in

which case an additional sanity checking is performed: we take advantage of the existence of inverse slots (*per:date_of_birth* vs *per:date_of_death* and *org:date_founded* vs *org:date_dissolved*) and filter the different fillers of these slots assuming that the initial date must be strictly smaller than the final one and that the number of years between both should make sense.

In the case of list-valued slots, to begin with the *compatible fillers* are filtered (except for the *per:title* slot, according to the KBP guidelines). Subsequently, additional sanity checks are carried out, such as:

- Removing those fillers coincidental with the query name from slots such as *alternate_names*, *members*, *member_of*, *spouse*... We do not remove from *per:parent* and *per:children*, which may coincide.
- Eliminate intersections among fillers of different slots which are not compatible (such as the different family slots for *person*, *org:parents* vs *org:subsidiaries* or *org:members* vs *org:member_of*): only the coincidental filler with higher score will remain.
- Limit the number of fillers for certain slots, selecting just the higher confidence score ones (top 2 for *per:parents* and top 3 for, among others, *per:spouses*, *per:schools_attended*, *org:parents*, or *org:political_religious_affiliation*).

This latter step looks a priori more relevant for the unsupervised approach, where the extraction process is more massive and precision should become more of an issue than recall.

4.5 Results and analysis

We presented the unsupervised based approaches as Run1 and Run2, and the distant-learning based approach as Run3. A shallow manual analysis of our submissions shows a very low performance, specially regarding recall, of both systems due to different reasons described below.

Regarding the query preprocessing, which is used in both systems, the generation of alternate names for ORG queries, unlike to PER queries, does not

perform properly. We consider that the rules for generating them have to be reviewed. The average number of alternate names per query was 8.7 for PER and only 3.3 for ORG. This difference has an obvious influence in the number of retrieved documents in the IR step.

Regarding the document preprocess, also used in both systems, there were some queries for which, wrongly, just the reference document was found as relevant. In general, compared with 2012 results, fewer documents have been collected for most of the queries, because this year we decided, perhaps erroneously, focusing on avoiding noisy documents. 28 queries (7 PER, 21 ORG) retrieved less than 50 documents in the IR step (8 of them less than 10 documents). The reason for such behaviour was our assumption that alternate names of queries occur as NEs in preprocessed documents. However, this fact strongly depends on the accuracy of the NERC system used. Concretely, no alternate name has been recognised as NE for the reference document of some queries with the NERC system we used. As a consequence, the set of keywords useful to retrieve more relevant documents is empty for these queries. This makes our relevant feedback approach stop without providing more documents than the reference ones.

The pooling approach used for the assessment means that the set of slot fillers in the official evaluation is not exhaustive. This has prevented us from being able to evaluate the impact of the subsequent integration step into the results, since a high percentage of the fillers discarded in this phase are new, i.e. non evaluated (the exact percentages of new fillers are 96% for Run1, 87% for Run2 and 59% for Run3). However, we have stated that, as expected, integration filters 50% of the slot fillers for the unsupervised Run1, while it only filters 24% of the fillers for Run2. But, surprisingly, it is the supervised Run3 who gets the higher percentage of fillers filtered (63%), indicating a concentration of a high number of (repeated and non-repeated) fillers for a small number of slots.

Run1 and Run2. These two runs differ in how the detected relations are classified. The official results and other experiments are presented in Table 5. Although the scores are very low, they greatly improve

our KBP 2012 results (Gonzalez et al., 2012). We believe that an F1 score of 8.62 is fair for an unsupervised system but unfortunately we are not aware of any other unsupervised relation extractor we can compare to. Our recall of 9.82% is even better than the 9.67% obtained by the 10th best ranked system in this evaluation.

Using the SF 2012 collection as a development set, we have taken these implementation decisions:

- In all the experiments we are using the same configuration for the relation detection: we use 10 randomly produced RD-EOWCS models with 5,000 weak clusterings in each one (a model computable in a reasonable time). An entity pair is considered to be related whenever any one of the 10 models considers it relevant.
- We have set three separate detectors, one for each part of the documents collection: news, weblogs and fora. We have experimentally tested that this strategy is better than having a single detector, probably due to the different nature of the text.
- The word based features (Table 4) are produced for words within a window of size 3 around the target entities.
- Finally, sentences not containing a verb are discarded and entity pairs distant more than 16 words are also discarded (for both producing the models and processing actual KBP queries). These decisions were tested on the 2012 SF corpus and help reduce both the amount of noise and the number of examples and computation time for the ensemble clustering.

