Multilingual Open Information Extraction *

Pablo Gamallo[1] and Marcos Garcia[2]

[1] Centro Singular de Investigacin en Tecnoloxas da Informacin (CITIUS),
Universidade de Santiago de Compostela
pablo.gamallo@usc.es
[2] Cilenis Language Technology
Santiago de Compostela
marcos.garcia@cilenis.com

Abstract. Open Information Extraction (OIE) is a recent unsupervised strategy to extract great amounts of basic propositions (verb-based triples) from massive text corpora which scales to Web-size document collections. We propose a multilingual rule-based OIE method that takes as input dependency parses in the CoNLL-X format, identifies argument structures within the dependency parses, and extracts a set of basic propositions from each argument structure. Our method requires no training data and, according to experimental studies, obtains higher recall and higher precision than existing approaches relying on training data. Experiments were performed in three languages: English, Portuguese, and Spanish.

1 Introduction

Recent advanced techniques in Information Extraction aim to capture shallow semantic representations of large amounts of natural language text. Shallow semantic representations can be applied to more complex semantic tasks involved in text understanding, such as textual entailment, filling knowledge gaps in text, or integration of text information into background knowledge bases. One of the most recent approches aimed at capturing shallow semantic representations is known as Open Information Extraction (OIE), whose main goal is to extract a large set of verb-based *triples* (or *propositions*) from unrestricted text. An Open Information Extraction (OIE) system reads in sentences and rapidly extracts one or more textual assertions, consisting in a verb relation and two arguments, which try to capture the main relationships in each sentence [3]. Wu and Weld [19] define an OIE system as a function from a document d, to a set of triples, $(arg1, rel, arg2)$, where $arg1$ and $arg2$ are verb arguments and rel is a textual fragment (containing a verb) denoting a semantic relation between the two verb arguments. Unlike other relation extraction methods focused on a predefined set of target relations, the Open Information Extraction paradigm is not limited

* This work has been supported by projects Plastic and Celtic, Innterconecta (CDTI).

to a small set of target relations known in advance, but extracts all types of (verbal) binary relations found in the text. The main general properties of OIE systems are the following: (i) they are domain independent, (ii) they rely on unsupervised extraction methods, and (iii) they are scalable to large amouts of text [5].

The objective of this article is to describe a heuristic-based OIE system, called ArgOE, which uses syntactic analysis to detect the argument structure of each verb, as well as a set of rules to generate the corresponding triples (or basic propositions) from each argument structure. In our work, an argument structure has a very broad sense, since it includes all those syntactic dependencies headed by a verb except specifiers, auxiliars, and adverbs. So, it includes all main clause constituents: subjects, objects, attributes, and prepositional phrases referring to locations, instrumentals, manners, causes, etc. So, there is no distinction between traditional arguments and adjuncts, both are used to build the argument structure.

Consider for example the sentence:

> *In May 2010, the principal opposition parties boycotted the polls after accusations of vote-rigging.*

First, our OIE system detects the argument structure of the verb *boycotted* in this sentence: there is a *subject*, a *direct object*, and two prepositional phrases functioning as verb *adjunts*. Then, a set of basic rules transform the argument structure into a set of triples:

("the principal opposition parties", "boycotted", "the polls"),
("the principal opposition parties", "boycotted the polls in", "May"),
("the principal opposition parties", "boycotted the polls after", "accusations of vote-rigging")

ArgOE requires no training data, generates triples without any post-processing, and takes as input dependency parses in CoNLL-X format [10, 14]. Given that such a dependency-based representation is provided by many robust parsers including multilingual systems, e.g., MaltParser [15] or DepPattern [9], ArgOE can be seen as a multilingual open information extractor. We will describe experiments of triples extraction performed on English, Portuguese, and Spanish text. ArgOE's source code configured for English, Spanish, Portuguese, French, and Galician, as well as other resources are released under GPL license.

