An Exploration of the Linguistic Knowledge for Semantic Relation Extraction in Spanish

Marcos Garcia
Center for Research in Information Technologies (CITIUS)
University of Santiago de Compostela
marcos.garcia.gonzalez@usc.es

Pablo Gamallo
Center for Research in Information Technologies (CITIUS)
University of Santiago de Compostela
pablo.gamallo@usc.es

Abstract

A common strategy for Question Answering systems uses high quality ontologies or databases in order to efficiently answer questions. Some approaches to build or enrich these databases rely on machine learning classifiers for obtaining semantically related terms from unstructured text. These classifiers are based on features that may contain several kinds of linguistic knowledge: from orthographic or lexical information to more complex features, including PoS-tags, syntactic dependencies or semantic information. In this paper we select four main types of linguistic features and systematically evaluate their performance on semantic Relation Extraction. Although the combination of some types of linguistic features allows us to improve the f-score of the classifiers, we observed that by adjusting the positive/negative ratio of the training examples, we can build high quality classifiers with just a single type of linguistic feature, based on generic lexico-syntactic patterns. Experiments were carried out with the Spanish version of Wikipedia.

1 Introduction

Recent approaches for Question Answering systems use high quality ontologies or databases for answering questions. Building these resources manually is time-consuming, so the improvement of Information Extraction (IE) methods becomes a crucial task. In this sense, Relation Extraction (RE) attempts to automatically obtain semantic information related to named entities from unstructured text, which, in turn, is incorporated into machine readable databases and ontologies. For instance, from the sentence *Amancio Ortega (born March 28, 1936) is a Spanish fashion entrepreneur*, a system may learn different properties about Amancio Ortega: his birthdate, his nationality as well as his occupation.

Most of recent RE techniques employ machine learning algorithms for extracting information. They arrange into a set of linguistic features the contexts or sentences in which pairs of related entities occur. These features are then used to train a classifier. The starting point is a number of labeled training examples (e.g., relations "Amancio Ortega - entrepreneur", "Billie Holiday - singer"...), as well as a corpus-based strategy to search, identify and represent as features those sentences (contexts) in which the training examples occur. These strategies may have different degrees of supervision, from manually constructed corpora to unsupervised methods that do not need human intervention for revising the sentences. Moreover, it must be pointed out that linguistic features can be represented at different levels of generality, according to different types of linguistic information. However, there are no much work on the importance of knowing what linguistic knowledge is actually useful in order to increase the performance of these systems.

In this article, our aim is to evaluate the impact of different types of linguistic features in the RE task for Spanish. For this purpose, a distant supervision learning system was built and tested with different types of features: bags of lemmas and PoS-tags, lexico-syntactic patterns and syntactic dependencies. We evaluated three main aspects concerning extraction of semantic relations: First, we measured the impact of linguistic information by training several classifiers that only differ in the type of linguistic knowledge used to define their features. Second, we analyzed the best feature combinations. And third, we found the best ratio of positive and negative examples according to each type of linguistic features.

Preliminary results show that, usually, the best performance was achieved using just a single type of features based on generic lexico-syntactic patterns, built by means of a *longest common string* algorithm. However, the combination of some types of linguistic features allows us to increase recall without losing precision. Furthermore, we also observed that a deep analysis of the positive/negative ratio of the training examples improves the f-score of the classifiers.

In Section 2 we show the related work. Section 3 explains the method for obtaining the data used to build the classifiers. In Section 4 we present the feature space as well as the different types of features used. Finally, Sections 5 and 6 show the results and the future directions of our work.

2 Related Work

Since the work of Hearst [1992], many approaches have been implemented in order to obtain patterns for extracting related terms, such as Brin [1998] or Ravichandran and Hovy [2002].

In the recent years, works like Mann [2002], Fleischman *et al.* [2003] or Agichtein [2005] use different features for

training machine learning classifiers with several algorithms. The selected features differ according to the linguistic information they process (word, lemma, PoS-tag, orthographic form, etc.). Other works [Bunescu and Mooney, 2005; Suchanek *et al.*, 2006; Nguyen *et al.*, 2007] created statistical models using the results of syntactic parsing. Despite that some preliminary results of these researches show that the use of deep linguistic knowledge is better for relation extraction, Bunescu and Mooney [2005] warn of the importance of knowing what of this linguistic information is actually useful in order to increase the performance of an IE system.