As can be seen in Table 5, we have tested seven variations of our relation classifier characteristics (Section 4.3.2) using the same set of detected relations:

- Provided that we are considering the relation of any pair of entities cooccurring in a sentence, there can be very distant pairs which will certainly have spurious words inbetween them. We tackle this issue setting a *semantic threshold* that is the number of words around the entities that are used to compute the vector defining the relation.

| ID | desc. | #fill. [†] | R(%) | P(%) | F1 |
|------|------------|---------------------|------|-------|------|
| Run1 | 10/A/+be | 1872 | 9.82 | 7.69 | 8.62 |
| — | 10/A/-be | 969 | 7.02 | 10.63 | 8.46 |
| — | 10/F/-be | 814 | 6.41 | 11.55 | 8.24 |
| Run2 | 3/A/-be | 639 | 5.73 | 13.15 | 7.98 |
| — | 3/F/-be | 555 | 5.52 | 14.59 | 8.01 |
| — | 10/E/-be | 2524 | 2.96 | 4.02 | 2.33 |
| — | 10/A+E/-be | 2746 | 3.75 | 5.38 | 2.87 |
| Run3 | — | 103 | 2.07 | 29.12 | 3.85 |

Table 5: Experimental results of the unsupervised learning approach with several configurations of the classifier. Meaning of the description column: **3/10**: number of context words used to define the vectors; **F/A/E**: using the First/All of the sentences that describe the slot or the Example sentences given in the SF guidelines; **+/-be**: the verb *to be* is or is not used in the vector.

†: n.b. Only the results of the official runs are confiable since our other runs are mainly composed of new (i.e. non-evaluated) fillers.

- The source of the slot definition it can be either the full description provided by the organization or only the first sentence. We have observed that the second and further sentences are more an aclaratory comentary than a definition. This is noted as F or A. We have also experimented using the example relations given in the guidelines as the slot definition (type E).
- Some verbs, like *be* and *have*, do not seem very informative about what relation they express but are extremely frequent in the examples. We have experimented dropping *be* and *have* form the slot definitions.

Two of this configurations were selected to be our official runs. Our objective was to have the largest recall (Run1) and the largest precision (Run2) according to our experiments with KBP 2012 corpus. The experimental results confirm the intuition that reducing either the vocabulary, context size or description length should reduce the recall and increase the precision.

Main reasons that explain the low figures are the following:

- First of all we have to consider the sheer amount of errors accumulated through our

pipeline (IR, NERC) and limitations of the approach (single sentence relations, only relations of NEs, no use of coreference). These are difficult to quantify.

- According to (González, 2012), EWOCS performance improves if the size of the ensemble of clusterings is selected taking into account the size of the data set, so that large data sets require large ensembles. In this sense, we think that our unsupervised approach requires much more than 5,000 clustering models to achieve good results for detecting slot fillers in KBP corpus. This does not unduly penalize the efficiency of the system given that the computation of the clustering models can be paralelized.
- The relation classifier has a very limited representational power since it uses a simple measure of lemma overlapping.

Finally, we have also evaluated the performance of the relation detector RD-EWOCS. For this task, we have taken all the detected relations (20,849) and taking into account the entity pair types we have checked if there exists a correct or redundant evaluation assessment for this filler with a suitable slot type (e.g. for a *per:per* relation, a correct fille for the types *spouse* or *parents* or *sibling*, and so on). This evaluation yelds 777 correct relations and 1,963 wrong relations (the rest does not exist in the assessments), which is a precision of 26.7%. Note that these 777 relations are not a real upper bound of what our systems can extract since we can have multiple and redundant relations for the same slot filler, the real upper bound would be much lower. Unfortunately the large amount of unjudged slot fillers makes impossible to draw sound conclusions.

Taking into account all these points, we think that there is room enough for improvements in the unsupervised approach to the SF task, specially for the classification part.

Run3. The statistics of the official results of our distant learning run were of 0.02 Recall (0.04 in 2012), 0.26 Precision (0.22 in 2012), and 0.04 F1 (0.07 in 2012). So, compared with our 2012 results we see a severe drop in recall and a clear improvement in precision (due to a clear improvement for

| Type of Text |
|------------------------|
| was born in |
| born |
| on <DATE> in |
| in |
| <DATE> in |
| born in |
| was born on |
| was born on <DATE> in |
| <DATE> |
| was born |
| was born and raised in |

Table 6: Some of the best scored patterns for the generic slot *per:date_of_birth*

PER, 12%, against a drop for ORG, from 0.15 to 0.07, i.e. a 48%). These results are not bad in terms of Precision but are very low in terms of Recall. As we do not use any confidence scoring for our answers, NIL is assigned to slots to which no valid assignment has been found. So, for analysing our errors we focus on not NIL answers. For analyzing our results we proceed grouping the results in two dimensions: queries and slots.

