Information Extraction from the Web: An Ontology–Based Method using Inductive Logic Programming

Rinaldo Lima, Hilário Oliveira, Fred Freitas
Informatics Center, Federal University of Pernambuco, Recife, Brazil
{rjl4,htao,fred}@cin.ufpe.br

Bernard Espinasse, Laura Pentagrossa
LSIS, Aix Marseille University, Marseille, France
{bernard.espinasse, laura.pentagrossa}@lsis.org

Abstract – **Relevant information extraction from text and web pages in particular is an intensive and time-consuming task that needs important semantic resources. Thus, to be efficient, automatic information extraction systems have to exploit semantic resources (or ontologies) and employ machine-learning techniques to make them more adaptive. This paper presents an Ontology-based Information Extraction method using Inductive Logic Programming that allows inducing symbolic predicates expressed in Horn clausal logic that subsume information extraction rules. Such rules allow the system to extract class and relation instances from English corpora for ontology population purposes. Several experiments were conducted and preliminary experimental results are promising, showing that the proposed approach improves previous work over extracting instances of classes and relations, either separately or altogether.**

Keywords— Ontology-based Information Extraction; Ontology Population; Inductive Logic Programming;

I. Introduction

The Web has become the major source of information, bearing the potential of being the world's greatest encyclopaedic source of all the news, data, etc. It brings up the interesting idea of converting this sheer volume of unstructured textual data into useful information available for everyone. However, accurate information extraction from web pages is an intensive and time-consuming task, which requires important background knowledge. Thus, the development of efficient and robust information extraction systems is a big challenge. In order to be accurate, such systems have to exploit semantic resources (e.g. ontologies) to take into account this background knowledge. Recently, Ontology-Based Information Extraction (OBIE) has emerged as a subfield of Information Extraction in which ontologies are used by the extraction process and the output is generally presented through an ontology. Furthermore, to be more quickly developed and adaptive to other domains, such systems also have to be based machine learning techniques.

The main goal of Information Extraction (IE) is recognizing and extracting certain types of information from natural language texts. The decision to leave out irrelevant information is a conscious one, and it reduces the difficulty associated with the task at hand. Because IE deals with natural language sources, it is seen as a subfield of Natural Language Processing (NLP). Two important subtasks in IE are Named Entity Recognition (NER) and Relation Extraction (RE). Machine learning is widely used to approach both subtasks. For NER extraction, the performance results of the state-of-the-art systems are around 90%. On the other hand, extracting relations among entities is still a substantially harder task than NER, and NER systems exhibit considerably lower performance [5].

Entity and relation extraction oppose numerical approaches vs. symbolic ones. Numerical approaches exploit the distributional aspect of data, and use statistical techniques, whereas symbolic ones exploit the structural aspect of data, and use structural information. Numerical methods have been widely used and they are the core learning component of robust, and fully automatic IE systems. However, they provide poor explanations for their results and, as observed in [16], they face some difficulties to grasp relations involving more than two entities. Moreover, they are relatively computationally burdensome and do not scale well with increasing amounts of input data.

On the other hand, symbolic methods can be distinguished between linguistic and machine learning methods. In the former, operational definitions of the elements to be acquired are manually established by linguists, mainly with the help of morpho-syntactic patterns that identify the target entities (terms) or relations [13]. In the latter, the relevant patterns are unknown, but examples of the target terms or relations are used as input for building supervised classifiers.

By adopting a symbolic machine learning approach, this paper presents an OBIE method and its implementation using Inductive Logic Programming (ILP) [7]. The proposed method permits to induce symbolic rules in order to extract, from textual data, instances of classes (entities) and relations between them as ontology classes and properties. It assumes that the dependencies among instances, considered here as relational features, can be exploited in an automatic induction of symbolic extraction rules from sentences. This method is mainly based on the principle that the establishment of a relationship among entities in a same sentence can be obtained by the (shortest) path between them in a specific graph-based model. In addition, as our experimental results have demonstrated, the proposed ILP-based approach had a significant improvement over propositional machine learning methods based on kernels and features.

The remainder of this paper is organized as follows: Section II describes some fundamental concepts addressed in this paper, namely OBIE and ILP. Section III presents in details the graph-based model of sentences used in this work, and the different tasks composing the OBIE method using ILP. The Section IV reports and discusses results of several experiments conducted on a reference dataset of texts in order to extract class instances, and relations between class instances. Section V presents related work. Finally, Section VI concludes this paper and outlines future work.

II. Foundations: OBIE and ILP

This section describes some fundamental concepts explored in this paper, namely: Ontology-based Information Extraction, and Inductive Logic Programming.

A. *Ontology-based Information Extraction*

In general, classical IE aims to retrieve certain types of information from natural language text by processing them automatically. For instance, an IE system might retrieve information about economical indicators of countries from a set of web pages while ignoring other types of information.

OBIE can be defined as the process of identifying in text, relevant concepts, properties, and relations expressed in an ontology [14]. In general such ontology is a domain ontology, which represents the domain of the application and captures the domain experts' knowledge. Ontologies contain concepts arranged in class/subclass hierarchies (e.g. a University is a type of Institution), relations between concepts (e.g., a University has a Campus), and properties. OBIE normally takes place by specifying a domain ontology for the domain targeted by an IE system which attempts to discover individuals for classes and values for properties.

