Lightly supervised acquisition of named entities and linguistic patterns for multilingual text mining

César de Pablo-Sánchez · Isabel Segura-Bedmar · Paloma Martínez · Ana Iglesias-Maqueda

Abstract Named Entity Recognition and Classification (NERC) is an important component of applications like Opinion Tracking, Information Extraction, or Question Answering. When these applications require to work in several languages, NERC becomes a bottleneck because its development requires language-specific tools and resources like lists of names or annotated corpora. This paper presents a lightly supervised system that acquires lists of names and linguistic patterns from large raw text collections in western languages and starting with only a few seeds per class selected by a human expert. Experiments have been carried out with English and Spanish news collections and with the Spanish Wikipedia. Evaluation of NE classification on standard datasets shows that NE lists achieve high precision and reveals that contextual patterns increase recall significantly. Therefore, it would be helpful for applications where annotated NERC data are not available such as those that have to deal with several western languages or information from different domains.

Keywords Named entity recognition and categorization · Information extraction · Multilingual natural language processing · Bootstrapping algorithms

1 Introduction

Nowadays there exists an increasing need for intelligent applications for helping to access information. Moreover, these applications are particularly necessary when the information is located in huge multilingual collections as is becoming common with the World Wide Web.

C. de Pablo-Sánchez (✉) · I. Segura-Bedmar · P. Martínez · A. Iglesias-Maqueda
Department of Computer Science, Universidad Carlos III de Madrid, 28911, Leganés, Madrid, Spain
e-mail: zesar.depablo@gmail.com

I. Segura-Bedmar
e-mail: isegura@inf.uc3m.es

P. Martínez
e-mail: pmf@inf.uc3m.es

A. Iglesias-Maqueda
e-mail: aiglesia@inf.uc3m.es

News Tracking (NT) [27], Opinion Mining (OM) [21], or Question Answering (QA) [12] are examples of intelligent applications for information access. All of them share a common requirement: they need to solve the task of Named Entity Recognition and Classification (NERC).

One of the main drawbacks in previous solutions for NERC is the requirement of specific language tools and resources. Particularly, annotated corpora is a typical resource needed for these kinds of applications in machine learning (ML) solutions [19,29,30]. Unfortunately, the annotation of a corpus is an expensive and tedious task that human experts have to provide for each domain and language. Producing annotated corpora quickly becomes the bottleneck in a multilingual application that considers more than two or three languages [27] not only because of the time, but because of the required expertize.

Multilingual applications are more and more needed due to the increasing process of globalization that, for instance, the World Wide Web presents. Moreover, multilingual societies like Europe or India need to extract information taking into account different languages.

In this work, we explore an alternative to acquire Named Entities and Linguistic Patterns (contextual patterns) for NERC in western languages nearly from scratch, without the requirement of annotated corpora or other supervised and costly language-dependant resources or tools. Moreover, the system presented in this paper has been designed with the objective of being easily adaptable to new western languages and new domains. The experiments address this need by requiring only an initial set of a few examples of Named Entities to bootstrap useful resources.

The paper is structured as follows: Sect. 2 presents the state of the art in multilingual NERC; the system proposed in this work is described in Sect. 3; the evaluation of the system, carried out on three different collections and two languages (Spanish and English), is detailed in Sect. 4. Finally, Sect. 5 outlines the main conclusions and future lines.

2 Related work

The objective of the NERC task consists of processing a text and identifying sequences of words which represent entities like locations or organizations, among others. This task can be performed in one step or broken into two subproblems. On one hand, NE recognition or detection aims at marking the boundaries of the mention of an entity in running text. On the other hand, NE classification should assign the correct category to the span of text. The most basic set of classes includes PERSON, LOCATION, and ORGANIZATION, but existing taxonomies have extended the number of classes up to 200 for the general domain [25]. Moreover, in specific areas, such as the biomedical domain, the classes of interest include genes, proteins, or drugs, for example. Recent surveys in NERC are [18] and [24].

Initially, we distinguish two main paradigms for NERC: Knowledge engineering techniques and ML approaches. Knowledge engineering techniques combine linguistic rules and the integration of handcrafted lists of names and other heuristics. This approach was initially taken by the NewsExplorer system [26,27] that combined the use of multilingual dictionaries, gazetteers, and rules for NE recognition and classification. Development environments like GATE [8] support the process with a specialized language (JAPE) and other tools. Nevertheless, it has proven hard to maintain due to the fact that adaptation of linguistic rules and resources to new domains and languages is required.

Machine learning techniques have been explored beginning with evaluations like the CoNLL forum. The shared tasks at the CONLL 2002 [29] and 2003 [30] explored NERC for Spanish, Dutch, English, and German. The best performing systems used supervised approaches like [4,11] and obtained results between 72 and 89 % in F-accuracy. The results depend on the language and the features used. While some of the features are largely language independent, it is also common that to obtain top results they integrate other language-dependant resources like NE lists, gazetteers, or POS taggers. In addition, supervised systems need a significant amount of annotated corpora in order to estimate the model parameters. On et al. [20] describe a work to disambiguate person names in digital libraries by using clustering methods based on graph partitioning algorithms that was trained with a subset of DBLP records and a set of web pages previously annotated. The cost of annotating data turns into a limitation when we approach a new language, a new domain, or a different set of classes. In all those cases, it is desirable to reduce the amount of supervision that is required. Therefore, both paradigms are highly language-dependent and involve a high cost in manual labor and supervision of domain experts.

Semi-supervised learning is a complementary line of research aimed at reducing the amount of tagged training data needed or improving accuracy results by using unannotated data. Collins and Singer [6] explore the use of semi-supervised learning for NE classification into the three basic classes (PERSON, LOCATION, and ORGANIZATION). It starts from a few rules and explores techniques like Expectation Maximization, self-learning, and co-training. Nevertheless, they use parsing to recognize NEs which is a strict requirement for languages other than English. Cucerzan and Yarowsky [7] report experiments on learning a NERC to extract and classify PERSON (first and last name) and LOCATION. Their approach achieves accuracy between 40 and 70 % for several languages. Their system learns weighted tries for contextual and morphological features using a list of initial seeds and a corpus.

Another line of work has focused on bootstrapping semantic dictionaries that can be integrated in Information Extraction modules. For example, [28] have used successive improved modifications of a bootstrapping algorithm to learn a list of concepts for a domain-dependent semantic model. This work and [32] have shown that learning multiple semantic classes simultaneously improves results. Both approaches use other Natural Language Processing (NLP) tools like POS tagging and parsing. On the other hand, the KnowItAll system [10] has been used to compile a list of names and their categories from the Web by querying a search engine. KnowItAll combines three acquisition techniques to mine the web: Hearst patterns [13], extraction from large structured lists, and pattern learning. Their experimental results show that Hearst patterns and structured list extraction are the most successful in a Web context. Structured list location and extraction have also been used by [17] to acquire names for a dictionary-based NERC system. However, they also note that it is difficult for locating large lists of names for languages other than English. There are other works that follow this bootstrapping approach applied to other text processing tasks, such as [16], that show a method for text classification based on fuzzy partition clustering in order to obtain a small quantity of labeled training data.

The compilation of large lists of names is also possible from Wikipedia, a multilingual collaborative, free-content encyclopedia. Although its language distribution varies widely, the English version already contains more than 2 million entries. Several of its features have been exploited in order to extract useful knowledge. For example, [31] and [15] use the first sentence of an article to compile a list of categorized entities using a variety of tools like Wordnet and POS tagging.

