Supervised Learning of Syntactic Contexts for Uncovering Definitions and Extracting Hypernym Relations in Text Databases

Guido Boella and Luigi Di Caro

Department of Computer Science
University of Turin
{boella,dicaro}@di.unito.it

Abstract. In this paper we address the problem of automatically constructing structured knowledge from plain texts. In particular, we present a supervised learning technique to first identify definitions in text data, while then finding hypernym relations within them making use of extracted syntactic structures. Instead of using pattern matching methods that rely on lexico-syntactic patterns, we propose a method which only uses syntactic dependencies between terms extracted with a syntactic parser. Our assumption is that syntax is more robust than patterns when coping with the length and the complexity of the texts. Then, we transform the syntactic contexts of each noun in a coarse-grained textual representation, that is later fed into hyponym/hypernym-centered Support Vector Machine classifiers. The results on an annotated dataset of definitional sentences demonstrate the validity of our approach overtaking the current state of the art.

1 Introduction

Nowadays, there is a huge amount of textual data coming from different sources of information. Wikipedia[1], for example, is a free encyclopedia that currently contains 4,168,348 English articles[2]. Even Social Networks play a role in the construction of data that can be useful for Information Extraction tasks like Sentiment Analysis, Question Answering, and so forth. From another point of view, there is the need of having more structured data in the forms of ontologies, in order to allow semantics-based retrieval and reasoning. Ontology Learning is a task that permits to automatically (or semi-automatically) extract structured knowledge from plain text. Manual construction of ontologies usually requires strong efforts from domain experts, and it thus needs an automatization in such sense. In this paper, we focus on the extraction of hypernym relations. The first step of such task relies on the identification of what [21] called *definitional sentences*, i.e., sentences that contain at least one hypernym relation. This subtask is important by itself for many tasks like Question Answering [8], construction

[1] http://www.wikipedia.org/
[2] February 19, 2013.

H. Blockeel et al. (Eds.): ECML PKDD 2013, Part II, LNAI 8189, pp. 64–79, 2013.

of glossaries [18], extraction of taxonomic and non-taxonomic relations [19,25], enrichment of concepts [13,6], and so forth.

The main contribution of this work is to relax the problem of inducing semantic relations by separating it into two easier subtasks and facing them independently with the most appropriate techniques. Indeed, hypernym relation extraction involves two aspects: linguistic knowlege, and model learning. Patterns uses to collapse both of them, preventing to tackle them separately with the most suitable techniques. On the one hand, patterns have limited expressivity, while linguistic knowledge inside patterns is learned from small corpora so it is likely to have low coverage. On the other hand, classification strictly depends on the learned patterns, so performance decreases and the available classification techniques are restricted to those compatible with the pattern approach. Instead, we use a syntactic parser for the first aspect (with all its native and domain-independent knowledge on language expressivity), and a state-of-the-art approach to learn models with the use of Support Vector Machine classifiers.

2 Motivating Examples

Most of the existing work in this field use manual rather than automatic generation of sequential patterns inducing hypernym relations. Although this approach achieves good results, as demonstrated in well-founded papers like [8] and [21], it is limited in the sense that it exclusively relies on the sequentiality of the expressions. Natural language offers potentially infinite ways of expressing concepts, without any limits on the length and complexity of the sentences.

Definitions can present a great variety of linguistic constructions in natural language. For instance, it is possible to have definitions that make use of punctuation, as in the following sentence:

> "*IP: a protocol for sending data across a network.*"

Some work concentrated on the English copular verb *to be*, as in [3], while even other verbs can indicate the presence of a hypernym relation, as in the sentence below:

> "*The term ontology stands for a formal representation of objects in a specific domain.*"

Still, in the sentence:

> "*Browsers, tools for navigating the Web, can also reproduce sound.*"

the identification of the syntactic apposition is necessary to determine the hypernym relation. Modifiers are other linguistic constructions that can lengthen the sentences, making them more complex to match with trained patterns, as in this example:

> "*The Aardvark is a medium-sized, burrowing, nocturnal mammal native to Africa*"

Notice that, however, a POS-aware pattern mathing technique like the one by [21] can handle this type of complexity. Finally, linguistic coordinations need to be identified to be able to extract all possible hypernyms, as in this last example:

"*Agathon was an Athenian tragic poet and friend of Euripides and Plato*"

where "*Agathon*" is a "*poet*", a "*friend of Euripides*", and a "*friend of Plato*".

3 Related Work

In this section we present the current state of the art concerning the automatic extraction of definitions and hypernym relations from plain text.