In this sense, some works must be cited: Kambhatla [2004], Zhou *et al.* [2005] and Jiang and Zhai [2007], which evaluate the effectiveness of diverse lexical, syntactic, and semantic information in RE, and Zhao and Grishman [2005] which use a more complex kernel-based strategy to combine features of different linguistic levels. However, their work differs from ours in a key point: the linguistic feature space is not the same, in particular they do not make use of lexico-syntactic patterns. Moreover, our evaluation concerns other languages than English as well as a deep analysis of the impact of negative examples on the training data.

Finally, Wu and Weld [2010] present *woe*, an Open Information Extraction method based on data obtained from the semi-structured resources of Wikipedia.

3 Method Overview

In order to easily evaluate the performance of various types of features, we first obtain an annotated corpora, used to build the different classifiers by the following distant supervision method (inspired by Mintz *et al.* [2009]):

A Wikipedia snapshot is converted into plain text, removing external links as well as formatting marks. Given a semantic relation, for instance *Occupation*, we get a large set of related pairs from Wikipedia infoboxes: e.g., "John Lennon - musician", "John Lennon - composer", "Fernando Pessoa - poet" (with a precision of about 95%). We use these pairs to select from the plain text of Wikipedia sentences that contain both a named entity and an occupation, so no bootstrapping is required. If the two terms match a known pair of the initial list, the example is annotated as positive. Otherwise, it is annotated as negative.

Then, we use FreeLing [Padró *et al.*, 2010] to lemmatize, PoS-tag and recognize the proper names in the sentences. Syntactic dependencies are identified with a robust, partial, and rule-based parser [Gamallo and González, 2011].

The two target entities are replaced by both **X** and **Y** (standing for the first and the second entities of the pair, respectively) and put labels to mark the left, middle, and right contexts.

All the process is performed without human revision, so it may lead us to automatically annotate *false positives* ("*Linus Torvalds* discussed with a *software engineer* in Italy", **true**) or *false negatives* ("*Fernando Pessoa* was a *literary critic*", **false**, since this attribute does not appear in the infobox). The manual revision of the test set showed that this method has a precision of about 80%. This issue will be addressed by using machine learning algorithms (e.g. SVM) that are tolerant to noise by minimizing the effect of the false samples [Aslam and Decatur, 1996].

Sentence: Amancio Ortega Gaona is a Spanish fashion entrepreneur.

Structure:

pos	token	lemma	tag	head	label
0	**X**	**X**	NP	1	subj
1	is	be	V	0	-
2	a	a	DI	5	spec
3	Spanish	spanish	ADJ	5	modif
4	fashion	fashion	N	5	modif
5	**Y**	**Y**	N	1	attr

Figure 1: Example of a sentence with its Linguistic Structure.

4 Feature Space and Types of Features

Each selected, tagged, and parsed sentence represents a *linguistic structure* containing all the relevant information required by the relation extraction systems. A linguistic structure or parsed sentence can be conceived as a knowledge-rich space incorporating several levels of linguistic information. This space should be as complete as possible in the sense that all features potentially useful for RE are included.

A linguistic structure contains the surrounding context of the related entities: **X** stands for the named entity and **Y** for the occupation name. We include within a linguistic structure the left context of the first entity, the middle context, and the right context of the second entity. Left and right contexts have a maximum size of 3 tokens (from *pos* -1 to -3 and +1 to +3). Middle context may contain 12 tokens (from 1 to 12). The window size was empirically set to 3 and 12 after having tested different values in preliminary experiments. We will distinguish experiments using all contexts (left, right, and middle) from those considering only the middle one.

In Figure 1, columns 1, 2, 3, and 4 respectively stands for position, token, lemma, and PoS-tag. Furthermore, the structure also contains the syntactic dependencies identified by the parser. Column 5 identifies the head position of the current token, while the label of the dependency is shown by column 6. Since we use a partial parser, not all possible word dependencies are identified. Our structure was inspired by the output parsing format defined in [Lin, 1998], adopted by learning tasks of CoNLL.

