Evaluating Various Linguistic Features on Semantic Relation Extraction

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

Machine learning approaches for Information Extraction use different types of features to acquire semantically related terms from free text. These features may contain several kinds of linguistic knowledge: from orthographic or lexical to more complex features, like PoS-tags or syntactic dependencies. In this paper we select four main types of linguistic features and evaluate their performance in a systematic way. Despite the combination of some types of features allows us to improve the f-score of the extraction, we observed that by adjusting the positive and 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 performed on the Portuguese version of Wikipedia.

1 Introduction

With the exponential growth of data, the interest in learning semantic information related to named entities has been increased. For instance, from the sentence *Ernest Hemingway (July 21, 1899 - July 2, 1961) was an American author*, a system may learn various properties about Hemingway (his birth and death dates, his origin as well as his occupation).

Many techniques employ Machine Learning (ML) algorithms for the extraction task. They arrange into a set of 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 ("Ernest Hemingway - author"), as well as a corpus-based strategy to identify and represent as features those sentences (contexts) in which the training examples occur. These techniques may have different degrees of supervision, from manually constructed corpora to weakly supervised on unsupervised methods. Moreover, it must be pointed out that features can be represented at different levels of generality, according to different types of knowledge. However, there are no much work on the importance of knowing what linguistic information is actually useful in order to increase the performance of these systems.

In this article, we evaluate and compare the impact of different types of linguistic features for Relation Extraction (RE). We built and tested a distant supervision system with different types of linguistic features: bags of lemmas and PoS-tags, lexico-syntactic patterns and syntactic dependencies. We evaluated the performance of these types of features individually and in several combinations. Preliminary results in Portuguese data show that, usually, 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 (P/N) ratio of the training examples improves the f-score of the classifiers.

Sect. 2 introduces some related work. Then, 3 shows the method for obtaining the data and Sect. 4 presents the features. In Sect. 5 we show the results. Finally, Sect. 6 draws the conclusions 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).

More recent works (Mann, 2002; Fleischman et al., 2003; Agichtein, 2005) use different features (words, lemmas, PoS-tags, etc.) for training ML systems with several algorithms. Other works (Snow et al., 2005; Bunescu and Mooney, 2005) created statistical models using the results of syntactic parsing. Despite that some preliminary results show that the use of deep linguistic knowledge is better for RE, Bunescu and Mooney (2005) warn of the importance of knowing what of this information is actually useful to increase the performance of an IE system.

In this sense, some works must be cited: Kambhatla (2004) and Zhou *et al.* (2005), which evaluate the effectiveness of diverse lexical, syntactic, and semantic information on 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 (see Garcia and Gamallo (2011) for some other evaluations in Spanish).

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

3 Method Overview

In order to easily evaluate the performance of various types of features, we use the following distant supervision method (Mintz et al., 2009):

We get a large set of entity pairs of a desired relation from (semi)structured resources, such as Wikipedia infoboxes. For instance, for the *Occupation* relation we get pairs such as "Michel Tournier - writer", "Edgar Snow - journalist", etc. (with about 95% precision). We use these pairs to select from the unstructured 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 lemmatize, PoS-tag and recognize the proper names with FreeLing (Padró et al., 2010; Garcia and Gamallo, 2010). Syntactic dependencies are identified by a robust, partial, and rule-based dependency 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. Let us note this method 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 ML algorithms that are tolerant to noise by minimizing the effect of the false samples.

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 systems. Linguistic structures can be conceived as knowledge-rich spaces incorporating several levels of linguistic information. These spaces 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), while middle context may contain 12 tokens (from *pos* 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 taking into account 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. 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

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 linguistic structure.

a partial parser, not all possible word dependencies are identified. These linguistic structures are used to extract the different types of features needed by our classifiers. We use 4 main different types of linguistic features in order to build the RE 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 types 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 (for the same sentence as in Figure 1):

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

(where V means verb, DI indefinite article, ADJ adjective, and N noun). We have to note that this kind of approach requires a huge training corpus, due to the lack of flexibility of the patterns. With this type of feature, the problem of sparse data is crucial.