3.1 Definition Classification

Considering the initial formal representation proposed by [26], a *definitional sentence* is composed by different information fields:

- a *definiendum* (DF), i.e., the word being defined with its modifiers,
- a *definitor* (VF), i.e., the verb phrase to introduce the definition,
- a *definiens* (GF), i.e., the genus phrase that usually contains the hypernym,
- and the *rest* of the sentence (REST), that can contain additional clauses.

An example of annotated definition is represented by the following sentence:

[In computer science, a $[pixel]_{DF}$ $[is]_{VF}$ [a **dot**$]_{GF}$ [that is part of a computer image$]_{REST}$.

In this paper, we will use the term *definitional sentence* referring to the more general meaning given by [21]: *A sentence that provides a formal explanation for the term of interest*, and more specifically as a sentence containing at least one hypernym relation.

So far, most of the proposed techniques rely on lexico-syntactic patterns, either manually or semi-automatically produced [17,31,28]. Such patterns are sequences of words like "*is a*" or "*refers to*", rather than more complex sequences including part-of-speech tags.

In the work of [28], after a manual identification of types of definitions and related patterns contained in a corpus, the author successively applied Machine Learning techniques on syntactic and location features to improve the results.

A fully-automatic approach has been proposed by [3], where the authors applied genetic algorithms to the extraction of English definitions containing the keyword "*is*". In detail, they assign weights to a set of features for the classification of definitional sentences, reaching a precision of 62% and a recall of 52%.

Then, [8] proposed an approach based on *soft patterns*, i.e., probabilistic lexico-semantic patterns that are able to generalize over rigid patterns enabling

partial matching by calculating a generative degree-of-match probability between a test instance and the set of training instances.

[10] used three different Machine Learning algorithms to distinguish actual definitions from other sentences, relying on syntactic features and reaching high accuracy levels.

The work of [18] relies on a rule-based system that makes use of "cue phrases" and structural indicators that frequently introduce definitions, reaching 87% of precision and 75% of recall on a small and domain-specific corpus.

Finally, [21] proposed a system based on Word-Class Lattices (WCL), i.e., graph structures that try to generalize over the POS-tagged definition patterns found in the training set. Nevertheless, these mechanisms are not properly able to handle linguistic exceptions and linguistic ambiguity.

3.2 Hypernym Extraction

According to [2] and [4], the problem of extracting ontologies from text can be faced at different levels of granuarity. According to the former, our approach belongs to the extraction of *terminological ontologies* based on IS-A relations, while for the latter we refer to the *concept hierarchies* of their *Ontology Learning layer cake*.

As for the task of definition extraction, most of the existing approaches use symbolic methods that are based on lexico-syntactic patterns, which are manually crafted or deduced automatically. The seminal work of [16] represents the main approach based on fixed patterns like "NP_x *is a/an* NP_y" and "NP_x *such as* NP_y", that usually imply $< x$ IS-A $y >$. The main drawback of such technique is that it does not face the high variability of how a relation can be expressed in natural language. Still, it generally extracts single-word terms rather than well-formed and compound concepts. The work of [21], as already mentioned in the previous section, is based on graph structures that generalize over the POS-tagged patterns between x and y. [1] proposed similar lexico-syntactic patterns to extract *part-whole* relationships.

[9] proposed a rule-based approach to the extraction of hypernyms that, however, leads to very low accuracy values in terms of Precision.

[23] proposed a technique to extract hypernym relations from Wikipedia by means of methods based on the connectivity of the network and classical lexico-syntactic patterns. [29] extended their work by combining extracted Wikipedia entries with new terms contained in additional web documents, using a distributional similarity-based approach.

Finally, pure statistical approaches present techniques for the extraction of hierarchies of terms based on words frequency as well as co-occurrence values, relying on clustering procedures [5,12,30]. The central hypothesis is that similar words tend to occur together in similar contexts [15]. Despite this, they are defined by [2] as *prototype-based ontologies* rather than formal terminological ontologies, and they usually suffer from the problem of data sparsity in case of small corpora.

4 Approach

In this section we present our approach to identify hypernym relations within plain text. Our methodology consists in relaxing the problem in two different subtasks. Given a semantic relation between two terms $rel(x, y)$ within a sentence, the task becomes to find 1) a possible x, and 2) a possible y. In case of more than one possible x or y, a further step is needed to associate the correct x to the right y.

