Hypernym Extraction: Combining Machine-Learning and Dependency Grammar

Luis Espinosa-Anke, Francesco Ronzano, and Horacio Saggion

TALN - Universitat Pompeu Fabra
C/Tànger, 122-134, 08018 Barcelona
{luis.espinosa,francesco.ronzano,horacio.saggion}@upf.edu

Abstract. Hypernym extraction is a crucial task for semantically motivated NLP tasks such as taxonomy and ontology learning, textual entailment or paraphrase identification. In this paper, we describe an approach to hypernym extraction from textual definitions, where machine-learning and post-classification refinement rules are combined. Our best-performing configuration shows competitive results compared to state-of-the-art systems in a well-known benchmarking dataset. The quality of our features is measured by combining them in different feature sets and by ranking them by their Information Gain score. Our experiments confirm that both syntactic and definitional information play a crucial role in the hypernym extraction task.

1 Introduction

Hypernym Extraction is the task to identify (hyponym, hypernym) relations in naturally-occurring text. For example, given the sentence "A mosque is a place of worship for followers of Islam", the objective is formalize an *is-a* relation between "mosque" and "place of worship". Such task is important for structuring knowledge hierarchically [1]. It is an appealing task in NLP applications such as Named Entity Recognition [2], Query Refinement [3], Image Classification [4], Taxonomy Learning [5], Question Answering [6], Automatic Glossary Construction [7], Ontology Learning [5] or Textual Entailment [8]. Two clear examples of its importance are: (1) The WordNet hierarchy [9], where senses are organized according to "is-a" relations, and (2) The Wikipedia BiTaxonomy Project [10], which produced a *taxonomized* version of Wikipedia, and which is based on a first step on Definition Parsing and Hypernym Extraction.

In this paper we present a set of experiments for hypernym extraction and report results that outperform state-of-the art systems in the WCL (Word-Class Lattices) dataset, a well-known benchmarking dataset of textual definitions from Wikipedia where term and hypernym are manually annotated [11]. We cast our approach as a sequential classification task where, for each word in a definition, the goal is to predict whether it is at the beginning, outside or inside a hypernym (which can be a single or a multiword phrase).

The main contribution of our paper is a set of experiments over a standard benchmarking dataset for hypernym extraction achieving state-of-the-art performance, by combining linguistic, definitional and graph-based information.

A. Gelbukh (Ed.): CICLing 2015, Part I, LNCS 9041, pp. 372–383, 2015.
DOI: 10.1007/978-3-319-18111-0_28

The remainder of this paper is structured as follows: Section 2 reviews prominent work carried out in this area; Section 3 describes the linguistic motivation behind this work; Section 3.2 details the features and feature sets used in our experiments; Section 4 shows (1) a comparative evaluation across feature sets, (2) a comparative evaluation with results reported in previous work and (3) a feature relevance discussion; and Section 5 summarizes this article and outlines directions for future work.

2 Background

Textual patterns constitute the backbone of the earliest works in inducing semantic relations between words [8]. Examples widely referred to in the literature include Hearst's lexical patterns (such as "NP and other NP") [12]. Moreover, [13] propose to automatically acquire a vast large number of lexico-syntactic patterns and apply them to the newswire domain. Another well-known example is the use of Robust Minimal Recursion Semantics for semantic pattern matching [14].

In general, the literature agrees on the fact that semantic relations like hypernymy show enough variability to make pure pattern-based approaches inefficient since these patterns are either noisy by nature, as the case of *is a*, or too domain-specific and therefore impossible to generalize across domains or genres.

For this reason, machine-learning and more recently purely distributional approaches have contributed to the task of hypernym discovery. Among the former, the system described by [11] learns generalized lexico-syntactic patterns which are used to maximize the score of candidate definition sentences and, within definitions, hypernymic phrases. Moreover, [15] explored the role of syntactic dependencies as features for an SVM-based classifier. This last method is conceptually similar to ours since raw text is modelled in terms of linguistic dependencies. We extend their approach by exploiting definitional and graph-based information, which contribute to improving the performance of the system.

Distributional approaches are also becoming increasingly popular. For example, [1] describe a hypernym-discovery system for Chinese based on the notion of word-embeddings, i.e. the observation that semantically related words have common contexts at different window sizes. They propose to train a *Skip-gram* and a *CBOW* model following [16], where they take into account the embedding offsets between hyponym-hypernym pairs, and from there a projection training is designed in order to find the best hypernym for a given hyponym.

On the other hand, [8] describe a set of experiments in which they explore the veracity of the *Distributional Inclusion Hypothesis*, which states that specific terms appear in distributional contexts that are a subset of more general but related distributional contexts of more general words.