Richman and Schone [22] take a different approach on using Wikipedia. They define a procedure to generate annotated corpora using wiki formatting and other structures such as category pages, articles, and interwiki pages. Corpora for several European languages are prepared using a limited language knowledge. An HMM tagger based on a BBN IdentiFinder [3] is trained with Wikipedia corpora and tested in newswire corpora with very good results. This study shows that the performance is comparable to training the system with annotated corpora ranging from 20,000 to 40,000 tokens. Other cross-lingual approach to NERC, [34], describes several methods to project the labels from a rich-resource mention tagger (English) to another poor-resource language using Statistical Machine Translation. Their model aligns the labels with their translations and achieves reasonable performance if we consider that there are no training data in the poor-resource language. Dorji [9] proposed an approach to extract terms that serve to identify document fields to be used in document classification and passage retrieval combining linguistic and statistical methods; this work uses Wikipedia articles classified by Wikipedia categories (that is, domain-specific corpora) as well as Reuter RCV1.

Knowledge engineering and ML techniques require language-dependent resources to process texts and involve the supervision of domain experts to define linguistic rules or annotate the training data, respectively. However, most works using semi-supervised learning or bootstrapping methods were based on language-dependant resources and have hardly ever carried out the detection of names for languages other than English. This work proposes a bootstrapping algorithm easily extendible to new languages and new domains since it only requires minimal supervision and few language-dependant resources. For each semantic category (PERSON, LOCATION, and ORGANIZATION), the algorithm only needs a few examples and retrieves the most frequent patterns co-occurring with them. These patterns are produced from a single text context substituting each token with a wildcard to retrieve new entity names. Those patterns that are only composed of stopwords and wildcards are filtered. Thus, the stopword list is the only specific language resource that we use. The process is repeated incrementally to acquire new entity patterns and new entity names. To the best of our knowledge, this is the first bootstrapping system applied to the detection of entity names for Spanish. Another main advantage of our algorithm is that it is also domain-independent, since it can be easily applied to the detection of other semantic categories by just providing a small set of examples with no need for any additional setting.

3 System description

This section describes the bootstrapping system proposed. The goals of the system design are (1) the acquisition of useful resources for NERC from a handful of examples and (2) to be applied across languages and domains. The system is able to simultaneously acquire two resource lists for every class of interest. The first list consists of examples of Named Entities of a class, also called instances of the class. If we think of a NE class as a unary predicate like PERSON(X), the list will be composed of person names like `Barack_Obama`, `Bill_Clinton`, etc. The second list contains textual patterns that frequently co-occur with valid entity instances. Among those, there would be examples of Hearst patterns like `persons_such_as` or `presidents_like` but also more general patterns like `told_reporters` that are indicative of a class like PERSON.

3.1 Definitions

Before describing the system in detail, we briefly introduce the main concepts that are used in this work.

– **Document collection** or **corpus** is denoted as D and represents a collection of documents in a language. The document collection is indexed in order to scale for efficient access.
– **Entity class** is each of the semantic categories that we have identified as important in our document collection or useful for our application. In news genre, they are classes like

Table 1 Examples of textual context with mentions for `Bill_Clinton`

		S_l	S_e	S_r	
1			Bill Clinton	and his wife	as part
2	it is	time for President	Bill Clinton	to be as	big a
3		President	Bill Clinton	often laments that	his achievements
4	The past	week may convince	Bill Clinton	that his most	recent predecessors
5			Bill Clinton	told reporters that	there

Table 2 Pattern mentions and text contexts for the pattern instance `p(and_his_wife,`$\rightarrow$`)`

		S_e	S_r	
1		Bill Clinton	and his wife	as
2	assertion by	President Clinton	and his wife	that
3	Long before	President Clinton	and his wife	began
4	his friend	Vince Foster	and his wife	
5	consul general	Alberto Boniver	and his wife	Suzy
6	Mao invited	Lin Piao	and his wife	to
7	expenses for	Davis	and his wife	Christina

PERSON, LOCATION, and ORGANIZATION. Entity classes may be interpreted as logical predicates. A predicate PERSON(X) will define a set of objects that belong to a set like {x: x is a person}.

- **Semantic model** is the set of entity classes that the algorithm simultaneously uses in a run. We name semantic models with the initials of their predicates when possible. For example, the PLO stands for a semantic model composed of PERSON, LOCATION, and ORGANIZATION classes. More formally, a semantic model is composed by k classes that would be denoted as $R_k(x)$ or R_k for short.
- **Entity instance** is used for each of the named members of an entity class. Valid entity instances that fulfill the predicate PERSON(X) could be `Bill_Clinton` and `Clinton`. We will consider them as two different entity instances, ignoring whether they refer to the same real-world entity or not. The set of entity instances is a relation in the mathematical sense. For each entity class, we will aim at obtaining a relation $E_k(x)$ of entity instances that satisfy the predicate R_k.
- **Entity mention** refers to the concrete occurrence of an entity instance in a document collection. Table 1 shows examples of mentions for the entity instance `Bill_Clinton`. The substring that corresponds to the entity mention is marked as S_e. S_l and S_r are the left and right substrings that are used as context for pattern generation.
- **Pattern instances** represent unique sequences of tokens that are frequently adjacent to an entity instance of a given class. $P_k(text, dir)$ is the relation of pattern instances associated with a predicate R_k. In this case, the relation requires a token sequence ($text$) and their location with respect to the entity instance (dir). The pattern may be appear to the left of an entity mention ($dir = \leftarrow$) or to the right ($dir = \rightarrow$).
- **Pattern mention** is therefore the occurrence of a pattern instance inside a document. Table 2 shows several pattern mentions related to the single pattern instance

(and_his_wife,$\rightarrow$). In this case, S_r is the substring that matches a pattern instance and S_e the text that would correspond to a hypothesized entity mention.

– **Text Context** is used to produce the graph that links pattern instances and entity instances. The distinction between instances and mentions is important. The algorithm acquires and classifies instances but it uses mentions to link entity instances and pattern instances when they are adjacent in the same text context. Two different types of text contexts are used. The contexts of entity mentions are used to generate left and right pattern instances which can be seen as links. If the context of a pattern mention matches an entity mention, it also generates a link.

3.2 Architecture

The algorithm that uses our bootstrapping system is detailed in Algorithm 1. The process begins with an indexed collection of documents D and a semantic model M with a set of predicates R_k from which we want to get the resources. For each predicate R_k, we will obtain a list of entity instances E_k and a list of pattern instances P_k. Each of the predicates requires a set of initial seeds and a regular expression that help us to identify candidate entity mentions. In our experiments, we have just used the heuristic that NE should be capitalized, but other soft heuristics may easily be applied. The algorithm uses dual bootstrapping of names and patterns. It proceeds in an iterative way repeating two phases, Pattern Expansion and Entity Expansion for each of the predicates. That is to say, Named Entity instances are used to discover new frequent Patterns instances and these patterns help to discover new name entities. The following subsections describe the bootstrapping algorithm in more detail with the help of example figures tracking the process performed in an iteration.

The system produces a bipartite directed graph $G(E, P, A)$ where E are entity instances, P are pattern instances, and A are the directed arcs between E and P vertices.[1] An entity instance is linked to a pattern instance when a query locates in the same context adjacent mentions of the entity and the pattern. Notice as a distinctive characteristic of this system that, in contrast with other bootstrapping algorithms, nodes are not only labeled but they are also discovered as iterations advance. The query exploration strategy could be classified as an Automatic Query Generation (AQG) strategy in the terminology of [14]. In fact, the property of exclusivity combined with the learning of k predicates would end up building a $k+1$-partite graph in the form $G(E_1, E_2, \ldots, E_k, P, A)$.

3.3 Pattern expansion

In this phase, a new set of entity instances are used to acquire new candidate patterns. Only a subset of the most promising ones are selected, evaluated, and consolidated according to the procedure described below.

3.3.1 Find pattern mentions

This step starts with a set of seeds selected by a human expert for each of the predicates. `Bill Clinton`,`George Bush`, and `Janet Reno` may be valid seeds for the predicate PERSON(X). Initial seeds were generated by introspection but a few guidelines were used for their selection:

[1] We would use A, from arcs, to denote edges instead of the more common E as the names clashes in our context.