By seeing the problem as two different classification problems, there is no need to create abstract patterns between the target terms x and y. In addition to this, the general problem of identifying definitional sentences can be seen as to find at least one x and one y in a single sentence.

4.1 Local Syntactic Information

Dependency parsing is a procedure that extracts syntactic dependencies among the terms contained in a sentence. The idea is that, given a hypernym relation, hyponyms and hypernyms may be characterized by specific sets of syntactic contexts. According to this assumption, the task can be seen as a classification problem where each noun in a sentence has to be classified as hyponym, hypernym, or neither of the two.

More in detail, for each noun "a" the system creates one instance composed by textual items describing its syntactic context. Each item can be seen as a classic word, and it represents a single syntactic relation taking a as one of its arguments. To build these items, extracted dependencies are transformed into abstract textual representation in the form of triples. In particular, for each syntactic dependency $dep(a, b)$ (or $dep(b, a)$) of a target noun "a", we create an abstract term *dep-target-*$\hat{b}$ (or *dep-*$\hat{b}$*-target*), where "a" becomes "*target*" and where "$\hat{b}$" is transformed into the generic string "*noun*" in case it is a noun; otherwise it is equal to "b". This way, the nouns are transformed into coarse-grained context abstractions, creating a level of generalization of the feature set that collapses the variability of the nouns involved in the syntactic dependencies. The string "*target*" is useful to determine the exact position of the noun in a syntactic dependency (as a left argument, or as a right argument).

Running Example. In order to describe the process of tranforming the input data to fit with a standard classification problem, we present here a step-by-step concrete example. Let us consider the sentence below:

> *The Albedo of an object is the extent to which it diffusely reflects light from the sun.*

The result of the Part-Of-Speech tagging procedure will be the following:

> The/*DT* Albedo/*NNP* of/*IN* an/*DT* object/*NN* is/*VBZ* the/*DT* extent/*NN* to/*TO* which/*WDT* it/*PRP* diffusely /*RB* reflects/*VBZ* light/*NN* from/*IN* the/*DT* sun/*NN*.

where *DT* stands for *determiner*, *NNP* is a proper name, and so on[3]. Then, the syntactic parsing will produce the following dependencies (the numbers are unique identifiers)[4]:

det(Albedo-2, The-1)
nsubj(extent-8, Albedo-2)
det(object-5, an-4)
prepof(Albedo-2, object-5)
cop(extent-8, be-6)
det(extent-8, the-7)
rel(reflects-13, which-10)
nsubj(reflects-13, it-11)
advmod(reflects-13, diffusely-12)
rcmod(extent-8, reflect-13)
dobj(reflect-13, light-14)
det(sun-17, the-16)
prepfrom(reflect-13, sun-17)

where the dependency *nsubj* represents a noun phrase which is the syntactic subject of a clause, *dobj* identifies a noun phrase which is the (accusative) object of the verb, and so on[5]. The related parse tree is shown in Figure 1.

At this point, the system creates one instance for each term labeled as "noun" by the POS-tagger. For example, for the noun "*Albedo*", the instance will be represented by three abstract terms, as shown in Table 1.

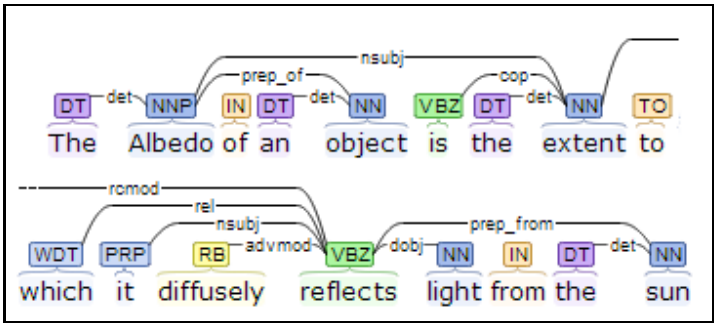

Fig. 1. The resulting parse tree of the example

Once the instance for the noun "*Albedo*" is created, it is passed to the classification process that will decide if "*Albedo*" can be considered as part of a

[3] A complete overview of the parts-of-speech can be found at `http://nlp.stanford.edu/software/tagger.shtml`

[4] We used the Stanford Syntactic Parser available at `http://nlp.stanford.edu/software/lex-parser.shtml`

[5] A complete overview of the Stanford dependencies is available at `http://nlp.stanford.edu/software/dependencies_manual.pdf`

Table 1. The instance created for the noun "*Albedo*" is composed by three items (one for each syntactic dependency related to "*Albedo*"). Note that the considered noun "*Albedo*" is replaced by the generic term "**target**", while the other nouns are replaced with "*noun*".