This article is organized as follows. Section 2 introduces previous work on OIE: in particular it describes different types of OIE systems. Next, in Section 3, the proposed method, ArgOE, is described in detail. Then, some experiments are performed in Section 4, where ArgOE system is compared against several systems and evaluated in several languages, including Portuguese. Finally, conclusions and future work are addressed in 5.

2 Related work

The goal of an OIE system is to extract triples *(arg1, rel, arg2)* describing basic propositions from large amounts of text. A great variety of OIE systems

has been developed in recent years. They can be organized in two broad categories: those systems requiring automatically generated training data to learn a classifier and those based on hand-crafted rules or heuristics. In addition, each system category can also be divided in two subtypes: those systems making use of shalow syntactic analyisis (PoS tagging and/or chunking), and those based on dependency parsing. In sum, we identify four categories of OIE systems:

(1) Training data and shallow syntax: The first OIE system, TextRunner [2], belongs to this category. A more recent version of TextRunner, also using training data (even if hand-labeled annotated) and shallow syntactic analysis is R2A2 [6]. Another system of this category is WOE^{pos} [19] whose classifier was trained with corpus obtained automatically from Wikipedia.

(2) Training data and dependency parsing: These systems make use of training data represented by means of dependency trees: WOE^{dep} [19] and OLLIE [13].

(3) Rule-based and shallow syntax: They rely on lexico-syntactic patterns hand-crafted from PoS tagged text: ReVerb [7], ExtrHech [22], and LSOE [20].

(4) Rule-based and dependency parsing: They make use of hand-crafted heuristics operating on dependency parses: ClauseIE [5], CSD-IE [4], KrakeN [1], and DepOE [8].

Our system belongs to the fourth category and, thus, is similar to ClauseIE and CSD-IE, which are the best OIE extractors to date according to the results reported in both [5] and [4]. However, these two systems are dependent on the output format of a particular syntactic parser, namely the Standford dependency parser [11]. In the same way, DepOE reported in [8], relies on a specific dependency parser, DepPattern [9], since it only operates on the by-default output given by this parser. ArgOE, by constrast, uses as input the standard CoNLL-X format and, then, does not depend on a specific dependency parser.

Another significant difference between ArgOE and the other rule-based systems is that ArgOE does not distinguish between arguments and adjuncts. As this distiction is not always clear and well identified by the syntactic parsers, we simplify the number of different verb constituents within the argument structure: all prepositional phrases headed by a verb are taken as verb complements, regardeless of their degree of dependency (internal arguments or external adjuncts) with the verb. So, the set of rules used to generate triples from this simplified argument structure is smaller than in other rule-based approaches.

In addition, we make extraction multilingual. More precisely, our system has the following properties:

- Extraction of triples represented at different levels of granularity: surface forms and dependency level.
- Multilingual extraction based on multilingual parsing.

3 The method

Our OIE method consists of two steps: detection of argument structures and generation of triples.

3.1 Step 1: Argument structure detection

For each parsed sentence in the ConLL-X format, all verbs are identified and, for each verb (V), the system selects all dependents whose syntactic function can be part of its argument structure. Each argument structure is the abstract representation of a clause. The functions considered in such representations are *subject* (S), *direct object* (O), *attribute* (A), and all complements headed by a preposition (C). Five types of argument structures were defined and used in the first experiments: SVO, SVC+, SVOC+, SVA, SVAC+, where "C+" means one or more complements. All these argument structures are correct syntactic options in our working languages: English, Portuguese, and Spanish. Table 1 shows English examples for each type of argument structure.

Table 1. Examples of argument structures extracted from our testing dataset.