These linguistic structures are used to extract the different types of linguistic features needed by our classifiers. We use four main different types of linguistic features in order to build the relation extraction systems:

Basic Patterns: The first type of feature uses all the explicit information contained in the linguistic structures except two elements: dependency relations and some lemmas. We only take into account lemmas of verbs, common nouns and prepositions. We have observed in preliminary experiments that the performance of classifiers decreased when either these type of lemmas were removed or all lemmas including grammatical words (stop words), adjectives and proper names were retained. It follows that verbs, common nouns and prepositions

are critical pieces of information to define the lexico-syntactic contexts of the target terms. An example of basic patterns associated to the relation "Occupation" is the following:

Sentence: Amancio Ortega Gaona is a Spanish fashion entrepreneur.

Pattern: <**X** be_V DI ADJ fashion_N **Y**>

(where V means verb, DI indefinite article, ADJ adjective, and N common noun). Let us note that basic lexico-syntactic patterns require a huge training corpus, due to their lack of flexibility. Small variations in punctuation, adjective or adverb modification, determiners, etc. will generate different features. In this case, the problem of sparse data is crucial.

Pattern Generalization: To minimize the sparse data problem, we applied an algorithm based on similarity between basic patterns, generalizing them and therefore increasing the coverage of the model. In order to generalize two patterns, we check first if they are similar and then all those units that they do not share are removed [Ruiz-Casado *et al.*, 2005]. The similarity, noted $Dice\_lcs$, between two patterns p_1 and p_2 is defined using the longest common string and Dice metric as follows:

$$Dice\_lcs(p_1, p_2) = \frac{2 * lcs(p_1, p_2)}{length(p_1) + length(p_2)} \quad (1)$$

where $lcs(p_1, p_2)$ is the size of the longest common string between patterns p_1 and p_2, and $length(p_i)$ represents the size of pattern p_i. It means the similarity between two patterns is a function of their longest common string and their lengths.

After computing the similarity between two patterns p_1 and p_2, the longest common string is extracted if and only if p_2 is the most similar pattern of p_1 and the similarity score is higher than a particular threshold (0.75 in our tests). The longest common string of two patterns is considered as the generalized pattern out of them.

Bags of Lemmas and Tags: Instead of using a set of entire patterns as features, a common method that increases the coverage of the system is the use of smaller items, such as lemmas with PoS-tags. In this case, the sentence *Amancio Ortega Gaona is a Spanish fashion entrepreneur* would generate the following features (notice again that for some categories only the tag is retained): <be_V>, <DI>, <ADJ> and <fashion_N>. The use of negative examples is probably more important here, allowing the algorithm to learn which lemmas are crucial for making a correct decision. These classifiers are then close to the baseline bag-of-words models.

Syntactic Dependencies: Dependency information is obtained by the refered robust, multilingual, and rule-based parser. The analysis of the parser is partial, but it provides the most frequent dependencies between the target terms and the units within the patterns. Here again only the lemmas of verbs, common nouns, and prepositions are retained.

We consider a subset of all the syntactic dependencies derived from the full linguistic structures. Given a sentence, two types of dependencies are retained:

- dependencies between the two target terms (**X** or **Y**),
- dependencies between one of the two target terms and one entity of the (left, middle, or right) context.

For instance, from the sentence *Amancio Ortega Gaona is a Spanish fashion entrepreneur*, the selected dependencies would be the following: <subj;**X**;be_V>, <attr;be_V;**Y**>, <spec;**Y**;DI>, <modif;**Y**;ADJ> and <modif;**Y**;fashion_N>.

Each feature is a triple constituted by the dependency label, the head, and the dependent expression. Only dependencies with at least one term (**X** or **Y**) were selected from the linguistic structure. The selected information, thus, corresponds to the local dependency context around the related terms.

Note that the analysis is very partial. In many cases, the parser is not able to complete the dependency path between the two terms. Rule-based grammars for other languages than English are often not complete or they are not freely available. Finally, labels of dependencies (e.g., modifier, specifier, subject, etc.) are also taken into account to define the features.

5 Experiments

We evaluated both the performance of the features individually as well as the best combinations of them. We also examined the effect of limited training input on the learning process by incrementally adding examples to the training data. Finally, we analyzed the impact of negative examples in the training set.