Algorithm 1 SPINDEL meta-algorithm

```
Input:
$D$ an indexed corpus,
$R_k$ entity classes to learn
$E_k$ entity instances seeds for class $R_k$,
$P_k = \{\}$ an empty list of patterns for class $R_k$,
$\tilde{A}_k^p = \{\}$ pool of contexts extracted from entities
$\tilde{A}_k^e = \{\}$ pool of contexts extracted from patterns

Output: $E_k$ seeds, $P_k$ patterns

$t = 0$ {Start iterations in time}
for each k do
  $\tilde{A}_k^p \leftarrow findPatternMentions(E_k^0)$
  $E_k \leftarrow E_k^0$
end for
repeat
  $t = t + 1$
  {Pattern Expansion: Find new patterns instances from entities}
  for each k do
    $P_k^t \leftarrow selectCandidatePatterns(\tilde{A}_k^p)$
    $\tilde{A}_k^e(P_k^t) \leftarrow findEntityMentions(P_k^t, D)$
  end for
  for each k do
    $evaluatePatterns(P_k^t, E_k^{t-1})$
    $storeGoodPatterns(P_k^t, \tilde{A}_k^e(P_k^t)) \wedge removeBadPatterns(P_k^t, \tilde{A}_k^e(P_k^t))$
  end for

  {Entity Expansion: Find new entity instances from patterns}
  for each k do
    $E_k^t \leftarrow selectCandidateEntities(\tilde{A}_k^e)$
    $\tilde{A}_k^p(E_k^t) \leftarrow findPatternMentions(E_k^i, D)$
  end for
  for each k do
    $evaluateEntities(E_k^t, P_k)$
    $storeGoodEntities(E_k^t, \tilde{A}_k^p(E_k^t)) \wedge removeBadEntities(E_k^t, \tilde{A}_k^p(E_k^t))$
  end for

  {Delete old candidates from the pool for efficiency}
  if $(t - t_{offset})\%t_w = 0$ then
    $deleteVeryOldPatternMentions(\tilde{A}_k^p)$
    $deleteVeryOldEntityMentions(\tilde{A}_k^p)$
  end if
until $(|goodPatterns(P_k^t)| = 0 \wedge (|goodSeeds(E_k^t)| = 0)$
```

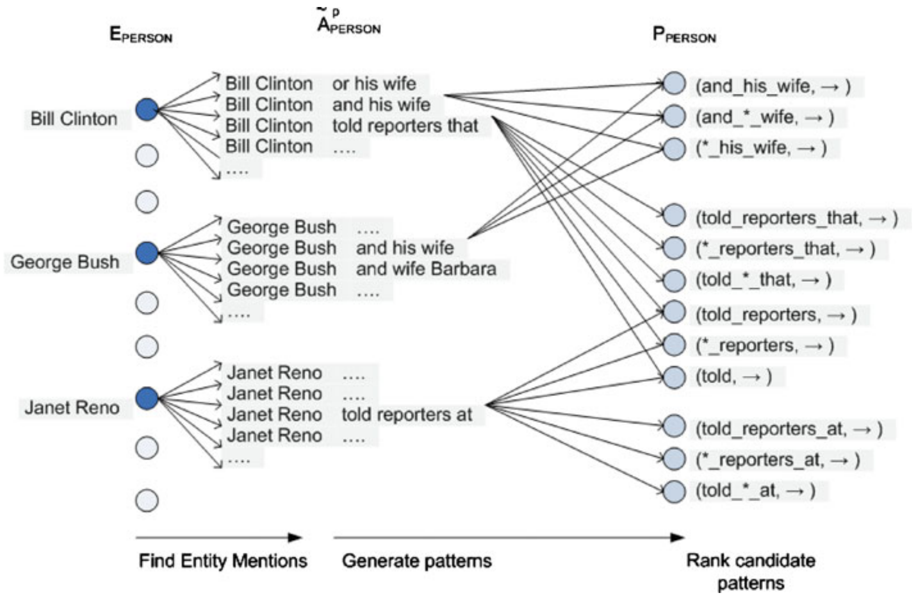


Fig. 1 The process of building the graph: Pattern expansion

– Moderate to large frequency in the corpus. List of frequent NE candidates based on a regexp pattern and a search interface were available. Initial bootstrapping requires that common patterns appear close to different entity instances.
– Some of the seeds may share similar context (e.g., politicians, sportsmen, etc.).
– Seeds should cover different subconcepts inside the semantic class.
– Avoid entities that several senses are known as initial seeds.
– Select long names (`Bill Clinton` better than short references `Clinton`)

For each instance, a query and a filter are generated that retrieve documents with entity mentions. For example, for the entity instance `Bill_Clinton`, we can generate a phrase query "`Bill Clinton`" and a similar regular expression filter `Bill s+Clinton`, where `s+` is the Java regular expression symbol matching one or more whitespaces. The textual context around each of the entity mentions (S_l and S_r) is used to discover candidate patterns. The process is depicted in Fig. 1. For simplicity, this example illustrates the process for right contexts; this context is similar for left ones.

3.3.2 Generate patterns

Pattern instances are generated from the tokens adjacent to an entity mention by applying a simple generalization process. In addition to the plain lexical pattern, each token may be substituted by a wildcard. Wildcards that appear on the edge of a pattern will always be matched; therefore, it is simpler to represent these patterns just as shorter ones. One parameter, w_l, controls the maximum length of the window used to generate patterns. Additionally, those patterns that are only composed of stopwords and wildcards are filtered.

We have used only lexical information in our patterns because our goal is to design an algorithm that will be useful for several languages. Therefore, the stopword list is the only

Table 3 Generation of pattern instances for a context to the right of an entity mention, *c* is for candidate patterns and *d* is for discarded ones

c	w_1	w_2	w_3	and	his	wife
c	*	w_2	w_3	*	his	wife
c	w_1	*	w_3	and	*	wife
c	*	*	w_3	*	*	wife
d	w_1	w_2		~~and~~	~~his~~	
d	*	w_2		~~*~~	~~his~~	
d	w_1			~~and~~		

specific language resource that we use. For this reason, no linguistic processing, including stemming, is used at this stage.

Table 3 depicts the pattern generation process for the right context S_r=`and his wife`. Assuming that the tokens *and* and *his* are in the stopword list, the four first pattern instances are accepted while the rest are discarded (`(and_his,`$\rightarrow$`)`, `(and,`$\rightarrow$`)`, `(*_his,`$\rightarrow$`)`). Figure 1 depicts the process for several entities and patterns. Note that a single pattern instance may be linked to several entity instances by means of appearing in the same or similar contexts once wildcards are introduced.

The whole process of finding entities and generating patterns is outlined in the Algorithm 2.

Algorithm 2 FindPatternMentions

```
Function: $FindPatterns(E, D)$
Input: $E$ a set of entity instances, an indexed document collection D
Parameters: $N_d^e$ the maximum number of documents retrieved in a query, $w_l$ is the window size for a pattern
Output: $\tilde{A}_p(e, p) = \{\}$ a set of textual contexts that link entities e in $E$ with candidate patterns $p$

for each $e$ in E do
  $q \leftarrow generateQuery(e)$;
  $f \leftarrow generateFilter(e)$;
  $T \leftarrow filterDocs(retrieveDocs(q, N_d^e), f)$;
  for each $t$ in $T$ do
    $\tilde{A}_p(e, p) \leftarrow \tilde{A}_p(e, p) \bigcup generatePatterns(e, t, w_l)$
  end for
end for
```

3.3.3 Select candidate patterns

The evaluation of candidate pattern instances is an expensive computational process and only a subset N_l^p of all the candidates is evaluated in each iteration. The selection must avoid a type of bias that appears as a consequence of retrieving candidates by querying text. Consider using the entity instance Bill_Clinton, it should sieve those patterns that are associated only to a particular instance ((and_Hillary_Clinton,$\rightarrow$)) from those that are more closely associated with the predicate at hand ((and_his_wife,$\rightarrow$)).