Type	**Example**	**Constituents**
SVO	*A Spanish official offered what he believed to be a perfectly reasonable explanation for why the portable facilities weren't in service*	**S**="A Spanish official", **V**="offered", **O**="what he believed to be a perfectly reasonable explanation for why the portable facilities weren't in service"
SVC_1C_2	*Output was reduced in 1996 after one of its three furnaces exploded*	**S**="Output", **V**="was reduced", $\mathbf{C}_1$="in 1996", $\mathbf{C}_2$="after one of its three furnaces exploded"
SVOC	*These immigrants deserve consideration under the laws that were in place*	**S**="These immigrants", **V**="deserve", **O**="consideration", **C**="under the laws"
SVA	*Koplowitz's next concert will be a more modest affair*	**S**="Koplowitz's next concert", **V**="will be", **A**="a more modest affair"
SVAC	*Gallery hours are 11 a.m. to 6 p.m. daily*	**S**="Gallery hours", **V**="are daily", **A**="11 a.m.", **C**="to 6 p.m."

Within a sentence, it is possible to find several argument structures corresponding to different clauses. For instance, the SVO example in Table 1 represents the argument structure associated with the clause introduced by the verb *offered*, but there are three more clauses introduced by other verbs (in bold): *he* ***believed*** *to be a perfectly reasonable explanation for why the portable facilities weren't in service*, *what* ***be*** *a perfectly reasonable explanation for why the portable facilities weren't in service*, and *the portable facilities* ***weren't*** *in service*, giving rise to the different argument structures shown in Table 2.

The constituents of an argument structure are the full phrases or clauses playing different syntactic functions within the structure. Each constituent is built by finding all dependency paths from its head to all its (direct and indirect) dependents. For instance, consider the SVA example in Table 1. To build the full constituents, the first step is to identify the head word of each constituent:

Table 2. Argument structures extracted from the sentence *A Spanish official offered what he believed to be a perfectly reasonable explanation for why the portable facilities weren't in service.*

Type	Constituents
SV0	**S**="A Spanish official", **V**="offered", **O**="what he believed to be a perfectly reasonable explanation for why the portable facilities weren't in service"
SV0	**S**="he", **V**="believed to", **O**="be a perfectly reasonable explanation for why the portable facilities weren't in service"
SVA	**S**="what", **V**="be", **A**="a perfectly reasonable explanation for why the portable facilities weren't in service"
SVA	**S**="the portable facilities", **V**="weren't", **A**="in service"

S="concert", V="be", A="affair". Then, each head is extended with all its dependency words by exploring the full dependency path and by taking into account the position in the sequence. This results in full phrases representing all constituents of the clause: S="Koplowitz's next concert", V="will be", A="a more modest affair".

There is, however, an important exception in the process of building full constituents: namely, relative clauses. The constituents we generate do not include those clauses introduced by a verb modifying a noun. For instance, the SVOC example in Table 1 contains the constituent C="under the laws", extracted from the expression *under the laws that were in place*. In this case, the relative clause was not taken into account to generate the constituent C within the argument structure of the main verb *deserve*. However, relative clauses and their antecedents also introduce argument structures. In the same example, we identify a SVA argument structure from the chain "the laws that were in place", where S="the laws", V="were", and A="in place". The main reason for removing relatives from constituents is to guarantee the generation of coherent and non over-specified propositions, as we will report in the next section.

Moreover, coordinatated conjunctions in verbal phrases are splitted into different argument structures, one for each coordinated verb. However, by taking into account the experiments performed in [5], coordinated phrases in the verb arguments are not processed.

Finally, notice that the argument structure SVO_1O_2 (e.g. *John gave Mary a present*) is not considered here, since it is not a correct syntactic structure in Spanish (nor in the rest of latin languages). In order the system to be multilingual, we have defined only those argument structures that are shared by our working languages.

3.2 Step 2: Generation of triples

One of the most discussed problems of OIE systems is that about 90% of the extracted triples are not concrete facts [3] expressing valid information about one or two named entities, e.g. "Obama was born in Honolulu". However, the vast amount of high confident relational triples (propositions) extracted by OIE systems are a very useful starting point for further NLP tasks and applications,

such as common sense knowledge acquisition [12], and extraction of domain-specific relations [18]. It follows that OIE systems are not suited to extract facts, but to transform unstructured texts into structured and coherent information (propositions), closer to ontology formats. Having this in mind, our objective is to generate propositions from argument structures, where propositions are defined as coherent and non over-specified pieces of basic information.