3 Modelling the Data

In the linguistic theory of Dependency Grammar, a syntactic structure is described by the distribution of lexical elements linked by asymmetrical relations called dependencies [17]. One of the main characteristics is that, unlike constituent structures, a dependency tree has no phrasal nodes. Moreover, the dependency representations provide a direct encoding of predicate-argument structures, and the relations between units in a dependency tree are bilexical, i.e. they constitute binary (head, argument) relations [18]. Finally, in a dependency parse tree, most informative nodes (like the subject or the direct object of the sentence) are likely to be closer to the root node (main verb of the sentence). This means that (1) long-distance relations can be safely captured in a parse tree regardless of the number of modifiers that precede a target node (e.g. (subject, verb, object) relations), and (2) in definitions, tree-traversal algorithms can be easily implemented for skipping over-generalizing hypernyms (e.g. “class”, “kind” or “type”) as they are likely to appear near the main verb of the sentence, e.g. “X is a type of Y”.

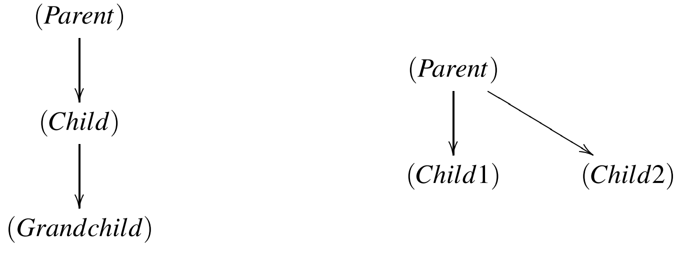
As mentioned before, and building up on previous work that exploits dependency parsing for Hypernym Extraction [10,15], we design a set of features that represent a sentence in terms of dependency relations among its lexical units.

3.1 Syntactic Motivation

We perform our experiments on the WCL dataset. This dataset is a subset of Wikipedia, where textual definitions and additional information are manually annotated. Such information, as described in [19], refers to: (1) The *definiendum*, i.e. concept that is being defined; (2) The *definitor*, i.e. the verb phrase to introduce the definition; (3) *definiens*, i.e. genus or phrase that contains the hypernym; and (4) *rest*, i.e. the rest of the sentence containing a definition. For simplicity, henceforth we refer to *definiens* as the union between *genus* and *rest*. A sample definition is illustrated below (see parse tree in Figure 1):

Sample Definition: “An <*term*> abbreviation</*term*> is a shortened form of a <*hyp*> word</*hyp*> or <*hyp*> phrase</*hyp*>.”

Firstly, we apply a dependency parser [20] to the WCL dataset and extract, for each sentence, all its subtrees with the following shapes:



Each node can either include surface form information, part of speech, the dependency relation of such node with its head, or a combination of any of the former[1]. We hypothesize that the encyclopedic genre is consistent enough as to be able to draw syntactic generalizations by firstly looking at its most recurrent patterns.

The representativeness of the two shapes described above in terms of encyclopedic language is very high. For example, the *is(Verb, Root)→in(Prep, Loc)→(Noun, PMOD)* amounts to almost 20% of the whole corpus[2]. In addition, over 98% of the definitions in

[1] For the remainder of the paper, we denote s as surface form, p as part of speech, and d as dependency relation.

[2] We denote syntactic dependencies as arrows (*head→governor*).

such dataset have one word with *PRD* syntactic function, and we found over 850 cases where the PRD token was a direct dependent of the Root verb, and was the first word of a manually tagged hypernym: this means that 46% of the (*term*,*hypernym*) relations in this dataset would be extracted applying a simple mapping rule. While this would introduce an undesirable amount of noise, it suggests that the common assumption that *textual definitions show a high syntactic variability* [6,11] depends on what we actually consider to be language variability, and the genre and domain to which the document or corpus belongs to. For this specific case (i.e. Wikipedia), there seems to be a fairly high syntactic consistence.

Having justified our data modelling choice, the next section describes the features we designed for informing our classifier.

3.2 Experimental Setup

What follows is a description of the features used to train our model. We can cluster them in three main groups, namely: Linguistic features (1-3); definitional features (4) and graph-based featuers (6-8). Our motivation for introducing graph-based features over the parse tree is the following: We hypothesize that a hypernym might be described in terms of the popularity of its word or phrase in the syntactic tree (computed in terms of adjacent edges), its children at several levels of depth, or its salience with regard to its frequency in informative subtrees like *SBJ←ROOT→PRD*. However, as our experiments reveal, while these features might be effectively used for Definition Extraction [21], only one out of four seems to contribute to the hypernym extraction task when a model already includes linguistic and definitional information.

1. **Surface form (*surface*) and lemma (*lemma*)**: Normalized (lower-case) surface form and lemma. Note that unlike the experiments shown in [22,23,15], we do not generalize the definiendum to a wildcard (TARGET or TERM). We argue that in a real-world scenario one does not necessarily know which is the definiendum term, and thus removing this information also contributes to a less biased classifier. Rather, we use this information as a feature in order to assess its contribution to the learning process.

2. **Part of Speech (*pos*)**: The part of speech of the current word

3. **Head Id (*headID*) and Dependency Relation (*depen*)**: These two features refer to the syntactic function of the current word and the unique identifier of its governor or head. For example, subject (SBJ), object (OBJ), predicative (PRD) or nominal modifier (NMOD).