In order to select the subset, the patterns are ranked in a two-step process. First, a degree measure, $Pos(p, R_k)$, considers the connectivity with several entity instances to support a good pattern instance. Pattern instances are ranked and a minimum support $\tau^p_{support}$ is defined to avoid the aforementioned instance bias. In the second step, the accuracy of the candidates is estimated using previously acquired entity instances. A second parameter τ^p_{acc} defines the minimum accuracy that is expected to consider patterns for further evaluation. In our experiments, this threshold is $\tau^p_{acc} > 0.5$ in order to consider patterns that are linked to more correct entities than negative ones.

$$Pos(p, R_k) = degree(p, R_k) = \left\| \left\{ (p, e) : e \in E_k^{t-1} \right\} \right\| \tag{1}$$

$$Neg(p, R_k) = \left\| \left\{ (p, e) : e \in E_j^{t-1}; \text{ where } j \neq k \right\} \right\| \tag{2}$$

$$Acc(p, R_k) = \frac{Pos(p, R_k)}{Pos(p, R_k) + Neg(p, R_k)} \tag{3}$$

Algorithm 3 SelectCandidatePatterns

```
**Function** $SelectCandidatePatterns(\tilde{A}_p, R_k, E^t_j)$
**Input:** $\tilde{A}_p$ the pool of candidate pattern instances, $E^t_j$ extracted entities until iteration $t$ for all predicates $j$
**Parameters:** $N^p_l$ the max number of candidates
**Output:** $P^t = \{\}$ the set of candidates at iteration t

$r$ : generates random numbers $r : r \in [0, 1]$
$\tilde{A}^t_p \leftarrow order(\tilde{A}^t_p, Pos(p, R_k))$

while $(\|P^t\| < N^p_l \wedge \|\tilde{A}^t_p\| > 0)$ do

  if $p \in P_k \wedge r > p_{rep}$ then
    {Skip repeated patterns}
    $\tilde{A}_p \leftarrow \tilde{A}_p - p$
  else if $(Pos(p, R_k) > \tau^p_{support} \wedge Acc(p, R_k) > \tau^p_{acc})$ then
    {Select frequent, accurate patterns, and sometimes repeated}
    $P^t \leftarrow P^t \bigcup p$
    $\tilde{A}_p \leftarrow \tilde{A}_p - p$
  end if

end while
```

The pool of patterns instances is maintained along iterations and its management resembles the front of exploration in a web crawler. As noted by [28], after some iterations the candidate pattern pool will saturate with previous pattern instances, which means that the same part of the graph is explored. The simplest solution consists of using only new candidate instances. Due to the iterative nature of the algorithm, patterns are evaluated with limited evi-

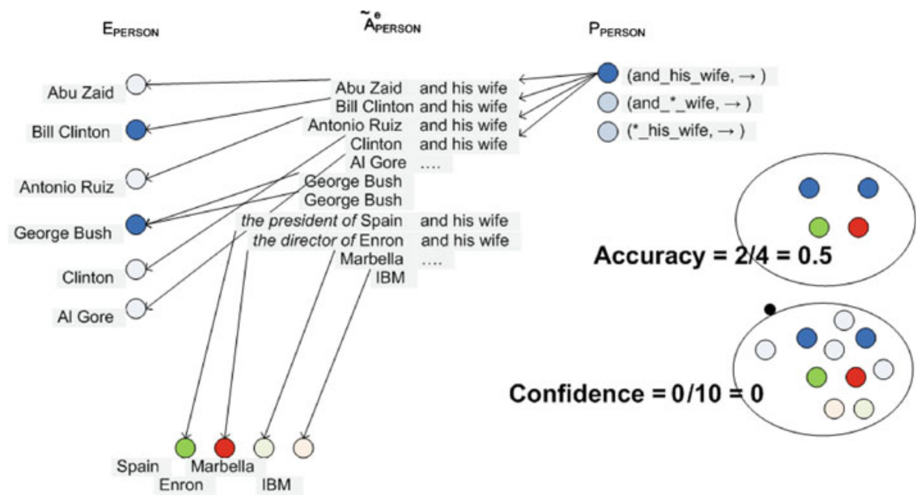


Fig. 2 The process of building the graph: Pattern expansion

dence and it is beneficial to re-evaluate patterns periodically. On the other hand, it is required to consider new patterns to explore new connections in the graph and acquire new instances. In order to enable this behavior, a threshold p_{rep} determines the probability that a repeated pattern instance is considered again.

3.3.4 Evaluate candidate pattern

Candidate patterns are evaluated by querying the document collection again and retrieving associated entity instances. Figure 2 depicts the evaluation of a candidate pattern which first locates pattern mentions and then finds entities mentions (old ones but also new ones). We use two evaluation measures to evaluate patterns, Accuracy (as defined above in 3) and Confidence (Eq. 5). Note that this step is not redundant, as both measures use the new sample obtained from the query with figures for positive, negative, and yet unknown entities.

$$Unk(p, R_k) = \left\| \left\{ (p, e) : e \notin E_j^{t-1} \right\} \right\| \tag{4}$$

$$Conf(p, R_k) = \frac{Pos(p, R_k) - Neg(p, R_k)}{Pos(p, R_k) + Neg(p, R_k) + Unk(p, R_k)} \tag{5}$$

An additional threshold is set on confidence (τ^{p}_{conf}) to discard ambiguous patterns or those for which the algorithm cannot assess a strong association yet. In other words, it controls the aggressiveness of the acquisition process. Patterns instances that are accurate and confident are added to the definitive list P_k and removed from the pool. Patterns that do not meet the accuracy threshold are removed; however, those patterns where the confidence is below the threshold are stored for evaluation in future iterations.

3.4 Entity expansion

Entity Expansion is dual to the process of Pattern Expansion, the process is to a large extent analogous (see Fig. 3 interpreted from right to left).

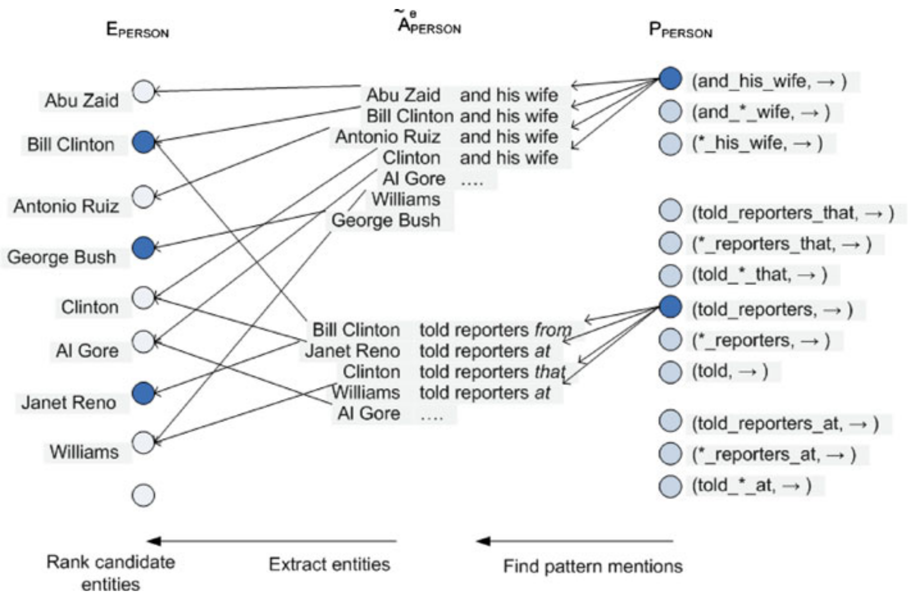


Fig. 3 The process of building the graph: Entity expansion

3.4.1 Find entity mentions

Acquired pattern instances are used to query and filter the text collection in search of new entity mentions.