The experiments were performed with WEKA [Witten and Frank, 2002], using its implementation of the SMO algorithm [Platt, 1999]. We made this choice because in preliminary experiments (using Naive Bayes, Decision Tree as well as SMO algorithms), SMO scored the best.

5.1 Initial Data

The training examples were obtained from the Spanish Wikipedia (May 2010) with the method showed in Section 3. We focused on examples of the relations "Occupation" (a kind of *is_a* relation) and "Birth Place". We first selected about $50,000$ relation pairs for each relation from infoboxes. Then, we identified near $600,000$ sentences containing a named entity and an occupation or a location, which were automatically classified as positive or negative. Finally, we randomly selected an initial set of 2000 sentences (for each relation) for training, 50% of them being positive examples.

For testing, we manually revised two randomly selected sets of about 700 and 500 different sentences for "Occupation" and "Birth Place" relations, respectively.

5.2 Results

Table 1 shows the results obtained with 7 types of linguistic features, extracted from training set of 2000 linguistic structures. The types of features are those explained in Section 4:

pattern-all and *pattern-mid* use the basic patterns extracted from the linguistic structures as features. The former was trained with all contexts (left, right, and middle), while the latter was only trained with the middle context.

pattern_gen-mid uses as features the generalized patterns and the middle context.

Model	Prec.	Rec.	f-score
pattern-all	**100%**	12.99%	23%
pattern-mid	98.04%	56.5%	71.68%
pattern_gen-mid	97.72%	**72.6%**	**83.31%**
bow-all	83.02%	62.15%	71.08%
bow-mid	88.28%	59.6%	71.16%
dep-all	86.62%	65.82%	74.8%
dep-mid	86.21%	35.31%	50.1%

Table 1: Precision, Recall and f-score of 7 classifiers trained with 2000 sentences ("Occupation" relation).

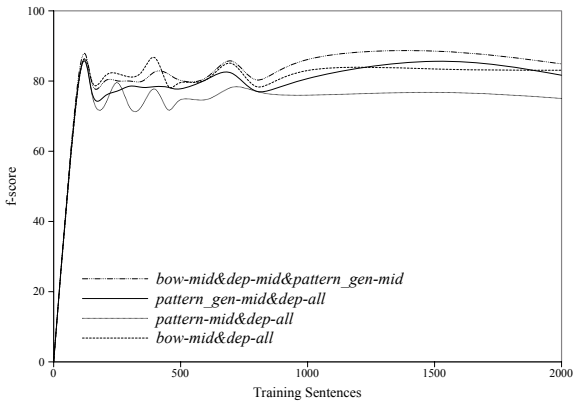

Figure 2: f-score vs training size of the 4 best combinations of features ("Occupation" relation).

bow-all and *bow-mid* were built with the bag of lemmas and tags technique.

dep-all and *dep-mid* are the dependency-based models described in the previous section.

Precision is the number of correct positive decisions divided by the number of positive decisions. Recall here refers to the number of correct positive decisions divided by the total number of positive examples in the test set.

Table 1 let us observe that the best features are those based on lexico-syntactic patterns with *mid* contexts: *pattern_gen-mid* and *pattern-mid*, with f-score values between 71%-83%. The score reached by *pattern-all* is much lower because of very poor recall values. This is due to the fact that, in this case, both left and right contexts tend to be too sparse. Table 1 also shows that features based on dependencies and on bags of lemmas and tags achieve similar precision (83%-86%), and that in these cases, *all* contexts scored better than *mid*.

The results of these tests allow us to know the performance of the individual types of features. The next experiment attempts to know the performance of several combinations of individual features, as well as their learning curve.

When analyzing the differences between the models, we both compute the Dice similarity coefficient and a simple count statistic aimed to find whether the decisions taken by two models are or not on the same instances. In general, a high Dice coefficient may imply that there are few correct decisions taken on different instances and, conversely, a low Dice coefficient means that there are many correct decisions taken on different instances. Only not very similar models were combined since they are likely to be complementary.