Dependence	Instance Item
det(Albedo, The)	det-**target**-the
nsubj(extent, Albedo)	nsubj-*noun*-**target**
prepof(Albedo, object)	prepof-**target**-*noun*

hypernym relation, as explained in the next section. This is done for each noun in a sentence.

4.2 Learning Phase

Our model assumes a transformation of the local syntactic information into labelled numeric vectors. More in detail, given a sentence S annotated with some terms linked by one hypernym relation, the system produces as many input instances as the number of nouns contained in S. Only the nouns that are involved in the annotated hypernym relation (as x or y in $rel(x, y)$) will be positive instances.

More specifically, for each noun n in S, the method produces two instances S_x^n and S_y^n (i.e., one for each argument of a hypernym relation). The difference between the two will be only the class label:

1. If $n = x$ in $rel(x, y)$, $label(S_x^n) = positive$, and $label(S_y^n) = negative$
2. If $n = y$ in $rel(x, y)$, $label(S_x^n) = negative$, and $label(S_y^n) = positive$
3. If $n \neq x \wedge n \neq y$ in $rel(x, y)$, $label(S_x^n) = negative$, and $label(S_y^n) = negative$

If a noun is not involved in a hypernym relation, both the two instances will have the label *negative*. At the end of this process, two training sets are built, i.e., one for each relation argument, namely the x-set and the y-set. All the instances of both the datasets are then transformed into numeric vectors according to the Vector Space Model [24], and are finally fed into a Support Vector Machine classifier[6] [7]. We refer to the two resulting models as the x-model and the y-model. These models are binary classifiers that, given the local syntactic information of a noun, estimate if it can be respectively an x or a y in a hypernym relation.

Once the x-model and the y-model are built, we can both classify definitional sentences and extract hypernym relations. In the next section we deepen our proposed strategy in that sense.

[6] We used the Sequential Minimal Optimization implementation of the Weka framework [14].

Running Example. We present here a complete example of the learning phase. In detail, all the nouns contained in all the sentences of the dataset are transformed into textual instances, as shown in Table 1. The result of the sentence illustrated in the previous section is shown in Table 2.

Table 2. The instances created for the sentence of the example (one for each noun). Note that the nouns "*Albedo*" and "*extent*" are labeled as x and y respectively, as in the annotated dataset. The nouns "*object*", "*light*" and "*sun*" are negative examples for both the x-set and the y-set that will be used by the classifier for learning the models.

noun	**Instance**	**x**	**y**
Albedo	det-**target**-the nsubj-*noun*-**target** prepof-**target**-*noun*	+	-
extent	nsubj-**target**-*noun* cop-**target**-be det-**target**-the rcmod-**target**-reflect	-	+
object	det-**target**-a prepof-*noun*-**target**	-	-
light	dobj-reflect-**target**	-	-
sun	det-**target**-the prepfrom-reflect-**target**	-	-

The whole set of instances of all the sentences are fed into two Support Vector Machine classifiers, one for each target label (i.e., x and y).At this point, it is possible to classify each term as possible x or y by querying the respective classifiers with its local syntactic information.

4.3 Classification of Definitional Sentences

As already mentioned in previous sections, we label as *definitional* all the sentences that contain at least one noun n classified as x, and one noun m classified as y (where $n \neq m$). In this phase, it is not further treated the case of having more than one x or y in one single sentence. Thus, given an input sentence:

1. we extract all the nouns (POS-tagging),
2. we extract all the syntactic dependencies of the nouns (dependency parsing),
3. we classify each noun (i.e., its instance) with the x-model and to the y model,
4. we check if there exist at least one noun classified as x and one noun classified as y: in this case, we classify the sentences as *definitional*.

4.4 Extraction of Hypernym Relations

Our method for extracting hypernym relations makes use of both the x-model and the y-model as for the the task of classifying definitional sentences. If exactly one x and one y are identified in the same sentence, they are directly connected and the relation is extracted. The only constraint is that x and y must be connected within the same parse tree.

Now, considering our target relation $hyp(x, y)$, in case the sentence contains more than one noun that is classified as x (or y), there are two possible scenarios:

1. there are actually more than one x (or y), or
2. the classifiers returned some false positive.