Different OBIE systems have already been proposed. In some of such implementations, the OBIE system is part of a larger system. The more popular IE methods used in OBIE systems are: linguistic rules, represented by regular expressions; gazetteer lists, classification techniques, and construction of partial parse trees.

One of the most important potentials of OBIE resides in its ability to automatically generate semantic contents for the Semantic Web. The Semantic Web has the goal of bringing meaning to the Web, creating an environment where software agents, roaming from page to page, can carry out sophisticated tasks for end- users [14].

B. *Inductive Logic Programming*

The Inductive Logic Programming (ILP) framework uses first order clauses as a uniform representation for examples, background knowledge (BK) and hypotheses [7].

According to De Raedt [3] there are two main motivations for using ILP. The first one is that it overcomes the representational limitations of attribute-value (propositional) learning systems that employ a table-based example representation. In this formalism, the representation of the examples to be learned from correspond to rows in a table, and the features to columns, in which a single value is assigned to each one of the attributes. The second motivation is that it rather employs a declarative representation, which means that hypotheses are understandable and interpretable by humans. Moreover, by using logic, ILP systems can exploit background knowledge (BK) in its learning (induction) process. For instance, such BK can be expressed in the form of auxiliary predicate definitions provided by the user.

The aim in ILP is to find a definition of the target relation p that is both *consistent* and *complete*. Informally, the ultimate goal of ILP is to explain the entire set of positive examples and none of the negative ones. More formally, as introduced by Muggleton [9], given:

- a set of examples $E = E^+ \cup E^-$, where E^+ contains positive and E^- negative examples, and;
- background knowledge *BK*.

the task of ILP is to find a theory T such that:

- $\forall$ e $\in E^+$: BK $\wedge T \models$ e (T is *complete*), and
- $\forall$ e $\in E^-$: BK $\wedge T \not\models$ e (T is *consistent*).

One of the main advantages of ILP over other statistical machine learning algorithms is that not only the learned patterns are expressed in a symbolic form, which is more easily interpreted by a knowledge engineer, but also allows the integration of considerable amount of prior knowledge as part of the solution to the problem. Moreover, according to Patel et al. [10], when compared with a handcrafting rule approach, an ILP-based method can provide a complete and consistent view of all significant patterns in the data at the level of abstraction specified by the knowledge engineer. Thus, an ILP-based approach enables the discovery of rules that could be missed by the domain expert, making also possible to scale the rule development to a larger training dataset.

Compared to other propositional rule learning techniques as decision trees and decision lists, ILP is more powerful in several aspects, according to [15]:

- ILP uses an expressive first-order rule formalism enabling the representation of concepts and hypotheses that cannot be represented in the attribute-value framework of traditional machine learning.
- ILP facilitates the representation and use of background knowledge that broadens the class of problems for which inductive learning techniques are applicable.
- Many techniques and theoretical results from computational logic can be used and adapted for the needs of inductively generating from specific observations and BK.

Most of existing ILP implementations like Aleph[1] and GILPS [12] are implemented in Prolog language and, therefore, they impose the following typical restriction to the way of how BK (in terms of predicates or rules), and examples are represented: (i) BK is restricted to Prolog clauses, which are in the form *head:- body$_1$, body$_2$, ..., body$_n$*. Thus, the head is implied by the body clauses, and (ii) E^+ and E^- are restricted to ground facts.

III. An ILP-based Method for OBIE

The method for OBIE proposed in this paper relies on both a domain ontology and an ILP learning component that induces symbolic extraction rules. These rules are used to classify/extract instances of classes and relations between them in a domain ontology. For that, our method employs a relational model of sentences based on dependencies among instances. Such dependencies are considered here as *logical predicates,* which can be exploited by an automatic induction process. This method is based on the principle that the establishment of a

[1] http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html

relationship between two entities in same sentence can be obtained, for example, by a path between them in a dependency graph, which encode grammatical relations between phrases or words [8].

This section first presents the graph-based model of sentences in our approach, which relies on a dependency graph as structural building blocks for rules induction in ILP. Then an overview of the method is presented, covering the specific tasks starting from a natural language pre-processing, followed by the symbolic extraction rules generation with ILP and the application of such rules for extracting instances of classes and relations, up to the population of a domain ontology. Each one of these tasks are presented in detail in the following subsections.

A. A Graph-based Model of Sentences for Rule Induction

The proposed representation that supports our OBIE method consists of a *graph-based model* of sentences. In this model, a simple relationship can be specified between conceptual entities (instances of classes and relations): each major phrasal constituent (nominal and verbal chunking) in a sentence are considered as a candidate instance for extraction. In other words, all phrases that express tokens or chunking constituents are potentially referencing real-world concepts defined by a domain ontology. Moreover, the relational representation of the syntactic structure of a sentence *S* provided by our graph-based model is defined as the mapping *G: sentence → tuples of relations*.

This model is based on a dependency analysis that consists of generating the typed dependencies parses of sentences from *phrase structure* parses, a.k.a. constituent parsing and produces a *dependency graph* [8]. This directed graph is the result of an all-path parsing algorithm based on a dependency grammar [6] in which the syntactic structure is expressed in terms of dependency relations between pairs of words, a *head* and a *modifier*. This relation defines a dependency tree, whose root is a word that does not depend on any word.