Pattern Generalization: To minimize the sparse data problem, we apply an algorithm based on similarity between basic patterns, generalizing them and thus increasing the coverage of the model. In order to generalize two patterns, we check first if they are similar and then we remove all those units that they do not share. After computing the similarity between two patterns p_1 and p_2, the longest common string (*lcs*) 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. The *lcs* of two patterns is considered as their generalized pattern.

Bags of Lemmas and Tags: Instead of using a set of entire patterns as features, a common method that allows us to increase the coverage of the extractor is the utilization of smaller items, such as lemmas with PoS annotation. In this case, the example sentence (Figure 1) would generate the following features (notice again that for some categories only the tag is retained): <be_V>, <DI>, <ADJ> and <fashion_N>.

Syntactic Dependencies: We consider a subset of all the syntactic dependencies derived from the full linguistic structures. Given a sentence, two types of dependencies are retained: (i) dependencies between the two target terms (**X** or **Y**), and (ii) dependencies between one of the two target terms and one entity of the (left, middle, or right) context.

For instance, from the previous example sentence 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 token. 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. Finally, labels of dependencies (e.g., modifier, subject, etc.) are also taken into account to define the features.

Note that the analysis is very partial. In many cases, the parser is not able to complete the dependency path between the two terms. Grammars for other languages than English are often not complete or they are not freely available.

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. Furthermore, we also performed some experiments concerning the ratio of positive and negative examples in the training set. The experiments were performed with WEKA (Witten and Frank, 2005), using its implementation of the SMO algorithm. This choice was made because in preliminary experiments (using Naive Bayes, Decision Trees as well as SMO algorithms), SMO scored the best.

The training examples were obtained from the Portuguese Wikipedia with the method showed in Section 3. We focused on examples of the semantic relation "Occupation", which is a kind of *is_a* relation. We first selected about $50,000$ relation pairs from infoboxes. Then, we identified near $500,000$ sentences containing a named entity and an occupation noun, which were automatically classified as positive or negative. Finally, we randomly selected an initial set of 2000 sentences for training, 50% of them being positive examples. For testing, we randomly extracted and manually revised a set of 700 sentences (different than those used for training).

5.1 Results

The evaluated classifiers were built with the types of features explained in Section 4:

pattern-all and *pattern-mid* use the basic patterns 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.

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 (true and false positives). Recall here refers to the number of correct positive decisions divided by the total number of positive examples in the test set.

Single Types of Features: Our first experiment consists of the evaluation of seven classifiers built with the individual types of features, extracted from the training set. Table 1 let us observe that the best features are those based on lexico-syntactic patterns with middle contexts: *pattern_gen-mid* and *pattern-mid*, with f-score values between 72%/77%. The

Model	Prec.	Rec.	f-score
pattern-all	91.66%	2.65%	5.16%
pattern-mid	**94.9%**	58.45%	72.34%
pattern_gen-mid	93.26%	66.9%	**77.9%**
bow-all	74.14%	**67.87%**	70.87%
bow-mid	74.13%	31.15%	43.87%
dep-all	79.21%	48.79%	60.38%
dep-mid	76.92%	41.06%	53.54%

Table 1: Precision, Recall and f-score of 7 classifiers based on different types of linguistic features.

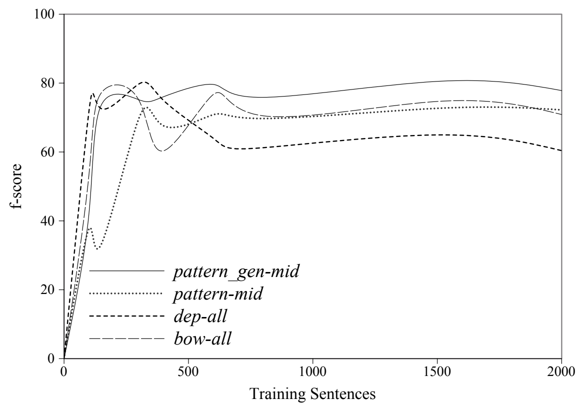

Figure 2: F-score vs training size ($0-2000$ sentences) of the 4 best features for each linguistic type.

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. By contrast, *bow* and *dep* classifiers improved their performance using *all* contexts.

Learning Curves: Figure 2, shows the f-score of the best individual features (for each of the main types) in different partitions. 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. We can also observe that, except for *pattern-mid*, f-score slightly decreases with more than 1500 examples.