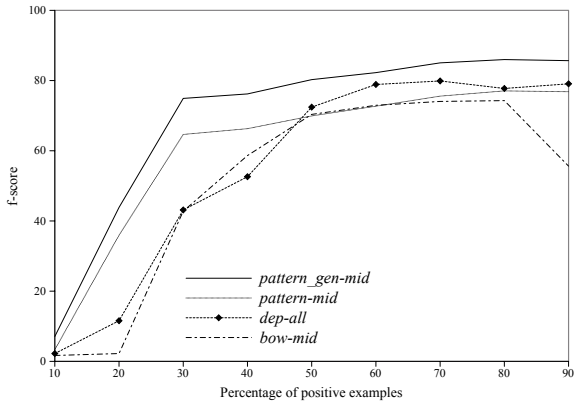

Figure 3: f-score vs ratio of positive-negative examples of the best classifiers for each linguistic type trained with 500 sentences ("Occupation" relation).

The results of several combinations based on a similarity analysis show that these classifiers may help to achieve a trade-off between precision and recall. Figure 2 shows the f-score curve of the four best combinations. Notice that the best combinations scored better than the best individual feature (*pattern_gen-mid*, which reached 83.31%), achieving values of 88% with around 1500 of training sentences. However, it is important to note that the best f-score values are reached without making use of the whole training set.

The initial set of 2000 sentences, with which we performed the previous experiments, had a 50%/50% ratio of positive and negative examples. Our next experiment concerns the evaluation of the positive/negative (P/N) ratio of the training examples for each type of linguistic feature.

In order to know the best P/N distribution, we collected several sets of sentences differing in the percentage of positive and negative examples they have. Note that this evaluation also deals with the amount of P/N instances, and not only with its P/N ratio. So, in order to avoid this effect, we performed two major experiments: (i) we collected nine sets of 500 sentences differing in the P/N ratio: from 10%/90% to 90%/10% and (ii) we did the same distributional partitions from a larger corpus (sets of 2000 sentences). Finally, we analyzed how the learning process is influenced by the P/N ratio as well as by the number of positive and negative examples.

Figures 3 and 4 show the f-score curve of the best classifiers for each main type of features, depending the P/N ratio of the training set. Classifiers from Figure 3 were trained with nine partitions of 500 sentences, while in Figure 4 the training sets had 2000 sentences. In most cases, the peak of the f-score curve is reached with between 60% and 80% of positive data (except for patterns-based features, whose performance gradually improves with more positive examples). This is true for the two tests (500 and 2000 sentences), so we can infer the best P/N ratio for each type of feature. Note that the 20%/80% ratio in Figure 4 has the same number of positive examples than the 90%/10% in Figure 3.

Pattern-based features need a large set of training data in order to learn a large enough set of patterns as well as their

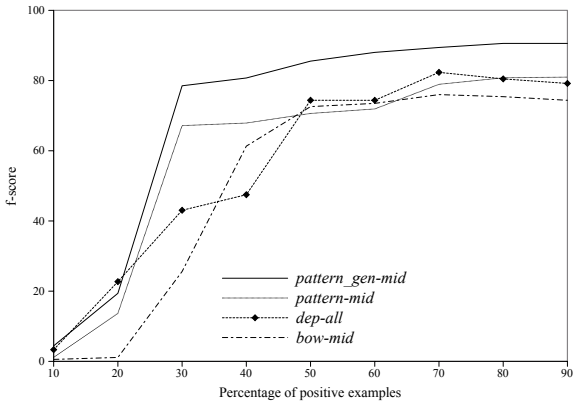

Figure 4: f-score vs ratio positive-negative examples of the best classifiers for each linguistic type trained with 2000 sentences ("Occupation" relation).

Model	Prec.	Rec.	f-score	Diff.
pattern-all	**100%**	14.56%	25.42%	+2.4
pattern-mid	95.23%	65.93%	77.92%	+6.2
pat_gen-mid	94.95%	82.69%	**88.39%**	+5.1
bow-all	72.86%	**95.87%**	82.79%	+11.7
bow-mid	82.33%	64.01%	72.02%	+0.9
dep-all	77.89%	81.31%	79.56%	+4.8
dep-mid	73.98%	60.16%	66.36%	+16.3

Table 2: Precision, Recall and f-score of 7 classifiers trained with 2000 sentences ("Occupation" relation). The positive/negative ratio was adjusted for each model. *Diff.* shows the f-score difference with 50%/50% models.

confidence. However, Figures 3 and 4 show that these models also need a low ratio of negative examples (its performance becomes stable with about 80%/20%). By contrast, *dep-all* (and also *dep-mid*) achieves its best score with 70%/30%, as it decreases with less negative data. Finally, features based on bags of lemmas are more variable, and work better with, at least, 20% of negative examples (but never with less than 50% of positive training data).