We have adopted the typed dependencies proposed in [8] and named *Stanford dependencies*. It is worth noticing that typed dependencies and phrases structures are different ways of representing the inner structure of sentences, in which a phrase structure (constituent) parsing represents the nesting of multi-word constituents, whereas a dependency parsing represents dependencies between individual words. In addition, a typed dependency graph labels dependencies with grammatical relations, such as *subject* or *direct object*.

Different variants of the Stanford typed dependency representation are available in the dependency parsing system provided with the Stanford parser [8]. We adopt the *collapsed tree* representation, where dependencies involving prepositions, conjuncts, as well as information about the referent of relative clauses, are collapsed to get direct dependencies between content words. This collapsed representation can simplify the relation extraction process.

In addition, the proposed graph-based model exploits *chunking analysis*, which is useful to define entity boundaries, and the head constituents of nominal, verbal and prepositional phrases. For example, consider the sentence "*Mary is reading a book on Semantic Web*". Fig 1 shows the head tokens of this sentence obtained after a previous chunking analysis. Usually, verbal phrases are possible candidates for relations, and nominal ones, can represent an entity or an instance of a class.

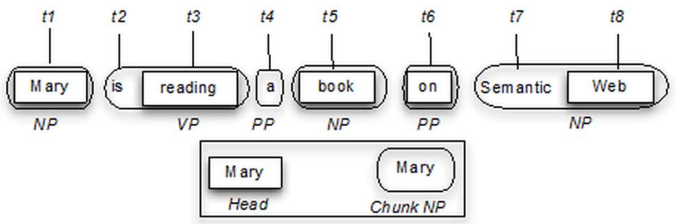

Fig. 1. Chunking analysis and head tokens of the sentence

In our method, dependencies are considered as relational features that can be exploited in the automatic induction of symbolic extraction rules from sentences. Indeed, we develop an ILP-based formulation of this IE problem and show how to cast this problem in it. In addition, the proposed approach is based on the observation that, when learning about properties of objects in relational domains, feature construction can be guided by the structure of individual objects [Raedt, 2010], which can be used for asserting relationships between two (or more) class instances in the same sentence.

Fig. 2 shows the graph-based model of a sentence obtained by: (i) a dependency analysis with *collapsed dependencies* (e.g. *prep-on*) according to the Stanford dependency parser, (ii) a *chunking analysis* (head tokens in bold), (iii) the sequencing of tokens in a sentence (*NextToken* vertices), and (iv) *morpho-syntactic features* as nodes attributes (arrows in gray color).

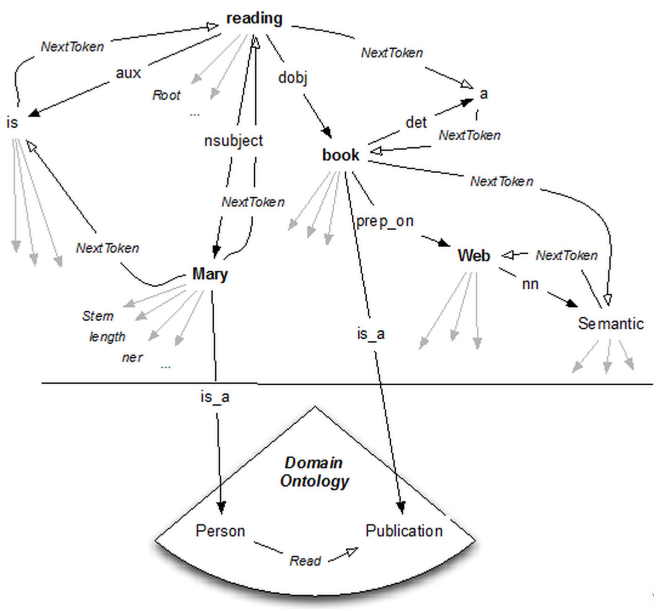

Fig. 2. Graph-based model of the sentence: “Mary is reading a book on Semantic Web”.

This graph-based model can be expressed by a set of binary relations or predicates. For the same sentence in Fig. 2, here is a list of some binary relations:

nsubj (reading, Mary)	*nextToken (Mary, is)*
aux (reading, is)	*nextToken (is, reading)*
det (book, a)	*length (Mary, 4)*
head (Mary, NP)	*ner(Marry, person)*

The graph-based model represents a collection of binary relations, and their arguments can be enriched with additional constraints on the types of the arguments. These additional binary relations are used in the ILP induction process to link terms in a sentence with classes and relations from a domain ontology. For example, if the predicate to be learned is *read (X, Y),* or putting it as ontological terms, the object property *read(X, Y)*, then the first argument *X* should be an instance of the *Person* class, and the second one *Y* should be an instance of the *Publication* class in the domain ontology. To sum up with, instances of classes and relations can be viewed, respectively, as nodes and edges in our model. Each node can have many attributes, e.g., the ontology class label which it belongs to.

In the present work, the task of identifying the labels of candidate instances of classes and relations is defined as the *target predicate* in our learning problem formulation. Concretely speaking, we learn such target predicates as a combination of base predicates on several constituents of the sentence.

Most previous research work in Relation Extraction and Ontology Population (OP) [2] has only considered attribute-value features, or propositional features derived from input text data [11], [5], [4]. On the contrary, we rely on a first-order logic representation of examples which provides a much richer representation formalism, allowing classification of objects whose structure is relevant to the classification task [15].