Up to now, we decided to keep all the possible combinations, without further filtering operations[7]. Finally, in case of multiple classification with both x and y (i.e., if there are multiple x and multiple y at the same time, the problem becomes to select which x is linked to which y. To do this, we simply calculate the distance between these terms in the parse tree (the closer the terms, the better the connection between the two). Nevertheless, in the used corpus, only around 1.4% of the sentences are classified with multiple x and y.

5 Evaluation

In this section we present the evaluation of our approach, that we carried out on an annotated dataset of definitional sentences [22]. The corpus contains 4,619 sentences extracted from Wikipedia, where 1,908 are annotated as *definitional*. On a first instance, we test the classifiers on the extraction of hyponyms (x) and hypernyms (y) from the definitional sentences, independently. Then, we evaluate the classification of definitional sentences. Finally, we evaluate the ability of our technique when extracting whole hypernym relations. With the used dataset, the constructed training sets for the two classifiers (x-set and y-set) resulted to have approximately 1.5k features.

5.1 Dataset Problems

In this section we present some problems we encountered in the dataset provided by [22]. In these cases, however, we decided not to remove such sentences from the data, in order to be fully compliant with the results obtained by [21].

Incorrect Relationships. We found relationships that were different from the target one, i.e., IS-A, or that were incorrectly annotated in general. For example, considering the sentence:

→ A *hull* is the **body of a ship** or **boat**.

the annotation indicates two hypernyms for the term "*hull*", namely "**body of a ship**" and "**boat**". First, this can be more correctly seen as a *part-whole* relationship. Then, the second relation <*hull* IS-A **boat**> is incorrect[8].

Incorrect Hypernyms. Some sentences present incorrectly annotated hypernyms. For instance, let us consider the following sentence:

[7] We only used the constraint that x has to be different from y.
[8] This is due to the untreated linguistic coordination between "ship" and "boat".

→ An *actor* is defined both as the person who originates or gives existence to anything and as **one** who sets forth written statements in the Oxford English Dictionary.

where *italic* and **bold** represent the hyponym and the hypernym respectively. In this case, the IS-A relationships should have linked the term *actor* with the hypernym **person**, while the chosen hypernym **one** seems quite forced.

Missing Hypernyms. Some sentences provide partial annotations, like:

→ In Greco-Roman mythology, *Aeneas* was a Trojan **hero**, the son of prince Anchises and the goddess Aphrodite.

where only <*Aeneas* IS-A **hero**> has been annotated, while also <*Aeneas* is-a **son of Anchises**> and <Aeneas is-a **son of Aphrodite**> can be part of the annotation.

Fixed Hyponym. In each sentence, only one hyponym has been annotated, while the data actually contain sentences with more than one possible hyponym. For this reason, we could not correctly evaluate and compare our Precision and Recall values if we did not fix the hyponym during the automatc construction of the relation, in case of multiple choices.

Misaligned Modifiers and Matching Strategy. The evaluation has been carried out only looking for substring matching between the manual annotation and the estimation given by our system (as also done by [21]), since it seems that no guideline has been given for the annotators during the annotation phase. In fact, identical cases were annotated differently in terms of inclusion/exclusion of noun modifiers. For instance, the following two similar instances present different annotations:

→ *Argon* is a chemical **element** designated by the symbol Ar.

→ An *acid* (often represented by the generic formula HA) is traditionally considered as any **chemical compound** that, when dissolved in water, (...)

where *italic* and **bold** represent the hyponym and the hypernym respectively. In the first case only the noun has been marked as hypernym, while in the second case even the modifier has been included. Notice that, in such case, both the two modifiers present the same degree of information, so they should have been identically annotated.

Finally, since our method is able to extract single nouns that can be involved in a hypernym relation, we included modifiers preceded by preposition "*of*", while the other modifiers are removed, as shown by the extracted hypernym relation of the following sentence:

→ An *Archipelago* is a **chain of islands**.

where the whole chunk **chain of islands** has been extracted, from the single triggered noun **chain**.

5.2 Results

In this section we present the evaluation of our technique on both the tasks of classifying definitional sentences and extracting hypernym relations. Notice that our approach is susceptible from the errors given by the POS-tagger and the syntactic parser. In spite of this, our approach demonstrates how syntax can be more robust for identifying semantic relations. Our approach does not make use of the full parse tree, thus we are not dependent on a complete and correct result of the parser.

The goal of our evaluation is threefold: first, we evaluate the ability of the proposed approach to classify single hypernyms or hyponyms by means of their (Bag-Of-Words tranformed) local syntactic information; then, we evaluate the ability of classifying definitional sentences; finally, we measure the accuracy of the hypernym relation extraction.