For example, for the pattern `p(and_his_wife.→)` we generate a phrase query `"and his wife"`. The filter `NERegExps\s+and\s+his\s+wife` where `\s+` stands out for one or more whitespace characters selects only those documents that are consistent with a named entity to the left of the pattern. All the candidate entity instances located are added to a candidate pool.

3.4.2 Select candidate entities

The selection of candidate entity instances is simpler and is carried out in one step based on the degree of connectivity to pattern instances. Entity instances are enforced to be assigned just to one entity class.

3.4.3 Evaluate entities

Their evaluation starts by querying the document collection with candidate entity instances and retrieving patterns. The confidence of a seed is measured in terms of a probabilistic Noisy-OR model [1]. As an entity like `Al Gore` co-occurs with a larger number of patterns that are associated with the predicate PERSON(X), the confidence on being a person increases.

$$Conf_{slot}(e, R_k) = 1 - \prod_i (1 - Conf_{pattern}(p_{i,slot}, R_k))) \tag{6}$$

In contrast, a different aggregation (probabilistic AND) is used to combine evidence from left and right patterns. The motivation is to correct the bias introduced by particular queries.

A correct entity instance should co-occur closed to left and right patterns. Entity instances with a confidence over the threshold τ^{e}_{conf} are considered correct.

$$Conf_{NE}(e, R_k) = Conf_{\leftarrow}(e, R_k) * Conf_{\rightarrow}(e, R_k) \tag{7}$$

Note that it is necessary for the feasibility of the algorithm to assign each NE instance to a single class. That is to say, each NE can only have a sense in this algorithm, and therefore, there exists exclusivity among classes. For example, instances like `Madrid` may be mentioned in the context of a LOCATION (city), a PERSON (surname), or even an ORGANIZATION (in reference to Real Madrid) but a much larger name has only one sense, especially in the scope of a single domain.

Although this seems like a large simplification, it helps to reduce the number of initial seeds and to avoid semantic drifting, the process of acquiring instances of a related class because of semantic relatedness. By means of the evaluation functions, the unambiguous instances are preferred to drive the bootstrapping process and, once assigned to a class, they cannot be assigned to another.

On the other hand, examples that have been acquired for one entity class are used as counter-examples for the rest of the classes. The process is carried out in parallel for all classes at the same time so they compete to acquire a coherent meaning for names. That supposes a counter-training characteristic of the algorithm.

3.5 Efficiency considerations

3.5.1 Delete old candidates

In principle, it would be possible to store all the arcs in the graph that have not been evaluated during an iteration. In practice, the distribution of arcs is long tailed and, after a large number of iterations, there would be a large number of low-frequency candidates in pools $\tilde{A}^{p}_{k}$ and $\tilde{A}^{e}_{k}$. These infrequent candidates slow down the selection of promising ones but would not be evaluated since selection is based on frequency. We have decided to implement a deletion policy that periodically trims the long tail of the pools. Periodically, after performing t_w iterations, candidates which have frequency one are removed. Because the first iteration may be more sensitive, an additional offset (t_{offset}) may be specified to perform deletions at iterations numbered $((N * t_w) + t_{offset})$.

3.5.2 Reuse queries for evaluation and exploration

On the other hand, the exploration of the collection is based on queries. This choice is motivated because full scanning in the collection would not be practical when we deal with a large collection and only few seeds or patterns per iteration. Another consideration for efficiency is the number of queries that we issue for the acquisition of mentions and their contexts. With a proper management of the textual contexts in the pool, it is possible to intertwine pattern expansion and entity expansion to reduce the number of queries. In fact, finding entity mentions must be performed prior to pattern evaluation as well as finding pattern mentions and is a required step to evaluate candidate entities.

Though our current experiments use indexed local collections in order to retrieve mentions and explore a new set of seeds in each interaction, it is especially interesting if the query engine is an external service which usually imposes a limit on the number of queries issued per day.

Table 4 Outline of parameters

Name	Description	Range	Experiments
General			
w_l	Window length for patterns	1–	3
t_w	Time window before cleaning the candidate pool	1–	200
t_{offset}	Time offset before cleaning the candidate pool	0–	100
Patterns instances			
N_l^p	Number of max. pattern instances by iteration	1–	40
N_d^p	Max number of documents per query	1–	400
$\tau_{support}^p$	Threshold support	2–	2
p_{rep}	Probability of select repeated	0.0–1.0	0.01
τ_{acc}^p	Threshold accuracy	0.5–1.0	0.5
τ_{conf}^p	Threshold confidence	0.0–1.0	0.1
Entity instances			
N_l^e	Number of max. entity instances by iteration	1–	40
N_d^e	Max number of documents per query	1–	400
$\tau_{support}^e$	Threshold support	2–	2
τ_{conf}^e	Threshold confidence	0.0–1.0	0.1

Table 4 summarizes all the parameters of the algorithm as well as their range of values. Despite the number of parameters, only a few of them have a large influence on results. Patterns too short would be uninformative but too long ones are infrequent, so we decided to fix it on $w_l = 3$. The threshold on accuracy τ_{acc}^p would trade between the precision of lists and their length. On the other hand, a large number of documents per query is desired but, considering all matches, is impractical. We found 400 documents a practical limit for most queries.

4 Experiments and results

We have carried out experiments for acquiring NE resources in two different western languages, Spanish and English. There are significative grammar differences between the two chosen languages like the word order in the sentences, genres, etc. Furthermore, the Spanish language is more flexible than the English language in the syntactical construction of the sentences, increasing the difficulty for the automatic NERC systems.

For Spanish, we have also evaluated the system with two different text genres, news and Wikipedia. Because these two languages have established evaluation resources used by supervised NERC systems, it is possible to understand the trade-off between this approach and ours, in the context of applications dealing with several languages or different domains.

It is important to highlight that our system does not use annotated corpora during training, but just a large plain collection of documents and only a handful of seeds to bootstrap knowledge that helps in recognizing and classifying NE. These resources may be integrated in rule-based or supervised NERC systems.

Table 5 Description of the bootstrapping document collections

Name	Source	Date	Lang	# Docs	# Tokens
EFE9495	EFE newswire	1994–1995	ES	454,000	136466,251
LAT94GH95	Los Angeles Times	1994	EN	113,005	72410,429
	Glasgow Herald	1995	EN	56,467	25015,576
WIKI-ES	Wikipedia snapshot	10-2006	ES	279,195	69515,149

Table 6 Description of the CONLL evaluation collections

Name	Source	Date	Lang	type	# tokens	# NE tokens
CONLL-ES (2002)	EFE newswire	2000	ES	train	273,037	32,794
				testa	54,837	7,567
				testb	53,049	6,178
CONLL-EN (2003)	Reuters	1996–1997	EN	train	204,567	34,044
				testa	51,578	8.603
				testb	46,666	8,112

We present results for two alternative evaluations, the first directly measures the quality of the name entity instance lists. Direct evaluation of the entity instances lists is desirable but expensive as there are no such large lists of names for persons, locations, and organizations that could be used as a gold standard. A human expert manually judged a sample of the acquired lists to estimate the precision. In contrast, it is not possible to estimate the recall of the acquisition system in a large unannotated collection.

Lists of patterns are even more difficult to judge as they can be associated with several entity classes. In order to evaluate their contribution, both resources are integrated in a very simple rule-based Named Entity classifier. Input is the text with NE mentions boundaries annotated and the goal is to assign the correct class. Lists of patterns are aimed at improving recall as they could signal promising contexts even if mentions are not found the entity list. On the other hand, the main advantage of the indirect evaluation is that it can be easily automatized.