From each argument structure detected in the previous step, our OIE system generates a set of triples representing the basic propositions underlying the linguistic structure. We assume that every argument structure can convey different pieces of basic information which are, in fact, minimal units of coherent, meaningful, and non over-specified information. For example, consider again the sentence:

> *In May 2010, the principal opposition parties boycotted the polls after accusations of vote-rigging.*

which gives rise to the following $SVOC_1C_2$ argument structure:

> **S**="the principal opposition parties" , **V**="boycotted", **O**="the polls",
> $\mathbf{C}_1$="In May",
> $\mathbf{C}_2$="after accusations of vote-rigging"

An incoherent and over-specified extraction would generate from this structure the following odd propositions:

> P_1=("the principal opposition parties", "boycotted in", "May")
> P_2=("the principal opposition parties", "boycotted after", "accusations of vote-rigging")
> P_3=("the principal opposition parties", "boycotted the polls after accusations of vote-rigging in", "May")

Propositions P_1 and P_2 are incoherent extractions because the direct object constituent (O) is not optional and, then, may not be omitted from any proposition built from that argument structure. In addition, P_3 contains an over-specified relation constituted by several constituents of the argument structure. To ensure a correct extraction, we defined a set of simple rules allowing us to extract only those propositions that are considered as coherent and non over-specified. For this purpose, direct objects are never omitted and relations cannot contain more than one clause constituent. This way, the three coherent propositions generated from the above argument structure are the following:

> P_1=("the principal opposition parties", "boycotted", "the polls")
> P_2=("the principal opposition parties", "boycotted the polls after", "accusations of vote-rigging")
> P_3=("the principal opposition parties", "boycotted the polls in", "May")

As has been said, another restriction to avoid over-specification is to remove relative clause from the constituents. In the same way, that-clauses that are

direct objects are never inserted in the relation so as to avoid long and over-specificied relations.

Propositions are generated using trivial extraction rules that transform argument structures into triples. Table 3 shows the set of rules we used to extract triples from our five types of argument structures. As in the case of all current OIE systems, we only consider the extraction of verb-based triples. We took this decision in order to make a fair comparison when evaluating the performance of our system against similar systems (see Section 4). However, nothing prevents us from defining extraction rules to generate several triples from non-verbal structures: noun-prep-noun, noun-noun, adj-noun, and verb-adverb dependencies.

Table 3. Rules applied on five argument structures to generate the corresponding triples

Argument Structure	**Rules**
SVO	**arg1**=S, **rel**=V, **arg2**=O
SVC+	for $i = 1$ to n where n is the number of Complements C: C_i is descomposed in $prep_i$ and $Term_i$ **arg1**=S, **rel**=V+$prep_i$, **arg2**=$Term_i$

SVOC+	if O is not a that-clause, then:
	arg1=S, **rel**=V, **arg2**=O
	for $i = 1$ to n where n is the number of Complements C:
	C_i is descomposed in $prep_i$ and $Term_i$
	arg1=S, **rel**=V+O+$prep_i$, **arg2**=$Term_i$
	
	if O is a that-clause, then:
	arg1=S, **rel**=V, **arg2**=O
	for $i = 1$ to n where n is the number of Complements C:
	C_i is descomposed in $prep_i$ and $Term_i$
	arg1=S, **rel**=V+$prep_i$, **arg2**=$Term_i$

SVA	**arg1**=S, **rel**=V, **arg2**=A
SVAC+	**arg1**=S, **rel**=V, **arg2**=A for $i = 1$ to n where n is the number of Complements C: C_i is descomposed in $prep_i$ and $Term_i$ **arg1**=S, **rel**=V+A+$prep_i$, **arg2**=$Term_i$

The output of ArgOE does not offer confidence values for each extraction. As the system is rule-based, there is not probabilistic information to be considered. Finally, with regard to the output format, it is worth mentioning that most OIE systems produce triples only in textual, surface form. This can be a problem if triples are used for NLP tasks requiring more linguistic information. This way, in addition to surface form triples, ArgOE also provides syntax-based information, with PoS tags, lemmas, and heads. If more syntactic information would be required, it can be easily obtained from the dependency analysis.