Table 3. Accuracy levels for the classification of single hyponyms (x) and hypernyms (y) using their local syntactic context, in terms of Precision (P), Recall (R), and F-Measure (F), using 10-folds cross validation

Target	P	R	F
x	93.85%	79.04%	85.81%
y	82.26%	76.77%	79.42%

In the first phase, no x-to-y linking procedure is evaluated. Table 3 shows the results, in terms of Precision, Recall, and F-Measure. As can be noticed, the approach is able to identify correct x and y with high accuracy. Interestingly, hyponyms seem to have more stable syntactic contexts rather than hypernyms. Moreover, while Recall seems to be quite similar between the two, Precision is much higher (+11.6%) for the extraction of hyponyms.

While these results demonstrate the potential of the approach, it is interesting to analyze which syntactic information frequently reveal hyponyms and hypernyms. Table 4 shows the top 10 most important features for both the x and the y in a $hyp(x, y)$ relation, computing the value of the chi-squared statistics with respect to the class (x and y, respectively). A part from dataset-specific features like *amod-target-geologic* (marked in *italic*), many interesting considerations can be done by looking at Table 4. For example, the syntactic dependency *nsubj* results to be important for the identification of both hyponyms and hypernyms. The formers, in fact, are often syntactic subjects of a clause, and vice versa for the latters. Interestingly, *nsubj-noun-target* (marked in **bold** in Table 4) is important to both identify a correct hyponym and to reveal that a noun *is not* a

Table 4. The top 10 most relevant features for the classification of single hyponyms and hypernyms from a sentence, computing the value of the chi-squared statistic with respect to the class (x and y, respectively). The feature "nsubj-noun-target" (marked in **bold**) is important to identify a correct hyponym and to estimate that a noun *is not* a hypernym, while this seems not true for "nsubj-target-noun". Clear dataset-specific features are marked in *italic*.

Top Features for x	Top Features for y
nsubj-noun-target	cop-target-be
det-target-a	nsubj-target-noun
nsubj-refer-target	det-target-a
cop-target-be	prepin-target-noun
nsubj-target-noun	**nsubj-noun-target**
prepof-noun-target	partmod-target-use
prepof-target-noun	prepto-refer-target
nn-noun-target	prepof-target-noun
det-noun-a	det-target-any
nsubjpass-define-target	*amod-target-geologic*

Table 5. Evaluation results for the classification of definitional sentences, in terms of Precision (P), Recall (R), F-Measure (F), and Accuracy (Acc), using 10-folds cross validation

Algorithm	P	R	F	Acc
WCL-1 [21]	**99.88%**	42.09%	59.22 %	76.06 %
WCL-3 [21]	98.81%	60.74%	75.23 %	83.48 %
Star Patterns [21]	86.74%	66.14%	75.05 %	81.84 %
Bigrams [8]	66.70%	**82.70%**	73.84 %	75.80 %
Our approach	88.09%	76.01%	**81.61%**	**89.67%**

hypernym (*nsubj-noun-target* is present in both the two columns x and y), while this seems not true for *nsubj-target-noun* (it is only important to say if a noun can be a hypernym, and not to say if such noun *is not* a hyponym).

A definitional sentences is extracted only if at least one x and one y are found in the same sentence. Table 5 shows the accuracy of the approach for this task. As can be seen, our proposed approach has a high Precision, with a high Recall. Although Precision is lower than the pattern matching approach proposed by [21], our Recall is higher, leading to an higher F-Measure.

Table 6 shows the results of the extraction of the whole hypernym relations. We also added the performance of a system named "Baseline", which implements our strategy but only using the POS tags of the nouns' neighbor words instead of their syntactic dependencies. Its low effectiveness demonstrates the importance of the syntactic information, independently from the learning phase. Finally, note that our approach reached high levels of accuracy. In particular, our system outperforms the pattern matching algorithm proposed by [21] in terms of both Precision and Recall.

Table 6. Evaluation results for the hypernym relation extraction, in terms of Precision (P), Recall (R), and F-Measure (F). These results are obtained using 10-folds cross validation (* Recall has been inherited from the definition classification task).

Algorithm	P	R	F
WCL-1 [21]	77.00%	42.09% *	54.42%
WCL-3 [21]	78.58%	60.74% *	68.56%
Baseline	57.66%	21.09%	30.76%
Our approach	**83.05%**	**68.64%**	**75.16%**

5.3 Further Considerations

The data provided by [22] also contain a dataset of over 300,000 sentences retrieved from the UkWac Corpus [11]. Unfortunately, Precision was only manually validated, therefore we could not be able to make any fair comparison. Nevertheless, they made available a subset of 99 definitional sentences. On such data, our technique obtained a Recall of 59.6% (59 out of 99), while their approaches reached 39.4%, 56.6%, and 63.6% respectively for WCL-1, WCL-3, and Star Patterns.