Table 5 summarizes the statistics for the collections that we use to acquire knowledge. Spanish and English news collection has been used in the CLEF evaluation forum. The third one is a snapshot of Wikipedia. The three collections are open-domain and unannotated collections where traditional NERC classes (Person, Location, and Organization) dominate but many other concept classes exist. Table 4 describes the meaning and the values of the parameters used in these experiments.

Annotated corpora from the 2002 and 2003 CONLL shared evaluation [29], summarized in Table 6, were used to evaluate NE classification. The CONLL collections use four different classes: PER (Person), LOC (Location), ORG (Organization), and MISC (Miscellanea). We present results for the evaluation test set (testb) which is also used to evaluate supervised systems.

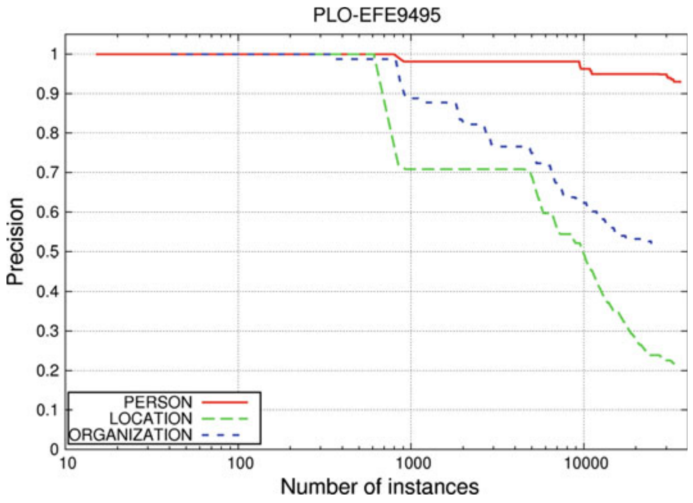


Fig. 4 Direct evaluation of precision for PLO-EFE9495

4.1 Spanish experiments

The first experiment uses the PLO semantic model to learn from the whole EFE-9495 corpus. Each relation has been initialized with a handful of seeds that guaranteed the iterative process to start and be maintained for some iterations. For these experiments, seeds have been selected by a native speaker who knows the collection and its domain. The introspection process takes no longer than one hour, and for each class no more than 40 seeds were used. Only the following rules have been considered:

- Seeds should appear more than once in the collection.
- Avoid ambiguous seeds that could belong to several classes or predicates.
- The set should match entities with both genres because these could affect lexical patterns that do not use any kind of linguistic analysis.

The parameters used for all the experiments were summarized in Table 4. The most critical parameters are the support for patterns ($\tau^{p}_{support}$) and entities ($\tau^{e}_{support}$) which are set to a value of 2. In the case of $\tau^{p}_{support}$, it requires that at least two entities instances co-occur with the same pattern in order to be considered a candidate. This value is justified because a small number of initial seeds (40) that will make it difficult for a common pattern to appear with greater frequency during the initial iterations.

4.1.1 Direct evaluation

The algorithm is able to extract very high precision lists of about 800 elements. After that, precision goes down at different rates (note that the X-axis uses logarithmic scale), but it extracts about 30.000 entity instances per class. Results of the direct evaluation are shown in Fig. 4 and summarized in Table 7. Precision for the Person predicate is fairly high and resulting dictionaries are directly useful. For the Location and Organization classes, the head of the lists achieves reasonable quality but the tail could be fairly improved.

During the direct evaluation of the PLO experiment, we identified some common errors due to semantic drifting. The Organization class acquires entity instances that are part of

Table 7 Direct evaluation of precision for PLO-EFE9495

No. of instances	Precision			Mean
	PER	LOC	ORG	
100	1.0	1.0	1.0	1.0
500	1.0	1.0	0.988	0.996
1,000	0.981	0.709	0.888	0.859
2,000	0.981	0.709	0.835	0.841
5,000	0.981	0.690	0.749	0.806
10,000	0.963	0.497	0.624	0.694
20,000	0.950	0.280	0.532	0.587
30,000	0.949	0.226	–	0.568
At end	0.929	0.217	0.522	0.556
AvgPrec	0.948	0.527	0.671	0.715
	Total # instances			
	36,316	33,335	24,673	

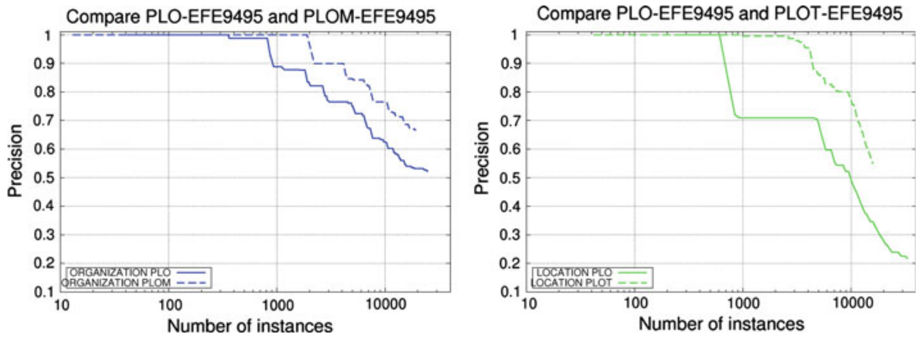


Fig. 5 Comparison of Precision for ORG between PLO and PLOM (*left*) and for LOCATION in PLO and PLOT experiments (*right*)

classes that are not modeled, but they are frequent enough in the corpus. Examples of some of the errors are events (*Juegos Olímpicos—Olympic Games*) or prizes (*Premio Príncipe de Asturias—Príncipe de Asturias Prize*). In order to minimize this kind of error, a different setup included a new MISC entity class (PLOM).

Regarding the errors of the LOCATION predicate, the main source of confusion is formed by sports teams, which in Spanish are often ambiguous with locations, especially city names. These errors are rooted, at least in part, in the assumption that a particular mention is associated with only one class. A different experiment with the PLOT semantic model included a TEAM class.

Both experiments aim to find if using additional entity classes would help to improve the precision of the lists of entity instances. Figure 5 depicts the comparison of results between semantic models and it shows that including additional classes had a positive effect.

Table 8 Name classification in CONLL-ES collection

	Baseline		Entities			Entities+Patterns		
	CONLL	ORG	PLO	PLOM	PLOT	PLO	PLOM	PLOT
P	26.27	–	77.33	**78.85**	78.72	66.12	73.65	66.35
R	56.48	–	54.34	51.53	41.58	57.97	**61.73**	56.62
F	35.86	–	63.83	62.36	54.42	61.78	**67.17**	61.10
Acc	–	39.34	64.04	66.24	62.18	63.17	**71.29**	62.50

Bold values indicate the best results in our experiments

4.1.2 NE classification with EFE-9495 relations

In the indirect evaluation, we solved the task of NE classification as a mean of evaluating the lists of entities instances and patterns. We have used two different baselines for comparison. The first baseline (ORG) assigns the most common class to each name mention, in this collection, that is, ORGANIZATION. The second baseline was proposed in the CONLL shared task and it assigns the most common label for an NE token as it is seen in the training data. This baseline would be roughly equivalent to producing a dictionary from the annotated training data.

We started from perfect NE recognition and each of the entity classes from the bootstrapping algorithm were mapped to their equivalent in the CONLL corpus. For instance, in experiments with the PLOT semantic model, both ORGANIZATION and TEAM were mapped to ORG. The first NE classifier (*entities*) used a naive pure dictionary approach to classify names. Only if the exact mention appears in one of the list of names, it is tagged with the corresponding label. Precision, Recall, and F-measure for this dictionary-based classifier are presented in Table 8 for the CONLL-ES test set. Accuracy results are produced when we add the majority rule. In other words, for those names that appear in no list, the majority class, ORG, is assigned. A second NE classifier (*entities+patterns*) uses the list of acquired patterns. When a name mention is not found in the NE lists, their contexts are matched with the lists of patterns and the name is classified with the most voted class.