The results of these tests allow us to know the performance of the classifiers based on individual types of features. In the next experiment we evaluate several combinations of individual types of features.

Similarity and Combination of Models: When analyzing the differences between the feature models, we compute the Dice similarity coefficient to

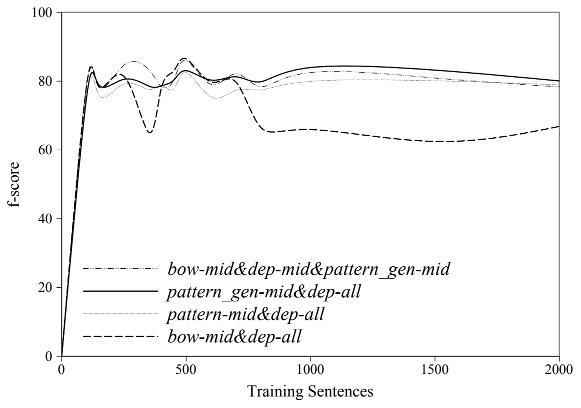

Figure 3: F-score vs training size ($0-2000$ sentences) of the 4 best combinations of features.

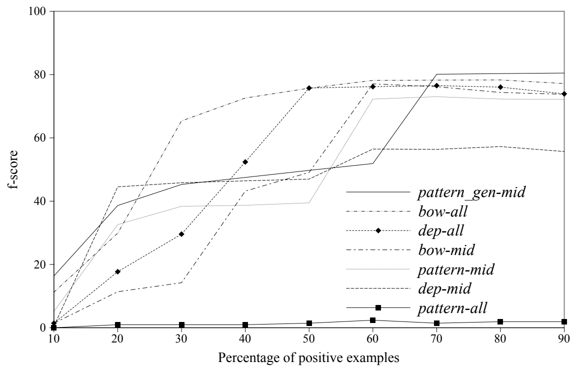

Figure 4: F-score vs P/N ratio of 7 classifiers. Training sets are 9 partitions of 500 sentences with different ratio of P/N examples (from 10%/90% to 90%/10%).

Model	**Prec.**	**Rec.**	**f-score**	**Diff.**	**P/N**
p-all	81.3%	3.1%	6.1%	+0.9	90%
p-mid	**91%**	68.4%	78.1%	+5.7	90%
p_gen-m.	90.1%	76.6%	**82.8%**	+4.9	90%
bow-all	67.9%	**87.2%**	76.3%	+5.4	70%
bow-mid	76.3%	78.5%	77.4%	+33.5	60%
dep-all	73.4%	84.1%	78.4%	+18.0	70%
dep-mid	63.5%	54.4%	57.4%	+3.9	90%

Table 2: Precision, Recall and f-score of 7 classifiers. The distribution of P/N examples in the training was adjusted for each classifier (*p* models are the *pattern* models).

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 pairs of models with low Dice coefficient were combined since they are likely to be complementary.

Figure 3 shows the results of combining the best individual features. 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. Furthermore, the best combined classifiers also improve the general f-score of the best single type of feature, *pattern_gen-mid*.

5.2 On-Going Experiments

We are evaluating the impact of negative examples in the training corpus, taking into account that the initial set of 2000 sentences had a 50%/50% ratio of positive and negative examples. In order to know the best P/N distribution, we collected several sets of sentences differing in the P/N ratio they have. Note that this kind of evaluation also deals with the amount of positive or negative instances, and not only with the P/N ratio. So, in order to avoid this effect, we performed two major experiments: (i) we automatically collected 9 sets of 500 sentences differing in the P/N ratio: from 10%/90% to 90%/10% (positive/negative) 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.

Figure 4 shows the f-score values of each model according to the P/N examples distribution. In most cases, the peak of the f-score curve is reached when the training sentences contain between 60% and 70% of positive examples, except for pattern-based features, whose performance gradually improves with more positive samples. This tendency occurs in both experiments (500 and 2000 sentences), so we can infer the best P/N ratio for each type of feature.