B. *System Architecture*

As previously mentioned, an inductive logic programming-based framework has been adopted as the core component for machine learning in our architecture. The main reason is that this learning component can generate extraction rules in symbolic form. As a result, such rules may be fully interpreted by a knowledge engineer, which could refine them in a posterior stage of the rule induction task, aiming at improving the whole extraction process. Moreover, symbolic rules can be automatically converted into other rule formalisms, such as SWRL, the proposed rule language for the Semantic Web.

In general terms, the method consists of a supervised approach to automatically infer efficient and expressive extraction patterns of complex terms or syntactic relations from examples (sentences). As a result, the induced rules can be applied on an unseen set of pre-processed documents in order to extract instances for populating a domain ontology.

The main goal of this work is to propose and assess our OBIE approach. For that, we developed a modular architecture for OBIE in which we integrate several system components in a pipeline architecture. Fig. 3 shows the functional architecture of the developed prototype.

The OBIE process is performed in two distinct phases. First, a theory (a set of rules) is induced from a given annotated corpus (*Rule Learning module*). This phase correspond to the *Learning phase* in Fig. 3, which generates extraction rules from an annotated learning corpus by induction. In the second phase, i.e., the *Exploitation phase,* the learned theory is then applied in pipeline to extract ontology instances from new tagged document. In both above phases, a previous preprocessing stage take place in which several Natural Language Processing (NLP) tools are used (*Natural Language Preprocessing module*), followed by an automatic representation of the examples in our approach (*Background Knowledge Generation module*).

In the following sections, we explain each component of the architecture shown in Fig. 3 in more detail.

C. *Text Processing*

The proposed OBIE system consists of automatically inducing extraction patterns that discover both hypernymy relations (*is-a* relationships) and other types of relations between two terms. To do so, we need a representation formalism that expresses these patterns in a simple and effective way.

Accordingly, we defined a set of lexico-syntactic features generated by the natural language preprocessing component in our architecture. These features are the building blocks that compose the BK that will be used later by the learning component. Thus, three NLP tasks are performed: lexico-*syntactic analysis*, *dependence parsing*, and *chunking analysis*. We use the Stanford CoreNLP[2] for carrying out the following sequence of NLP subtasks: *sentence splitting*, *tokenization*, *Part-of-Speech tagging*, *lemmatization* (which determines the base form of words), *Named Entity Recognition* (NER), and *dependency parsing* on the input corpus. Finally, we rely on the OpenNLP[3] tool for the chunking analysis. In addition, we employed the simple heuristic of considering the rightmost token in a chunking (nominal or verbal) as its head element.

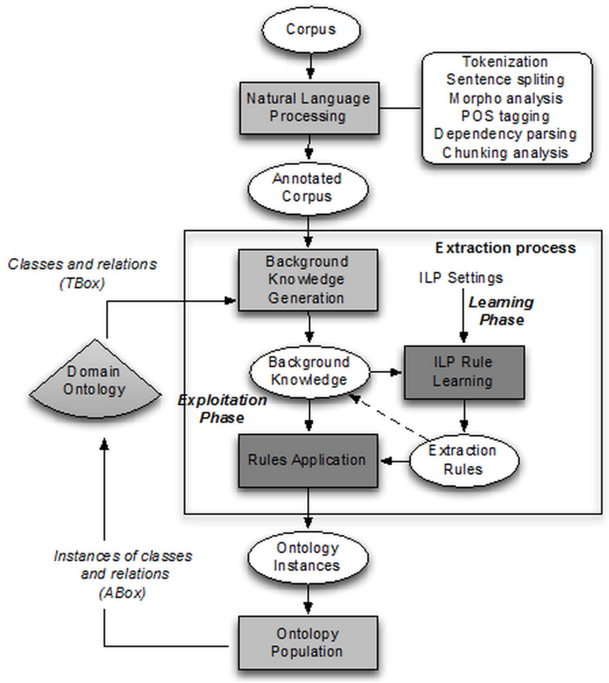

Fig. 3. Overview of our OBIE process using ILP

D. *Background Knowledge Generation*

After the previous *text preprocessing step*, we carry out the critical task on identifying, extracting, and appropriately representing relevant background knowledge (BK). This process is not trivial because, without it, the ILP features that make it

[2] http://nlp.stanford.edu/software/corenlp.shtml
[3] http://opennlp.apache.org

different from traditional learning algorithms cannot be truly exploited.

Previous research has shown that shallow semantic parsing can provide very useful features in several IE related tasks [4]. Accordingly, we explore the features listed in Table I, which constitute the BK in our solution. These features provide a suitable feature space for the classification problem of ontological instances, describing each token in the corpus. We illustrate in Table I the BK in logical predicates that characterizes the candidate instance of the *Person* class, "Mary".

Differently from other machine-learning approaches for OP that use feature vectors for representing *context windows* (*n* tokens on the right/left of a given word *w* in a sentence), we use the *next/2* predicate which relates one token to its immediate successor in a sentence, along with additional predicates that compose our BK (Table I). All predicates are expressed in Prolog syntax.