4 Experiments

We conducted thre experimental studies: with English, Spanish, and Portuguese texts. Preliminary studies were performed to select an appropriate syntactic parser as input of ArgOE. Two multilingual dependency parsers were tested: MaltParser 1.7.1[3] and DepPattern 3.0 [4], which is provided with a format converter that changes the standard output of the parser into the CoNLL-X format. We opted for DepPattern as input of ArgOE because the tagset and dependency names of DepPattern is the same for all the languages it is able to analyze, and then, there is no need to configure and adapt ArgOE for each new language. The use of MaltParser with different languages would require implementing converters from tagsets and dependency names defined for a particular language to a common set of PoS tags and dependency names. Besides DepPattern, we also use two different PoS taggers as input of the syntactic analyzer: TreeTagger [17] for English texts and FreeLing [16] for Spanish and Portuguese. All datasets, extractions and labels of the two experiments, as well as a version of ArgOE configured for English, Spanish, Portuguese, French, and Galician, are freely available[5].

4.1 English evaluation

We compare ArgOE against several OIE existing systems for English, namely TextRunner, ReVerb, OLLI, WOE^{parse}, and ClausIE. In this experiment, we will report the results obtained by the the best version of ClauseIE, i.e., without considering redundancy and without processing conjunctions in the arguments. Note that we are comparing four systems based on training data (TextRunner, ReVerb, OLLI, and WOE^{parse}) against two rule-based methods: ClausIE and ArgOE.

The dataset used in the experiment is the Reverb dataset[6] manually labeled for the evaluation reported in [5][7]. The dataset consists of 500 sentences with manually-labeled extractions for the five systems enumerated above. In addition, we manually labeled the extractions obtained from ArgOE for the same 500 sentences. To maintain consistency among the labels associated to the five systems and those associated to ArgOE, we automatically identified those triples extracted by ArgOE that also appear in, at least, one of the other labeled extractions. As a result, we obtained 355 triples extracted by ArgOE that were labeled by annotators of previous work. Then, the extractions of ArgOE were given to two annotators who were instructed to consider the 355 already labeled extractions as starting point. So, our annotators were required to study and analyze the evaluation criteria used by other annotators before starting annotating

[3] `htpp://www.maltparser.org/`
[4] `http://gramatica.usc.es/pln/tools/deppattern.html/`
[5] `http://172.24.193.8/ArgOE-epia2015.tgz` (anonymous version)
[6] `http://reverb.cs.washington.edu/`
[7] `http://www-mpi-inf.mpg.de/departments/d5/software/clausie`

the rest of extracted triples. We also instructed the annotators to treat as incorrect those triples denoting incoherent and uninformative propositions, as well as those triples constituted by over-specified relations, i.e., relations containing numbers, named entities, or excessively long phrases (e.g., *boycotted the polls after accusations of vote-rigging in*). An extraction was considered as correct if it was labeled as correct by both annotators. The two annotators agreed on 75% of extractions (Cohen's kappa k = 0,50), which is considered a moderate agreement. In sum, we follow similar criteria to those defined in previous OIE evaluations [6].

The results of our evaluation are summarized in Table 4 and Figure 1. Table 4 shows the number of correct expressions extracted as well as the total number of extractions for each system. *Precision* is defined as the number of correct extractions divided by the number of returned extractions. *Recall* is estimated by identifying a pool of relevant extractions which is the total number of different correct extractions made by all the systems (this pool is our gold-standard). So, *recall* is the number of correct extractions made by the system divided by the total number of correct expressions in the pool $(3,222)$.

Table 4. Number of correct extractions and total number of extractions in the Reverb dataset, according to the evaluation reported in [5] and our own contribution with ArgOE.