The results from the previous experiment provide us information to train new models with the P/N distribution adjusted to each model. So, we built classifiers based on the same 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 feature, and test them on the test set. Table 2 shows that by adjusting the P/N ratio in the training involves several differences in the performance of the models. We observed that the precision values present some decrease (namely in *pattern-all*, *bow-all* and *dep-mid*). However, since all the systems show dramatically recall improvements, the f-score of the seven classifiers increase. Column 5 in Table 2 shows the f-score differences compared to those classifiers trained with a 50%/50% P/N ratio. Column 6 shows the percentage of positive training samples for each classifier.

We have to note that with this adjustment in the P/N ratio, the performance of the best classifiers based on individual features (*pattern_gen-mid*, with 82.77% f-score) scored similar to the best combinations of features (*pattern_gen-mid & dep-all*, reaching maximum values of 83.2%).

6 Conclusions and Further Work

This paper analyzes the impact of various linguistic features in a distant-supervision system for extracting semantic relations from unstructured text.

Experiments performed in Portuguese data show that features based on lexico-syntactic patterns achieve higher precision values than those with bags of lemmas and tags or syntactic dependencies. Pattern-based models performed better with *middle* contexts than with *all* contexts, but in case of *dep* models, *all* contexts behave better. Models based on bags of lemmas and tags tend to be more unstable.

Moreover, the combination of some types of features helps to achieve a trade-off between precision and recall, improving the performance of the single features. However, we observed that the adjustment of the positive and negative examples ratio in the training set involves dramatically increases in recall.

Further experiments will analyze the performance of combinations of the single features with an adjusted training set. Moreover, we will test these classifiers with different text genres, as well as with other relations and languages.

Acknowledgments

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

References

Agichtein, Y. E. 2005. *Extracting Relations from Large Text Collections*. Ph.D. thesis, Columbia University.

Brin, S. 1998. Extracting patterns and relations from the world wide web. *WebDB Workshop at EDBT'98*: 172–183.

Bunescu, R. C. and Mooney, R. J. 2005. A Shortest Path Dependency Kernel for Relation Extraction. *Proceedings of HLT/EMNLP'05*: 724–731. ACL, Vancouver.

Fleischman, M., Hovy, E., and Echihabi, A. 2003. Offline strategies for online question answering: Answering questions before they are asked. *Proceedings of ACL'03*: 1–7.

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

Garcia, M. and Gamallo, P. 2010. Análise Morfossintáctica para Português Europeu e Galego: Problemas, Soluções e Avaliação. *Linguamática. Revista para o Processamento Automático das Línguas Ibéricas* 2 (2): 59–67.

Garcia, M. and Gamallo, P. 2011. An Exploration of the Linguistic Knowledge for Semantic Relation Extraction in Spanish. *Proceedings of Joint Workshop FAM-LbR/KRAQ'11* at IJCAI 2011. Barcelona.

Hearst, M. A. 1992. Automatic acquisition of hyponyms from large text corpora. *Proceedings of COLING'92*, 2: 539–545.

Kambhatla, N. 2004. Combining lexical, syntactic and semantic features with Maximum Entropy models for extracting relations. *Proceedings of ACL'04*.

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

Mintz, M., Bills, S., Snow, R. and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. *Proceedings of the ACL/IJCNLP*, 2.

Padró, Ll., Collado, M., Reese, S., Lloberes, M. and Castellón, I. 2010. FreeLing 2.1: Five Years of Open-Source Language Processing Tools. In *Proceedings of LREC'10*, ELRA. La Valletta, Malta.

Snow, R., Jurafsky, D. and Ng, A. Y. 2005. Learning Syntactic Patterns for Automatic Hypernym Discovery. *Advances in Neural Information Processing Systems* 17: 1297–1304.

Witten, I. H. and Frank, E. 2005. *Data mining: practical machine learning tools and techniques with java implementations*. Elsevier Inc., San Francisco.

Wu, F. and Weld, D. S. 2010. Open information extraction using Wikipedia. In *Proceedings of ACL'10*: 118–127.

Zhao, S. and R. Grishman 2005. Extracting Relations with Integrated Information Using Kernel Methods *Proceedings of ACL'05*: 419–426.

Zhou, G., Su, J., Zhang, J., and Zhang, M. 2005. Exploring Various Knowledge in Relation Extraction *Proceedings of ACL'05*: 427–434.