Results for the first classifier show a significant improvement over the baselines. Precision is also enhanced by modeling more NE classes (PLOM and PLOT semantic models). On the other hand, we observed that when the modeling assumptions are too restrictive, Recall could be seriously affected. If we compare the performance of the classifiers using the PLO model and the PLOM model, the decrease in Recall is not compensated by the increase in Precision. Therefore, the overall performance of the classifier is not improved. Regarding the use of patterns, results show that there is a consistent improvement in Recall while the global improvement (F-measure) is not always guaranteed. Nevertheless, best results have been achieved when the PLOM semantic model is used and both dictionaries, entities and patterns, are integrated.

4.1.3 NEC with WIKI-ES relations

We have carried out similar experiments by learning the dictionaries from the snapshot of the Spanish version of Wikipedia (WIKI-ES). Our motivation is to compare the system in different genres and determine whether resources acquired in one collection may help when

Table 9 Name classification in CONLL-ES collection with Wiki dictionaries

	Baseline		Entities		Entities+Patterns	
	CONLL	ORG	PLO	PLOM	PLO	PLOM
P	26.27	–	**78.89**	77.82	73.42	73.86
R	56.48	–	47.34	46.64	**53.86**	53.75
F	35.86	–	59.17	58.33	62.14	**62.22**
Acc	–	39.34	61.30	61.01	62.60	**63.05**

Bold values indicate the best results in our experiments

they are applied to a different collection. The experiments were conducted with the same parameter values for the PLO and PLOM semantic models. Classification was evaluated on the CONLL test corpus (news) and results are presented in Table 9. Precision for the classifier using entity dictionaries is comparable to the models trained on EFE corpora. In contrast, Recall is lower than the equivalent experiments using EFE lists. Despite that the size of the acquired relations is similar for both collections, the intersection between the list of NE extracted from Wikipedia and the one extracted from EFE is only moderate. This fact accounts for the lower recall of runs that use only NE compared to EFE.

Regarding patterns from Wikipedia, they help to achieve higher recall, but their contribution is lower than patterns acquired from the EFE collection. Contextual patterns from Wikipedia do not always appear as contextual patterns in news genre. Moreover, patterns that help to classify names may be too specific to that genre or they appear with different frequency across genres.

4.2 Experiments with English NEC

In order to test the usefulness of the approach for other languages, we have performed similar experiments for English. The setting and the parameter values are similar to the Spanish collection. We use the English CLEF collections for NE and pattern acquisition and the CONLL 2003 English collection for the evaluation. Adaptations are required because annotations conventions differ from the Spanish CONLL collection. For example, nationalities and other demonyms are capitalized in English and therefore tagged as NE. On the other hand, days and months are often capitalized, too, but they are not tagged. As our bootstrapping system relies on capitalization to identify NE, we adapted the English experiments to take into account these differences and represented nationalities and temporal expressions as predicates.

We have experimented with two semantic models, PLOM that groups these classes in the single MISC entity class and PLONT which uses two additional separate entity classes, Nationality and Time. Results are reported in Table 10.

As it was already observed for Spanish, the use of acquired patterns in the classifier improves recall figures. Comparison of results for PLOM and PLONT models supports that modeling additional entity classes in the bootstrapping process helps to achieve slightly higher precision.

Nevertheless, the most significant fact is the difference between results for Spanish and English. Our approach performed worse (though not by much) than the CONLL baseline built from the annotated training dataset. In other words, the quality of the automatically generated dictionaries is close to the dictionary created from a large number of human annotations. One of the explanations to these lower results is the degree of NE overlap among the

Table 10 Name classification in CONLL-EN collection with LAT94GH95

	Baseline		Entities		Entities+Patterns	
	CONLL	LOC	PLOM	PLONT	PLOM	PLONT
P	71.91	–	63.30	**66.32**	62.96	65.48
R	50.90	–	40.07	41.87	47.17	**48.90**
F	59.61	–	49.07	51.33	53.93	**55.99**
Acc	–	29.45	60.08	61.97	60.69	**62.61**

Bold values indicate the best results in our experiments

test and bootstrapping collections. They belong to different time spans and different sources. This is supported by the fact that higher recall is obtained for locations, which tend to change slowly over time in news, while this is much lower for persons and organizations. Nevertheless, further research is required to clarify differences between collections and languages.

5 Conclusions

This paper presents a new bootstrapping algorithm of useful semantic resources for multilingual information access applications. The main aim of this proposal is to be able to acquire lists of Named Entities and linguistic patterns from unannotated document collections almost from scratch using only a few seeds provided by human experts. Moreover, a second motivation is to be a language-independent (for western languages) and domain-independent tool which may be used when no manually annotated data are available.

Our experiments in this paper aim to explore the feasibility of this approach as an alternative to other methods that require more labor, whether in annotation or rule construction. The trade-off is interesting in multilingual applications because of the costs associated with annotating training corpora or developing rules in several languages.

The system builds a graph by discovering and selecting frequent patterns that co-occur with entities of a predefined class. Those patterns also help to discover new entities. It expands the number of seeds by a factor larger than 500 with reasonable precision. This is achieved by the combination of dual bootstrapping of several exclusive classes, query-based exploration, throttling, and the incremental assessments of NE and patterns based on previously acquired types. The discovery process is analogous to web crawlers that discover links, giving higher priority to those candidate links that appear often.

Lists of entities acquired by bootstrapping are definitely useful for building NERC systems as this achieves high precision. Additionally, the contextual patterns are also useful to generalize beyond the acquired names and help to improve recall, particularly if they are obtained from the same corpus or a similar genre.

It should also be noted that the system does not assume the use of language-specific NLP tools like POS taggers, chunkers, or parsers which could not be available or tuned to the domain. It is fairly language independent (a solution for western languages), as it relies on a simple regular expression that uses formatting cues like capitalization and general heuristics to locate candidate NE. However, it uses a language-specific list of stopwords to filter and generalize candidate patterns, which in contrast is a common requirement for large collections that include an IR component. The Spanish and English experiments demonstrate the ability of the proposed bootstrapping method to acquire linguistic patterns and recognize named entities using very limited supervision. Several research works have shown that contextual patterns are useful for named entity recognition in western languages such as Italian [33,2], Catalan [5], Portuguese [23], among others. Therefore, if our proposal works on English and Spanish, it should work on those languages. However, the causes of the performance difference warrant further investigation. A future research line should extend the current work to use other simple heuristics that locate NE in other language families (Eastern and Arabic languages) as well as languages where current ones may be not so effective as in the case of German.

Acknowledgments This research work has been supported by the Regional Government of Madrid under the Research Network MA2VICMR (S2009/TIC-1542), by the Spanish Ministry of Education under the project MULTIMEDICA (TIN2010-20644-C03-01) and by the Spanish Center for Industry Technological Development (CDTI, Ministry of Industry, Tourism and Trade), through the BUSCAMEDIA Project (CEN-20091026).