The results from the previous experiment provide us information to train new classifiers with the P/N ratio adjusted to each type of feature. So, we built seven new classifiers based on the same seven individual types of features described above. We randomly selected seven sets of 2000 sentences, each one with a distribution of positive and negative examples adjusted to the needs of each type of features, and tested them on the 700 manually revised examples.

Figure 5, shows the f-score of the best individual features (for each one of the 4 main types) in different partitions. The P/N ratio in the training set was adjusted for each classifier. Note that the performance of all classifiers was improved (namely due to the improvement in recall) from those obtained with the 50%/50% P/N training corpus (4th column in Tables 1 and 2). Furthermore, it can be observed that the curve stabilizes when the training corpus is constituted by about 1000 sentences. So, no more training corpus is required to improve results.

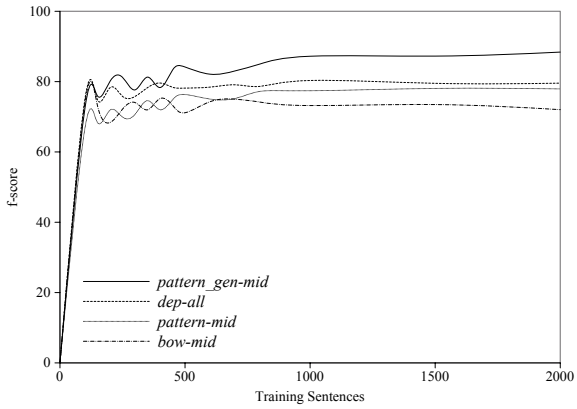

Figure 5: f-score vs training sentences of the best classifiers for each linguistic feature ("Occupation" relation). Positive/negative ratio was adjusted for each classifier.

Model	Prec.	Rec.	f-score
pattern-all	**88.88%**	13.55%	23.52%
pattern-mid	76.47%	44.06%	55.91%
pat_gen-mid	88%	74.57%	**80.73%**
bow-all	42.6%	**83.05%**	56.32%
dep-all	49.36%	66.1%	56.52%

Table 3: Precision, Recall and f-score of 5 classifiers trained with 2000 sentences ("Birth Place" relation). The positive/negative ratio was adjusted for each classifier.

Finally, Table 3 contains the results obtained from an ongoing test with "Birth Place" semantic relation. Unlike what occurred with "Occupation", *bow* models (and also *dep*) achieved lower precision values. However, results still follow the general tendencies. Note that the generalization of lexico-syntactic patterns also achieves high f-score values (80.73%) in these experiments. With both semantic relations, the performance of the best classifiers based on individual types of features, built on the analysis of the best P/N ratios, is similar than the best combinations.

6 Conclusions and Further Work

The experiments described in this article can help us to outline some ideas regarding the impact of the linguistic knowledge on Relation Extraction for Spanish.

First, the application of the *longest common string* over the lemma_tag patterns are good features to define models with high precision and relatively good recall. Models built of bags of lemmas and tags as well as dependency-based show more variation in precision values. Concerning these dependency models, we have to note that the use of *partial* parsing may also involve variations in recall.

Second, the adaptation of the positive/negative ratio in the training set helps to improve the performance of the classifiers. We observed that features based on patterns of PoS-tags and (some) lemmas need a high percentage of positive examples. Those based on bags of lemmas and syntactic dependencies perform better with more than 50% of positive

examples, but they need at least 20% of negative data.

If the objective is to obtain a trade-off between precision and recall, we can analyze the similarity between the results of several types of features in order choose the best combinations for a specific relation. However, if we attempt to rapidly build new RE systems, the construction and generalization of patterns with PoS-tags and lemmas is the best solution.