TABLE I. PROLOG PREDICATES FOR THE TOKEN "MARY" (T_1)

Predicates Generated	Meaning
token (t_1)	t_1 is the token identifier
t_dep (nsubj, t_3, t_1)	there is a noun subject dependency between token t_3 and t_1
t_next (t_1, t_2)	token t_2 follows token t_1
t_stem (t_1, "Mary")	the stemming of the token t_1 is "Mary"
t_length (t_1, 4)	t_1 has length of 4
t_orth (t_1, upperInitial)	t_1 has an initial uppercase letter
t_type(t_1, word)	t_1 is a word
t_pos (t_1, nnp)	t_1 is a singular proper noun
t_ner (t_1, person)	t_1 is a person entity
t_root (t_3)	t_3 is the root of the dependency graph
t_bigposbef (t_n, ...)	POS tag bigram of the tokens after t_n
t_bigposaft (t_1, vbz-vbg)	POS tag bigram of the tokens before t_1
t_trigposbef (t_n, ...)	POS tag trigram of the tokens before t_n
t_trigposaft (t_1, vbz-vbg-dt)	POS tag trigram of the tokens after t_1
t_isHeadNP (t_1)	t_1 is the head of the nominal chunking
t_isHeardVP (t_n...)	t_n is the head of the verbal chunking
t_isHeardPP (t_n...)	t_n is the head of the prepositional chunking
t_ck_tag (t_1, NP)	t_1 is part of a nominal chunking

E. Rule Learning

In the last step from our method, we had to adjust some parameters of the GILPS system for the task at hand. One of such parameters defines the *language bias*, which delimits and biases the possibly huge hypothesis search space. In GILPS, this is achieved by providing appropriate *mode declarations*.

Mode declarations characterize the format of a valid hypothesis (rule). They also inform both the *type*, and the *input/output modes* of the predicate arguments in a rule. There are two types of mode declarations in GILPS: *head* and *body*. Mode head declarations (*modeh*) state the target predicate, the head of a valid rule that the ILP system has to induce, whereas *mode body* declarations (*modeb*) determine the literals (or predicates), which may appear in a rule body. Mode body declarations usually refer to predicates defined in the BK, but they can also refer to the target predicate in the case of recursive theories [13].

In mode declarations, each argument of a predicate has a type and an associated symbol that appears before the type indicator. The symbol " + " means that the argument is an *input argument* (i.e. the argument must be instantiated before the predicate be called); '-' stands for an *output argument*, i.e., the argument does not need to be instantiated before the predicate be called. The symbol '#' denotes a *constant*, which is yielded directly in the hypothesis body. By both declaring typed arguments of the predicates and imposing input/output restrictions, one guarantees that the clauses generated as hypothesis are at least executable by the Prolog engine. Note that these restrictions also reduce considerably the hypothesis space [12].

Another important choice is related to the *engine* parameter in GILPS, since it permits the user to choose the way rules are specialized/generalized, i.e., how the hypotheses space are traversed, top-down or bottom-up manner. For the purposes of this research work, we had chosen the bottom-up engine (ProGolem) available in GILPS, as this engine has demonstrated competitive performance among others ILP systems [12].

F. Rule Application and Ontology Population

Once the OBIE system has learned a set of extraction rules, the *Rule Application* module applies the induced rules on the knowledge base (Prolog factual base) generated from new documents similar to the ones used in the Learning phase. As a result, new instances of classes and/or relations are extracted, and they can be finally integrated into the domain input ontology. For instance, considering the sentence given in the introductory section of this paper, the obtained extraction rules could identify the following instances of binary relations:

is_a(Mary, Person), is_a (book, Publication), read (Mary, book)

These new relations are used for populating the domain ontology, i.e., after converting the above Prolog predicates into OWL assertional axioms. In addition, we perform a redundancy checking in order to avoid repeated instances in the domain ontology.

IV. EXPERIMENTAL EVALUATION

In this section, we present and discuss the results of experiments conducted on a reference corpus. At first, we describe the corpus also presenting the frequency distribution of the instances of classes and relations in it. Next, we discuss the results of our experimental assessment on this dataset concerning the extraction of both class and relation instances. Finally we compare our method with other supervised propositional IE methods evaluated on the same corpus.

A. Dataset Description and Preparation

The experiments reported in this section are based on the TREC dataset[4], which consists of articles from WSJ (Wall Street Journal). This dataset have been annotated for named entities and relations, containing 1,441 sentences with 5,349 entities, namely, 1,691 people, 1,968 locations, 984 organizations, and 706 miscellaneous names. Each one of the 1,441

[4] http://cogcomp.cs.illinois.edu/Data/ER/conll04.corp

sentences has at least one active relation. Among those sentences, there are 19,080 possible binary relations with the frequency distribution of the positive ones as shown in Table II. This table also shows an example of each relation and the constraints with respect to its two arguments. It worth noticing that most candidate binary relations have no active relations at all; this results in a significant unbalanced distribution between positive and negative examples. Fig. 4 shows a sentence from this dataset, available in a column format with one POS tagged element per line. We used this original tokenization in our experiments.

```
286 Loc  0   O   NNP France   O   O   O
286 O    1   O   POS 's   O   O   O
286 O    2   O   JJ   deputy   O   O   O
286 O    3   O   NN   representativeO   O   O
286 O    4   O   ,    COMMA   O   O   O
286 Peop     5   O   NNP/NNP/NNP     M./Pierre/Brochand     O   O   O
286 O    6   O   ,    COMMA   O   O   O
286 O    7   O   VBD expressed    O   O   O
286 O    8   O   JJ   similar    O   O   O
286 O    9   O   NNS sentiments   O   O   O
286 O    10  O   .    .   O   O   O

5    0    Live_In
```

Fig. 4. Example of a sentence in the TREC dataset: col-2: class label, col-5: POS tags, col-6: word form

TABLE II. BINARY RELATIONS AND THEIR ARGUMENTS TYPES

Relation	Arg-1	Arg-2	Example	# of relations
located_in	LOC	LOC	(Toledo, Ohio)	405
work_for	PER	ORG	(Winter, Court)	401
orgBased_in	ORG	LOC	(HP, Palo Alto)	452
live_in	PER	LOC	(Tvazir, Israel)	521
kill	PER	PER	(Oswald, JFK)	268

Fig. 5 depicts the domain ontology used for storing the instances extracted by our OBIE system.