Systems	**correct extractions**	**total extractions**
textrunner	286	798
reverb	388	727
woe	447	1028
ollie	547	1242
argoe	582	1162
clausie	1706	2975

The results show that the two rule-based systems, ClausIE and ArgOE, perform better than the classifiers based on automatically generated training data. This is in accordance with previous work reported in [5, 4]. Moreover, the four systems based on dependency analysis (ClausIE, ArgOE, OLLIE, and WOE^{parse}) improve over those relying on shallow syntax (TextRunner and ReVerb). And finally, ClausIE clearly outperforms the other systems, in terms of both precision and recall. A common problem for parse-based OIE systems is the large influence of parser errors. So, the quality of the parser can determine the quality of the OIE extractor. ClausIE uses the Standford Dependency Parser, while ArgOE uses DepPattern, and OLLIE the MaltParser. One possible reason for the comparably low precision of our system against ClausIE might be the lower parsing performance of DepPattern against the Standford Dependency Parser for the English language.

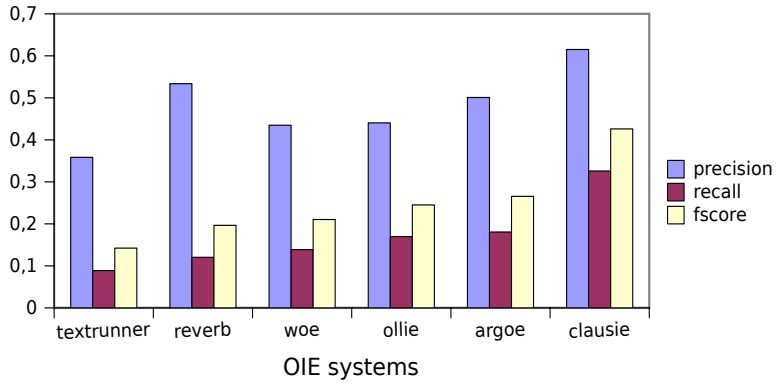

Fig. 1. Evaluation of six OIE systems

4.2 Spanish evaluation

In this experiment, we compare ArgOE against the only OIE system that has been evaluated for other language than English: ExtrHech [22]. This is also a rule-based system, but it does not operate on dependency parsing but on shallow syntax (patterns of PoS tags). The Spanish dataset, called Raw Web[8], contains 159 sentences randomly extracted with a web crawler from over 5 billion web pages in Spanish. Each extraction was labeled by two independent annotators. An extraction was considered as correct if it was labeled as correct by both annotators. They agreed on 81% of extractions (Cohen's kappa k = 0,62). Table 5 depicts the results obtained by the two systems on these sentences. Unfortunately, the extractions made by ExtrHech are not available, so it is not possible to create a pool of correct triples extracted by the two systems to measure recall. Only precision can be compared even if we were not able to unify the criteria given to our annotators with those defined in [22]. Notice that the precision of ArgOE is identical to that obtained for English (50%), which can be seen as an indirect evidence that the two parsers used by our system have similar performance.

Table 5. Precision of both ArgOE and ExtrHech on the Spanish dataset

Systems	**correct extractions**	**total extractions**	**Precision (%)**
argoe	107	214	50%
extrahech	-	-	55%

[8] `http://www.gelbukh.com/resources/spanish-open-fact-extraction`

Most errors made by our OIE system come from three different sources: the syntactic parser, the PoS tagger, and the Named Entity Recognition module used by the PoS tagger. So, the improvement of our system relies on the performance of other NLP tasks.

4.3 Portuguese Evaluation

For this purpose, we selected 103 test sentences from a domain-specific corpus, called *CorpusEco* [21], containing texts on ecological issues. ArgOE was applied on the sentences and 190 triples was extracted. One annotator labeled the extracted triples and Table 6 shows the number of correct triples and precision achieved by the system. To the best of our knowledge, this is the first experiment that reports an OIE system working on Portuguese. Precision is again similar (53%) to that obtained in the previous experiments. Again, most errors are due to problems from the syntactic parser and PoS tagger.