From the queries dimension we observe that the distribution of correct answers is extremely query biased. Compared with our 2012 results, for 36 out of 50 PER queries some slot filler are found (not always correct) against 13 out of 40 last year. For ORG only 5 slot fillers were found against 5 last year. In fact most of the queries have no answers at all (only 36 from the 50 PER queries and 5 from 50 ORG queries generated some results). This explain our low Recall figures. A second observation regarding the extremely unbalanced performance of our system for PER and ORG is that 26 correct answers were extracted in top position for PER (0.29 Precision) but only 1 for ORG (0.08 Precision).

Moving to the slot dimension we discover that 11 out of the 16 ORG slots produce no results (only 13 out of 25 for PER). We have manually analyzed a sample of 25 patterns from the pattern sets of all the slots. The results were significant: for PER, all but one (*per:age*) of the slots got an accuracy over 0.9, while for ORG only one slot (*org:alternate_name*) got an accuracy over 0.5.

The reasons why this happens are multiple:

- *PagesORG* are less accurate than *PagesPER* possibly due to the difficulty of obtaining the set of relevant categories for ORG.
- Not disposing of a grammar of names for ORG (difficult to get due to the high variability of ORG subtypes) reduces the number of alternate names and accordingly the number of documents retrieved (in fact most of the ORG variants came from acronym expansion/detection).
- Less infoboxes are filled for ORG.
- ORG generic slots are more difficult than PER ones, For many slots the grammars used are really precise (as DATE or PLACE) in the case of PER, but present a great variability in the case of ORG. Locating a PERSON, a DATE or a LOCATION is easier than locating an ORGANIZATION.
- The patterns extracted for PER are in many cases very short (as shown in Table 6) and occur many times. This is not the case for ORG where many patterns are long and occur with very low frequency.
- Most of generic slots for PERSON are single-valued, in the case of ORG the situation is the contrary.
- While more or less all the PERSON present a similar profile, ORGANIZATIONS, present a great variability, for instance a political PARTY or a football TEAM have few points in common.
- Sometimes the mappings between generic and specific slots provided by KBP organizers were not accurate enough. For instance, for *per:age*, the slots contain a large number of varied wordings containing the age together with many other useless information. The grammar learned from this material is obviously extremely unaccurate.

Acknowledgements

This work has been produced with the support of the SKATER project (TIN2012-38584-C06-01).

References

- E. Agirre, A. X. Chang, D. S. Jurafsky, C. D. Manning, V. I. Spitzkovsky, and E. Yeh. 2009. Stanford-UBC at TAC-KBP. In *Proceedings of the Second Text Analysis Conference (TAC 2009)*, Gaithersburg, USA.
- R. C. Carrasco and J. Oncina. 1994. Learning stochastic regular grammars by means of a state merging method. In *Grammatical Inference and Applications*, pages 139–152. Springer-Verlag.
- S. Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague*.
- J. Ellis. 2013. *TAC KBP 2013 Slots Descriptions, version 1.0*. Linguistic Data Consortium. <http://projects.ldc.upenn.edu/kbp/>.
- C. Fellbaum. 1998. Wordnet: An electronic lexical database. In *MIT Press*.
- G. Garrido, B. Cabaleiro, A. Peñas, A. Rodrigo, and D. Spina. 2011. Distant supervised learning for the TAC-KBP Slot Filling and Temporal Slot Filling Tasks. In *Text Analysis Conference (TAC 2011)*.
- E. González and J. Turmo. 2009. Unsupervised relation extraction by massive clustering. In *Proceedings of the 9th IEEE International Conference on Data Mining (ICDM)*.
- E. González and J. Turmo. 2012. Unsupervised ensemble minority clustering. In *Research report. Department of Llenguatges i Sistemes Informàtics (LSI - UPC)*.
- E. Gonzalez, H. Rodriguez, J. Turmo, P.R. Comas, A. Naderi, A. Ageno, E. Sapena, M. Vila, and M.A. Marti. 2012. The talp participation at tac-kbp 2012. In *Text Analysis Conference, USA*.
- E. González. 2012. *Unsupervised Learning of Relation Detection Patterns*. Ph.D. thesis, UPC Programme in Artificial Intelligence.
- G. Gupta and J. Ghosh. 2006. Bregman bubble clustering: A robust, scalable framework for locating multiple, dense regions in data. In *Proceedings of the ICDM conference*.
- J. Hoffart, M. A. Yosef, I. Bordino, H. Furstenu, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. 2011. Robust disambiguation of named entities in text. In *the EMNLP Conference, Scotland*.
- J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. 2013. Yago2: a spatially and temporal-ly enhanced knowledge base from wikipedia. In *Artificial Intelligence Journal*.
- M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the ACL conference*, pages 1003–1011, Singapore.
- L. Ratnoff and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*.
- M. Surdeanu, D. McClosky, J. Tibshirani, J. Bauer, A. X. Chang, V. I. Spitzkovsky, and C. D. Manning. 2010. A simple distant supervision approach for the TAC-KBP slot filling task. In *Proceedings of the TAC-KBP 2010 Workshop*.