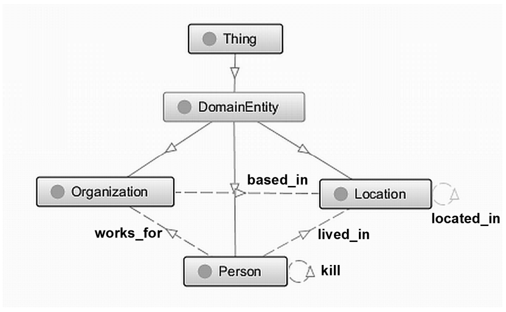

Fig. 5. Domain Ontology used in the OP process

B. *Evaluation Measures and Rule Induction Setting*

Aiming at assessing the effectiveness of our approach, we conducted several experiments on the TREC dataset. The performance evaluation is based on the IR classical measures, i.e., *Precision* P, *Recall* R, and F1-*measure* [1].

In all experiments reported here, we used 5-fold cross-validation that provides unbiased performance estimates of the learning algorithms. Furthermore, although our solution provides a named entity tagger in its preprocessing component, we decided not to use it, in order to have a fair experimental setup for all IE tasks reported in this paper.

C. *Results and Discussion*

In all experiments reported in this section, we adopted the same assumption in [11] that the problem of *phrase detection* is solved, and the entity boundaries are provided by the dataset as input. Thus, our OBIE system needs only to concentrate on the classification task.

We carried out several experiments for evaluating the effectiveness of the proposed method concerning three models, all applied to entities (instances of classes) and relations (instances of relations), namely: (i) *separate*, (ii) *pipeline*, and (iii) *omniscient*.

The first model, separate E_S and R_S, are constructed by training entities and relation classifiers separately, i.e., the entity classifier E_S does not know the labels of relations in the sentence, whereas the relation classifier R_S is not aware of the labels of its entity arguments, either.

The pipeline model for entities, denoted as E_P (for entity classifiers) is obtained by first training a separate relation classifier R_s, in which its output is then used as additional features for training the E_P classifier. Analogously, the R_P model uses the predictions on the two entity arguments given by a separate entity classifier (E_S) as additional features in its learning process.

In the last model, omniscient, it is assumed that the entity classifier E_O knows the correct relation labels given to it as additional input features. Similarly the relation classifier R_O knows the correct entity labels as well. These additional features are then used in both training and testing. Although this assumption may appear unrealistic, it may reveal how accurate the classifiers can be without this information.

Tables III and IV show the results obtained for all aforementioned models. The classification results achieved by our richer models (E_O and E_P) for entities, the ones with additional features as input, did not take much profit of them, since the score for P, R and F1 are very close to each other. For all relation classifiers, except for the *orgBased_in* relation, the additional feature information decrease the precision of the classifiers, but contributes to better recall scores in all relation classifiers. This can be explained by the fact that the noisy information in the dataset itself can be mitigated by these further clues to the classifiers, and can enable them to cover more examples in this case. Furthermore, in R_P and R_O models, more rules are added to the BK of the ILP rule-learning component. Such additional rules might increase the number of false positives. A further investigation of this problem is our ongoing work.

In the following, we show an induced rule expressed in terms of (number of literals), (positive examples covered), (negative examples covered), and the (rule precision P):

Rule: #Literals=4, PosScore=187, NegScore=19, Prec = 90.8%

located_in(A,B):- t_class(A,loc), t_next(A,B), t_class(B,loc).

This rule classifies an instance of the *located_in* relation in which the good precision score is mainly due to the presence of several phrases similar to "Perugia, Italy", indicating that the first argument (*A*) "Perugia" is followed by (predicate *next*) the second argument (*B*) "Italy", not considering the punctuation symbol between them.

Learning Curves. We performed a further evaluation of the E_S and R_S classifiers in the intent of examining the effect of limited training examples in the learning process by incrementally adding provided subsets as training data. Thus, we conducted 9 experiments in which incremental parts of the training data, corresponding to 10% of the total number of examples are added to the classifier. Considering the learning curves in Fig. 6 (left), one can observe that for LOCATION and PERSON entities, their classifiers obtained a reasonable F1 score with just 10% of the total number of training examples. This corresponds to 30 and 26 extraction rules for LOCATION and PERSON, respectively in the final induced model. In contrast, for the ORGANIZATION entity classifier, more learning examples were needed for attaining the same performance.