References

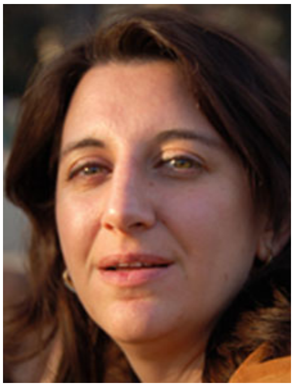
1. Agichtein E, Gravano L (2000) Snowball: extracting relations from large plain-text collections. In: Proceedings of the fifth ACM conference on digital libraries (DL '00), ACM Press, New York, pp 85–94
2. Biggio S, Giuliano C, Poesio M, Versley Y, Uryupina O, Zanoli, R (2009) Local entity detection and recognition task. In: Proceedings of evaluation of NLP and speech tools for Italian (Evalita 2009), Rome, pp 1–8
3. Bikel DM, Schwartz RM, Weischedel RM (1999) An algorithm that learns what's in a name. Mach Learn 34(1–3):211–231
4. Carreras X, Márquez L, Padró L (2002) Named entity extraction using adaboost. In: Proceedings of the 6th conference on natural language learning (CONLL-2002), Toulouse, pp 1–4
5. Carreras X, Màrquez L, Padró L (2003) Named entity recognition for catalan using spanish resources. In: Proceedings of the tenth conference on European chapter of the association for computational linguistics, Association for Computational Linguistics, Sapporo, pp 43–50
6. Collins M, Singer Y (1999) Unsupervised models for named entity classification. In: Proceedings of empirical methods in natural language processing and very large corpora (EMNLP 99), New Brunswick, pp 189–196
7. Cucerzan S, Yarowsky D (1999) Language independent named entity recognition combining morphological and contextual evidence. In: Proceedings of the joint SIGDAT conference on EMNLP and VLC 1999 joint SIGDAT conference on EMNLP and VLC, pp 90–99
8. Li Y, Funk, A (2008) Developing language processing components with GATE version 5 (a user guide). University of Sheffield, Sheffield, last edited February
9. Dorji T, Atlam E, Yata S, Fuketa M, Morita K, Aoe J-I (2011) Extraction, selection and ranking of field association (fa) terms from domain-specific corpora for building a comprehensive fa terms dictionary. Knowl Inf Syst 27:141–161
10. Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A (2005) Unsupervised named-entity extraction from the web: an experimental study. Artif Intell 165(1):91–134
11. Florian R, Ittycheriah A, Jing H, Zhang T (2003) Named entity recognition through classifier combination. In: Proceedings of human language technology conference (HLT-NAACL '03), Edmonton, pp 168–171
12. Harabagiu S, Strzalkowski T (2006) Advances in open domain question answering. Springerg, New York
13. Hearst MA (1992) Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on computational linguistics, Association for Computational Linguistics, Morristown, pp 539–545
14. Ipeirotis PG, Agichtein E, Jain P, Gravano L (2006) To search or to crawl?: towards a query optimizer for text-centric tasks. In: Proceedings of international conference on management of data/principles of database systems (SIGMOD '06), New York, pp 265–276
15. Kazama J, Torisawa K (2007) Exploiting wikipedia as external knowledge for named entity recognition. In: Proceedings of joint meeting of the conference on empirical methods on natural language processing (EMNLP) and the conference on natural language learning (CONLL), Prague, pp 698–707
16. Liu L, Liang Q (2011) A high-performing comprehensive learning algorithm for text classification without pre-labeled training set. Knowl Inf Syst 29(3):727–738
17. Nadeau D, Turney P, Matwin S (2006) Unsupervised named-entity recognition: generating gazetteers and resolving ambiguity. In: Advances in artificial intelligence (LNCS), vol 401, pp 266–277

18. Nadeau D, Sekine S (2007) A survey of named entity recognition and classification. Linguist Investig 30(1):3–26
19. NIST (2008) Automatic content extraction 2008 evaluation plan (ace 2008). Assessment of detection and recognition of entities and relations within and across documents, technical report, National Institute of Standards and Technology
20. On BW, Lee I, Lee D (2011) Scalable clustering methods for the name disambiguation problem. Knowle Inf Syst 31:1–23
21. Pang B, Lee L (2008) Opinion mining and sentiment analysis. Found Trends Inf Retr 2(1–2):1–135
22. Richman AE, Schone P (2008) Mining wiki resources for multilingual named entity recognition. In: Proceedings of human language technologies conference, Association for Computational Linguistics, Columbus, pp 1–9
23. Santos D, Seco N, Cardoso N, Vilela R (2006) Harem: An advanced ner evaluation contest for portuguese. In: Proceedings of the 5th international conference on language resources and evaluation (LREC), Genoa, pp 1986–1991
24. Sarawagi S (2008) Information extraction. Found Trends Databases 1(3):261–377
25. Sekine S, Sudo K, Nobata C (2002) Extended named entity hierarchy. In: Proceedings of the international conference on language resources and evaluation conference (LREC), Las Palmas, pp 1–7
26. Steinberger R, Bruno P, Ignat C (2004) Exploiting multilingual nomenclatures and language-independent text features as an interlingua for cross-lingual text analysis applications. In: Proceedings of the 4th Slovenian language technology conference. Information Society 2004 (IS'2004), Ljubljana
27. Steinberger R, Pouliquen B, Ignat C (2005) Navigating multilingual news collections using automatically extracted information. J Comput Inf Technol 13:257–264
28. Thelen M, Riloff E (2002) A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In: Proceedings of the conference on Empirical methods in natural language processing, Morristown, pp 214–221
29. Tjong-Kim-Sang EF (2002) Introduction to the conll-2002 shared task: Language-independent named entity recognition. In: Proceedings of the conference on natural language learning (CoNLL-2002), Taipei, pp 155–158
30. Tjong-Kim-Sang EF, Meulder FD (2003) Introduction to the conll-2003 shared task: language-independent named entity recognition. In: Proceedings of the conference on natural language learning (CoNLL-2003), Edmonton, pp 142–147
31. Toral A, Munoz R (2006) A proposal to automatically build and maintain gazetteers for named entity recognition using wikipedia. In: Proceedings of the conference of the European chapter of the Association for computational linguistic (EACL '06), Trento, pp 56–62
32. Yangarber R, Lin W, Grishman R (2002) Unsupervised learning of generalized names. In: Proceedings of the 19th international conference on computational linguistics, Morristown, pp 1–7
33. Zanoli R, Pianta E, Giuliano C (2009) Named entity recognition through redundancy driven classifiers. In: In Proceedings of evaluation of NLP and speech tools for Italian (Evalita 2009), Rome, pp 1–5
34. Zitouni I, Florian R (2008) Mention detection crossing the language barrier. In: Proceedings of Conference on empirical methods on natural language processing (EMPNLP), Honolulu, pp 600–609

Author Biographies

César de Pablo-Sánchez got a Ph.D. in Computer Science from the Universidad Carlos III de Madrid (Spain) in 2010. He has been working on several research projects at the intersection of Natural Language Processing and Information Retrieval. His research interests include multilingual Information Extraction, Sentiment Analysis, and Semantic Search Engines.

Isabel Segura-Bedmar got the European Ph.D. in Computer Science from the Universidad Carlos III de Madrid (Spain) in 2010. Since 2004, she has been with the Advanced Databases Group in the Computer Science Department, Universidad Carlos III de Madrid where she is currently teaching data structure and algorithms. Her main area of research interest is the application of Information Extraction Techniques to the pharmacological domain, in particular, the extraction of drug-drug interactions and the detection of drug targets from biomedical texts.



Paloma Martínez received the degree in Computer Science and the Ph.D. degree in Computer Science from the Universidad Politécnica de Madrid (Spain) in 1992 and 1998, respectively. Since 1992, she has been with the Advanced Databases Group in the Computer Science Department, Universidad Carlos III de Madrid, where she is currently teaching database design and management. Her research lines are human language technologies (multilingual information extraction and retrieval in several domains, question answering, name entity recognition, and temporal information management) as well as web accessibility.

Ana Iglesias-Maqueda is a member of the faculty of the Computer Science Department of Carlos III University of Madrid, since February 2006. She obtained a degree in Computer Science from Carlos III University of Madrid in 1999, and her Ph.D. in Computer Science from Carlos III University of Madrid (UC3M) in 2004. Since 2000, She worked at the Advanced Databases Group in the Computer Science Department at Carlos III University of Madrid. Her research interests include Accessibility, Database Design and Advanced Database Technologies, Adaptive Intelligent Educational Systems, Natural Language Processing, and Information Retrieval.