Further experiments will be focused on the construction of new classifiers based on combinations of the best types of features, each one with its positive/negative ratio adjusted. Moreover, we are testing the impact of noisy data in the training phase, as well as the adaptation of our strategy for extracting other semantic relations and text typologies.

Acknowledgments

This work has been supported by the MICINN, within the project with reference FFI2010-14986.

References

[Agichtein, 2005] E. Agichtein. *Extracting Relations from Large Text Collections*. PhD thesis, Columbia University, 2005.

[Aslam and Decatur, 1996] J. A. Aslam and S. E. Decatur. On the sample complexity of noise-tolerant learning. *Information Processing Letters*, 57:189–195, 1996.

[Brin, 1998] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology*, pages 172–183, 1998.

[Bunescu and Mooney, 2005] R. C. Bunescu and R. J. Mooney. A Shortest Path Dependency Kernel for Relation Extraction. In *Proceedings of the HLTC on EMNLP*, pages 724–731, 2005.

[Fleischman *et al.*, 2003] M.l Fleischman, E. Hovy, and A. Echihabi. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 1–7, 2003.

[Gamallo and González, 2011] P. Gamallo and I. González. A Grammatical Formalism based on Patterns of Part-of-Speech Tags. *Journal of Corpus Linguistics*, 16(1):45–71, 2011.

[Hearst, 1992] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, volume 2, pages 539–545, Morristown, NJ, USA, 1992.

[Jiang and Zhai, 2007] J. Jiang and C. Zhai. A Systematic Exploration of the Feature Space for Relation Extraction. In *Proceedings of NAACL/HLT*, pages 113–120, 2007.

[Kambhatla, 2004] N. Kambhatla. Combining lexical, syntactic and semantic features with Maximum Entropy models for extracting relations. In *Proceedings of the 42th Annual Meeting of the ACL*, pages 178–181, 2004.

[Lin, 1998] Dekang Lin. Dependency-based Evaluation of MINIPAR. In *Proceedings of the Workshop on the Evaluation of Parsing Systems*, Granada, 1998.

[Mann, 2002] G. S. Mann. Fine-Grained Proper Noun Ontologies for Question Answering. In *SemaNet'02: Building and Using Semantic Networks*, Taipei, Taiwan, 2002.

[Mintz *et al.*, 2009] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the ACL/IJCNLP*, pages 1003–1011, 2009.

[Nguyen *et al.*, 2007] Dat P. T. Nguyen, Y. Matsuo, and M. Ishizuka. Relation Extraction from Wikipedia Using Subtree Mining. In *Proceedings of the 22nd national conference on Artificial intelligence*, volume 2, pages 1414–1420, 2007.

[Padró *et al.*, 2010] Ll. Padró, M. Collado, S. Reese, M. Lloberes, and I. Castellón. FreeLing 2.1: Five Years of Open-Source Language Processing Tools. In *Proceedings of the 7th Language Resources and Evaluation Conference*, La Valletta, Malta, 2010.

[Platt, 1999] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods: support vector learning*, pages 185–208, Cambridge, MA, USA, 1999.

[Ravichandran and Hovy, 2002] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 41–47, 2002.

[Ruiz-Casado *et al.*, 2005] Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. Automatic assignment of Wikipedia encyclopedic entries to WordNet synsets. volume 3528, pages 380–386, 2005.

[Suchanek *et al.*, 2006] F. M. Suchanek, G. Ifrim, and G. Weikum. LEILA: Learning to Extract Information by Linguistic Analysis. In *Second Workshop on Ontology Population at ACL/COLING*, 2006.

[Witten and Frank, 2002] I. Witten and E. Frank. Data mining: practical machine learning tools and techniques with Java implementations. *SIGMOD Rec.*, 31:76–77, 2002.

[Wu and Weld, 2010] F. Wu and D. S. Weld. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the ACL*, pages 118–127, 2010.

[Zhao and Grishman, 2005] S. Zhao and R. Grishman. Extracting Relations with Integrated Information Using Kernel Methods. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 419–426, 2005.

[Zhou *et al.*, 2005] G. Zhou, J. Su, J. Zhang, and M. Zhang. Exploring Various Knowledge in Relation Extraction. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 427–434, 2005.