Table 6. Precision of ArgOE on the Portuguese dataset

Systems	**correct extractions**	**total extractions**	**Precision (%)**
argoe	95	190	53%

5 Conclusion

We have described a rule-based OIE system to extract verb-based triples than takes as input dependency parsers in the CoNLL-X format. So, it may take advantage of efficient, robust, and multilingual syntactic parsers. Even if our system is outperformed by other similar rule-based methods, it reaches better results than those strategies based on training data. As far as we know, ArgOE is the first OIE system working on more than one language. In future work, we will include NLP modules to find linguistic generalizations over the extracted triples: e.g., co-reference resolution to link the arguments of different triples, and synonymy detection of verbs to reduce the open set of extracted relations and, then, to enable semantic inference.

References

1. Alan Akbik and Alexandre Loser. Kraken: N-ary facts in open information extraction. In *Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 52–56, 2012.
2. Michele Banko, , and Oren Etzioni. The tradeoffs between open and traditional relation extraction. In *ACL-08*, 2008.

3. Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. Open information extraction from the web. In *International Joint Conference on Artificial Intelligence*, 2007.
4. Hannah Bast and Elmar Haussmann. Open information extraction via contextual sententence decomposition. In *ICSC 2013*, pages 154–159, 2013.
5. Luciano Del Corro and Rainer Gemulla. Clausie: Clause-based open information extraction. In *Proceedings of the World Wide Web Conference (WWW-2013)*, pages 355–366, Rio de Janeiro, Brazil, 2013.
6. Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam. Open information extraction: the second generation. In *International Joint Conference on Artificial Intelligence*, 2011.
7. Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *EMNLP-11*, 2011.
8. Pablo Gamallo, Marcos Garcia, and Santiago Fernández-Lanza. Dependency-based open information extraction. In *ROBUS-UNSUP Workshop at EACL-2012*, Avignon, France, 2012.
9. Pablo Gamallo and Isaac González. A grammatical formalism based on patterns of part-of-speech tags. *Journal of Corpus Linguistics*, 16(1):45–71, 2011.
10. Johan Hall and Jens Nilsson. CoNLL-X shared task on multilingual dependency parsing. In *The tenth CoNLL*, 2006.
11. Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *ACL-03*, pages 423–430, 2003.
12. Thomas Lin, Mausmam, and Oren Etzioni. Identifying functional relations in web text. In *Conference on Empirical Methods in Natural Language Processing*, 2010.
13. Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. Open language learning for information extraction. In *EMNLP-12*, pages 523–534, 2012.
14. J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilson, S. Riedel, and D. Yuret. The CoNLL-2007 shared task on dependency parsing. In *Proceedings of the Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, 2007.
15. Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):115–135, 2007.
16. Lluís. Padró and Evgeny Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *LREC'12*, Istanbul, Turkey, 2012.
17. Helmut Schimd. Improvements in part-of-speech tagging with an application to german. In *ACL SIGDAT Workshop*, Dublin, Ireland, 1995.
18. Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Mausam, and Oren Etzioni. Adapting open information extraction to domain-specific relations. *AI Magazine*, 31(3):93–102, 2010.
19. Fei Wu and Daniel S. Weld. Open information extraction using wikipedia. In *Annual Meeting of the Association for Computational Linguistics*, 2010.
20. Clarissa C. Xavier, Marlo Souza, and Vera S. de Lima. Open information extraction based on lexical-syntactic patterns. In *Brazilian Conference on Intelligent Systems*, pages 189–194, 2013.
21. C. Zavaglia. O papel do léxico na elaboraĉão de ontologias computacionais: do seu resgate à sua disponibilizaĉão. In *Lingüística IN FOCUS - Léxico e morfofonologia: perspectivas e análises*, pages 233–274. Uberlândia: EDUFU, 2006.
22. Alisa Zhilla and Alexander Gelbukh. Comparison of open information extraction for engish and spanish. In *Dialogue 2014*, 2014.