TABLE III. RESULTS FOR ENTITY CLASSIFICATION (ALL MODELS)

Model	**LOC**			**ORG**			**PER**		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
E_O	95.9	92.4	94.1	98.7	79.2	87.8	93.7	91.2	92.4
E_P	95.2	92.0	93.5	97.5	76.5	85.7	93.5	89.0	91.3
E_S	96.0	88.4	92.0	97.0	74.4	84.3	94.8	87.5	91.0

TABLE IV. RESULTS FOR RELATION CLASSIFICATION (ALL MODELS)

Model	**located_in**			**work_for**			**orgBased_in**		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
R_O	90.5	78.6	84.0	85.7	86.1	85.8	88.7	82.5	85.4
R_P	91.1	78.0	83.9	87.2	80.8	83.8	91.5	84.0	87.5
R_S	91.2	75.9	82.6	93.1	72.9	81.7	88.4	77.0	82.2

Model	**live_in**			**kill**		
	P	*R*	*F1*	*P*	*R*	*F1*
R_O	87.4	76.9	81.7	92.3	78.0	84.3
R_P	85.7	72.1	78.2	91.5	77.6	83.9
R_S	92.5	67.4	78.0	97.5	73.7	83.8

In Fig. 6 (right), for each relation, the difference among the learning curves is fairly constant, except for the *orgbased*_in and *live_in* relations. In fact, these two relations have as one of its arguments an ORGANIZATION entity, and its lower learning curve performance is explained by that fact.

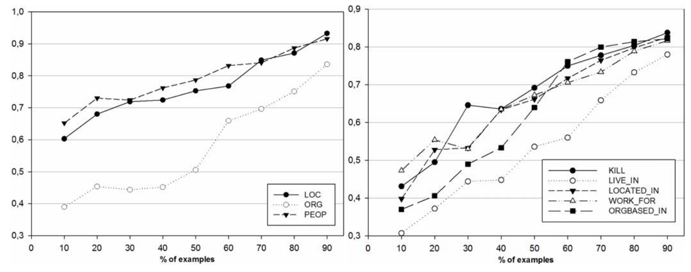

Fig. 6. Learning curves for entity and relations classifiers: E_S and R_S.

Comparative Results. We provide a comparative evaluation of our OBIE method with the best ones proposed in [11] and [5].

For the entity classification comparison, we took the best result in [5], i.e., their M_C classifier. This classifier also assumes that entity boundaries have already been determined, and it corresponds to our E_S model. In [11] the best result for entity classification was obtained by their *Separate w/Inf* model, which uses an additional global inference procedure to produce the final decision. The *Separate w/Inf* assumes that the entity classifier knows the correct relation labels and it equally relates to our E_S model.

The results in Table V suggest that the M_C model has superior performance in terms of F1 score. However, statistical significance tests (*paired Student t-test*) for the difference between F1 scores of the E_S model and the M_C model revealed that there is no significant difference at $\alpha = 0.05$ (95% confidence interval) between them. The same result occurs when we compare the E_S model with the *Separate w/Inf.* A further look at Table V shows that the E_S model is more precise than the other ones, but has a lower recall performance. For some applications in which precision is more desirable than recall, the E_S model could be a good alternative, as it avoids overloading end-users with too many false positives.

On the other hand, considering the relation classifiers, the results in Table VI show that our OBIE method significantly outperforms the others. The main reason for that probably relies on the better sentence representation model employed. In our graph-based model, relationships between terms in a sentence can be represented using a richer formalism of representation that, as it was demonstrated by the experiments reported here, are more expressive than the propositional representation of the related work evaluated, since it applies attribute-value representation models.

TABLE V. COMPARATIVE RESULTS FOR ENTITY CLASSIFICATION (SEPARATE MODELS)

Model	**LOC**			**ORG**			**PER**		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
E_S	96.0	88.4	92.0	97.0	74.4	84.3	94.8	87.5	91.0
M_C	94.2	94.4	**94.3**	91.9	88.5	**90.2**	94.8	96.6	**95.7**
Separate w/Inf	91.8	88.6	90.1	91.2	71.0	79.4	90.6	90.5	90.4

TABLE VI. COMPARATIVE RESULTS FOR RELATION CLASSIFICATION (BEST MODELS)

Approach	**located_in**			**work_for**			**orgBased_in**		
	P	*R*	*F1*	*P*	*R*	*F1*	*P*	*R*	*F1*
R_O	90.5	78.6	**84.0**	85.7	86.1	**85.8**	88.7	82.5	**85.4**
$M_O\|K_{SL}$	79.6	76.0	77.8	76.8	80.0	78.4	74.3	77.2	75.7
Omni w/Inf	61.9	62.9	59.1	79.2	50.3	61.4	81.7	50.9	62.5

Approach	**live_in**			**kill**		
	P	*R*	*F1*	*P*	*R*	*F1*
R_O	87.4	76.9	**81.7**	92.3	78.0	**84.3**
$M_{O\|KSL}$	78.0	65.8	71.4	82.8	81.0	81.9
Omni w/Inf	63.9	57.3	59.9	79.9	81.4	79.9

Limitations. It is worth mentioning that the proposed OBIE method does not consider the class hierarchy when assigning class instances to classes into the ontology, for example. The current system version that implements our approach only populates those classes chosen by the user at the beginning of the population process. Indeed, we intend to overcome these limitations in future work in order to achieve a more robust framework for OP.

V. Related Work

Several machine learning approaches to IE have been proposed.

Patel et al. [10] conducted some experiments using ILP techniques to induce rules that extract instances of various named entity classes. They also reported a substantial reduction in development time by a factor of 240 when ILP is used for inducing rules, instead of involving a domain specialist in the entire rule development process. More recently, for relation extraction between two entities, the authors in [13] propose a multi-agent system, which uses the ILP framework. The learning capability of agents is exploited by training an agent to learn extraction rules from the syntactic structure of sentences. Typed dependencies of the syntactic constituents of sentences provide the background information.