In the dataset, the syntactic parser found hundreds of cases of coordinated hyponyms, while the annotation provides only one hyponym for each sentence. For this reason, we were not able to evaluate our method on the extraction of all possible relations with all coordinated hyponyms.

The *really-desired* result of the task of extracting hypernym relations from text (as for any semantic relationships in general) depends on the domain and the specific later application. Thus, we think that a precise evaluation and comparison of any systems strictly depends on these factors. For instance, given a sentence like:

> → In mathematics, computing, linguistics and related disciplines, an *algorithm* is a sequence of instructions.

one could want to extract only "**instructions**" as hypernym (as done in the annotation), rather than the entire chunk "**sequence of instructions**" (as extracted by our technique). Both results can be valid, and a further discrimination can only be done if a specific application or use of this knowlege is taken into consideration.

In this work, we only suggest how syntax can be more robust for identifying semantic relations, avoiding general discussions on the growth of web data and the complexity / noise of the contents deriving from personal blogs and social network communities. Nevertheless, we are not dependent on a complete and correct result of the parser. For example, we could apply our methodology to the result of simple chunk parsers. Still, to the best of our knowledge, no other work considers noisy data on this specific task, and we based our idea thinking on encyclopedic and formal texts, where syntax is less subjected to language inflections and the need to support semantic resources construction is even more tangible.

6 Conclusion and Future Work

We presented an approach to reveal definitions and extract underlying hypernym relations from plain text, making use of local syntactic information fed into Support Vector Machine classifiers. The aim of this work was to revisit these tasks as classical supervised learning problems that usually carry to high accuracy levels with high performance when faced with standard Machine Learning techniques. Our approach demonstrates that relaxing the problem into two different subtasks can actually improve the identification of hypernym relations. Nevertheless, this could not be true for any possible semantic relations, since semantics is independent from syntax to a certain extent. The results of the presented approach highlight its validity by significantly improving current state-of-the-art techniques in the classification of definitional sentences as well as in the extraction of hypernym relations from text. In future works, we aim at using larger syntactic contexts as well as additional semantic information. Despite the successful results, we plan to also examine the context between x and y in order to further strengthen the technique. Then, the problem of finding meaningful noun modifiers as part of the entities involved in hypernym relations needs to be studied carefully, starting from a task-specific annotated corpus. Finally, we aim at evaluating our approach on the construction of entire taxonomies relying on domain-specific text corpora, as in [20,27].

References

1. Berland, M., Charniak, E.: Finding parts in very large corpora. In: Annual Meeting Association for Computational Linguistics, vol. 37, pp. 57–64. Association for Computational Linguistics (1999)
2. Biemann, C.: Ontology learning from text: A survey of methods. LDV Forum 20, 75–93 (2005)
3. Borg, C., Rosner, M., Pace, G.: Evolutionary algorithms for definition extraction. In: Proceedings of the 1st Workshop on Definition Extraction, pp. 26–32. Association for Computational Linguistics (2009)
4. Buitelaar, P., Cimiano, P., Magnini, B.: Ontology learning from text: An overview. Ontology Learning from Text: Methods, Evaluation and Applications 123, 3–12 (2005)
5. Candan, K., Di Caro, L., Sapino, M.: Creating tag hierarchies for effective navigation in social media. In: Proceedings of the 2008 ACM Workshop on Search in Social Media, pp. 75–82. ACM (2008)
6. Cataldi, M., Schifanella, C., Candan, K.S., Sapino, M.L., Di Caro, L.: Cosena: a context-based search and navigation system. In: Proceedings of the International Conference on Management of Emergent Digital EcoSystems, p. 33. ACM (2009)
7. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
8. Cui, H., Kan, M.Y., Chua, T.S.: Soft pattern matching models for definitional question answering. ACM Trans. Inf. Syst. 25(2) (April 2007), http://doi.acm.org/10.1145/1229179.1229182