The system proposed in [13], similar to ours, has also used typed dependencies and ILP for RE, but it has some drawbacks: (i) it uses a less expressive propositional learner as the key component for rule induction; (ii) no comparative evaluation is reported; and (iii) its experimental results concern only one relation (*located_in*) which was conducted using a small corpus of 13 Wikipedia pages.

Roth and Yih [11] proposed an entity and relation extraction system based on global inference. In their approach, predictors that identify entities and relations among them are first learned from local information in the sentence. The constraints induced from the dependencies among entity types and relations constitute a relational structure over the outcomes of the predictors and are used to make global inference.

Reference [5] describes a system based on shallow linguistic processing (such as tokenization, POS tagging and lemmatization) that uses kernel methods to perform NER and RE tasks. This system also uses a combination of kernel functions that model two distinct information sources: (i) the global context where entities appear and (ii) the local contexts around the interacting entities. The whole sentence containing the entities (i.e., global context) is used to discover the presence of a relation between two entities. Text windows of limited size centred on the entities (i.e., local contexts) provide clues to identify the roles played by the entities within a relation.

VI. Conclusions and Future Work

We have proposed an ILP-based method for extracting instances of classes and relations from textual data, such as web pages, that mainly relies on shallow syntactic parsing of individual sentences in a document followed by an effective graph-based model of the sentences.

A further important conclusion was that the ILP-based method represents a considerable advantage concerning the inclusion of new BK into the OP process by means of simple predicates provided by the domain expert. Next, we conducted several assessments in order to evaluate the effectiveness of the proposed approach. The obtained results were compared with related research work on relation extraction using the same corpus, showing that the ILP-based approach had a significant improvement over other approaches based on propositional machine learning methods based on kernels and features.

Although we have achieved encouraging results, there is still room for improvement. Indeed, the method presented here currently relies on shallow syntactic parsing of English sentences, which does not take into account semantic aspects relating entities to verbs, like the ones treated in Word Sense Disambiguation or Semantic Role Labelling, for example. Thus, we would like to evaluate the contribution of semantic shallow parsing to OP on other domains. In this case, the desirable ILP characteristics of allowing such an incremental BK would be put to test. Our long term goal here is to fully exploit different types of BK aiming at investigating which kind of BK are more useful on specific domains. Finally, we also intent to extend our approach to n-ary relation extraction.

Acknowledgment

The authors would like to thank the National Council for Scientific and Technological Development (CNPq/Brazil) for financial support (Grant No. 140791/2010-8).

References

[1] R. A. Baeza-Yates, and B. Ribeiro-Neto, Modern Information Retrieval: Addison-Wesley Longman Publishing, Boston, USA, 1999.

[2] P. Cimiano, Ontology Learning and Population from Text: Algorithms, Evaluation and Applications, Springer-Verlag, New York, USA, 2006.

[3] L. D. Raedt, Inductive Logic Programming, Encyclopedia of Machine Learning, 2010, pp. 529-537.

[4] A. Finn, A Multi-Level Boundary Classification Approach to Information Extraction, Phd thesis, University College Dublin, 2006.

[5] C. Giuliano, A. Lavelli and L. Romano, Relation Extraction and the Influence of Automatic NER, ACM Transactions on Speech and Language Processing, vol 5, no. 1, ACM, 2007.

[6] G.-J. M Kruijff. Formal and Computational Aspects of Dependency Grammar: History and Development of DG, Tech. report, ESSLLI, 2002.

[7] N. Lavrac and S. Dzeroski, Inductive Logic Programming Techniques and Application, Ellis Horwood, New York, 1994.

[8] M-C de Marneffe and C. D. Manning, Stanford Dedendencies Manual, 2008.

[9] S. Muggleton, Inductive Logic Programming. New Generation Computing 8 (4): 29, 1991.

[10] A. Patel; G. Ramakrishnan, and P. Bhattacharya. Incorporating Linguistic Expertise Using ILP for Named Entity Recognition in Data Hungry Indian Languages, LNCS, vol. 5989, , Springer Berlin Heidelberg, 2010, pp. 178-185.

[11] D. Roth, and Yih,W. 2007. Global Inference for Entity and Relation Identification via Linear Programming Formulation. In Introd. to Statistical Relational Learning, L. Getoor and B. Taskar, Eds. MIT Press, 2007.

[12] J. Santos, Efficient Learning and Evaluation of Complex Concepts in Inductive Logic Programming, Ph.D. Thesis, Imperial College, 2010.

[13] M. D. S. Seneviratne and D. N. Ranasinghe, Inductive Logic Programming in an Agent System for Ontological Relation Extraction, International Journal of Machine Learning and Computing vol. 1, no. 4, 2011, pp. 344-352.

[14] D. C. Wimalasuriya and D. Dou, Ontology-Based Information Extraction: An Introduction and a Survey of Current Approaches, Journal of Information Science, vol. 36, no. 3, June 2010, pp. 306-323.

[15] J. Fürnkranz, D. Gamberger and N. Lavrac, Foundations of Rule Learning, Springer-Verlag, 2012.

[16] N. Bach and S. Badaskar. A Survey on Relation Extraction. Language Technologies Institute, Carnegie Mellon University, 2007.