9. Del Gaudio, R., Branco, A.: Automatic extraction of definitions in portuguese: A rule-based approach. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS (LNAI), vol. 4874, pp. 659–670. Springer, Heidelberg (2007)
10. Fahmi, I., Bouma, G.: Learning to identify definitions using syntactic features. In: Proceedings of the EACL 2006 Workshop on Learning Structured Information in Natural Language Applications, pp. 64–71 (2006)
11. Ferraresi, A., Zanchetta, E., Baroni, M., Bernardini, S.: Introducing and evaluating ukwac, a very large web-derived corpus of english. In: Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can We Beat Google, pp. 47–54 (2008)
12. Fortuna, B., Mladenič, D., Grobelnik, M.: Semi-automatic construction of topic ontologies. In: Ackermann, M., et al. (eds.) EWMF 2005 and KDO 2005. LNCS (LNAI), vol. 4289, pp. 121–131. Springer, Heidelberg (2006)
13. Gangemi, A., Navigli, R., Velardi, P.: The ontowordnet project: Extension and axiomatization of conceptual relations in wordnet. In: Meersman, R., Schmidt, D.C. (eds.) CoopIS 2003, DOA 2003, and ODBASE 2003. LNCS, vol. 2888, pp. 820–838. Springer, Heidelberg (2003), http://dx.doi.org/10.1007/978-3-540-39964-3_52
14. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. ACM SIGKDD Explorations Newsletter 11(1), 10–18 (2009)
15. Harris, Z.: Distributional structure. Word 10(23), 146–162 (1954)
16. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th Conference on Computational Linguistics, vol. 2, pp. 539–545. Association for Computational Linguistics (1992)
17. Hovy, E., Philpot, A., Klavans, J., Germann, U., Davis, P., Popper, S.: Extending metadata definitions by automatically extracting and organizing glossary definitions. In: Proceedings of the 2003 Annual National Conference on Digital Government Research, pp. 1–6. Digital Government Society of North America (2003)
18. Klavans, J., Muresan, S.: Evaluation of the definder system for fully automatic glossary construction. In: Proceedings of the AMIA Symposium, p. 324. American Medical Informatics Association (2001)
19. Navigli, R.: Using cycles and quasi-cycles to disambiguate dictionary glosses. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, pp. 594–602. Association for Computational Linguistics (2009)
20. Navigli, R., Velardi, P., Faralli, S.: A graph-based algorithm for inducing lexical taxonomies from scratch. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, vol. 3, pp. 1872–1877. AAAI Press (2011)
21. Navigli, R., Velardi, P.: Learning word-class lattices for definition and hypernym extraction. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 1318–1327. Association for Computational Linguistics, Uppsala (2010), http://www.aclweb.org/anthology/P10-1134
22. Navigli, R., Velardi, P., Ruiz-Martnez, J.M.: An annotated dataset for extracting definitions and hypernyms from the web. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010). European Language Resources Association (ELRA), Valletta (2010)
23. Ponzetto, S., Strube, M.: Deriving a large scale taxonomy from wikipedia. In: Proceedings of the National Conference on Artificial Intelligence, vol. 22, p. 1440. AAAI Press, MIT Press, Menlo Park, Cambridge (2007)

24. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM 18(11), 613–620 (1975), http://doi.acm.org/10.1145/361219.361220
25. Snow, R., Jurafsky, D., Ng, A.: Learning syntactic patterns for automatic hypernym discovery. In: Advances in Neural Information Processing Systems 17 (2004)
26. Storrer, A., Wellinghoff, S.: Automated detection and annotation of term definitions in german text corpora. In: Proceedings of LREC, vol. 2006 (2006)
27. Velardi, P., Faralli, S., Navigli, R.: Ontolearn reloaded: A graph-based algorithm for taxonomy induction. Computational Linguistics, 1–72 (2012)
28. Westerhout, E.: Definition extraction using linguistic and structural features. In: Proceedings of the 1st Workshop on Definition Extraction, WDE 2009, pp. 61–67. Association for Computational Linguistics, Stroudsburg (2009), http://dl.acm.org/citation.cfm?id=1859765.1859775
29. Yamada, I., Torisawa, K., Kazama, J., Kuroda, K., Murata, M., De Saeger, S., Bond, F., Sumida, A.: Hypernym discovery based on distributional similarity and hierarchical structures. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, vol. 2, pp. 929–937. Association for Computational Linguistics (2009)
30. Yang, H., Callan, J.: Ontology generation for large email collections. In: Proceedings of the 2008 International Conference on Digital Government Research, pp. 254–261. Digital Government Society of North America (2008)
31. Zhang, C., Jiang, P.: Automatic extraction of definitions. In: 2nd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2009, pp. 364–368 (August 2009)