4. **Definiendum (*term*) and definiens (*def-ndef*)**: Whether the word is a definiendum term (i.e. it matches exactly the Wikipiedia page title to which the text snippet belongs to), and whether such word is part of the definiens. We apply a simple heuristic rule that tags all words after the first verb of the sentence as definiens.

5. **PageRank (*p-rank*)**: We compute the popularity of a node in a sentence with the PageRank algorithm. To attain this, we use an off-the-shelf Python library: NetworkX [24].

6. **Node Outdegree (*outdegree*)**: The out-degree of a node in a syntactic dependency tree is equal to the number of dependents.

7. **Morphosyntactic chains (*chains*)**: We extract all children of a node recursively until we reach the tree leaves in breadth-first fashion. For each node, we extract part-of-speech and dependency relation. This feature is a string that represents such path. While this approach is inspired by previous work on Semantic Role Labelling [25], ours differs in that we also include the dependency information.

8. **Syntactic Salience (*syntS*)**: In addition to the above features, we are interested in a more general metric to assess the extent to which a word and its associated linguistic information describes a textual genre. Motivated by the fact that in textual definitions not only are hypernyms likely to appear, but they show syntactic regularities, we count how many times a word is part of the most frequent subtrees in the dataset taking into consideration different ranges of linguistic information (from only the word's surface form to subtrees including the word's surface form, part-of-speech and syntactic funtion).

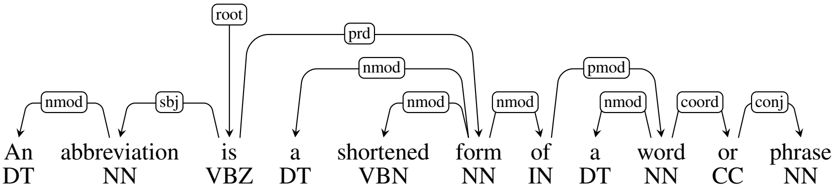


Fig. 1. Dependency parse tree of a textual definition

Numeric features such as node degree, pagerank or syntactic salience are discretized, i.e. within a range between the smallest and highest score, each value is assigned a discrete type between 1 and 10. This coarse-grained set of attributes allows us to understand better each feature's effect in the learning process and perform more sensible error analysis.

Having prepared our sets of features, these are used for training and testing a Conditional Random Fields (CRF) [26] classifier using CRF++[3]. Given the inherent ability of CRF for learning prior and posterior contextual information in a sequential classification task, we design three experiments where three context windows are considered: [-1,1], [-2,2] and [-3,3]. For each window, we design feature sets incrementally adding one feature at a time (see in Table 1 a matrix outlining all the feature sets used

[3] `https://code.google.com/p/crfpp/`

Table 1. Different feature sets adding one feature at a time

	surface	**lemma**	**pos**	**headID**	**depen**	**def-ndef**	**term**	**p-rank**	**outdegree**	**chains**	**syntS**
FeatSet1	x										
FeatSet2	x	x									
FeatSet3	x	x	x								
FeatSet4	x	x	x	x							
FeatSet5	x	x	x	x	x						
FeatSet6	x	x	x	x	x	x					
FeatSet7	x	x	x	x	x	x	x				
FeatSet8	x	x	x	x	x	x	x	x			
FeatSet9	x	x	x	x	x	x	x	x	x		
FeatSet10	x	x	x	x	x	x	x	x	x	x	
FeatSet11	x	x	x	x	x	x	x	x	x	x	x

in our experiments). The scores reported in this paper are derived from 10-fold cross validation.

3.3 Recall-Boosting Heuristics

After manually inspecting the output of the classifier, we observe that there are cases in which the discrepancy between the predicted label and the gold standard can be at questioned. In fact, [15] mention issues derived from the complexity of what actually constitutes a valid hypernym in a textual definition and its effect on the quality of the annotation of the WCL dataset. Among others, they refer to incorrect relationships, e.g. incorrectly annotating a meronym as a hypernym, or inconsistent modifier attachment, e.g. cases where the same modifier attached to two semantically-related concepts is sometimes included as part of a multiword hypernymic phrase, and others not.

This motivated a post-classification heuristic inspired by [27] consisting in a set of rules for label-switching. Let token$_i$ be a word classified as not being part of a hypernymic phrase (O), we perform the label-switching step replacing its current label with either B, i.e. at the beginning of a hypernym phrase, or I, i.e. inside a hypernym phrase, yielding token$_i^{update}$. The following conditions are considered:

$$\text{token}_i^{update} = \begin{cases} \text{B} & \text{if } P(\text{token}_i) = \text{B} > \theta \wedge P(\text{token}_i) = \text{B} > P(\text{token}_i) = \text{I} \\ \text{I} & \text{if } P(\text{token}_i) = \text{I} > \theta \wedge P(\text{token}_i) = \text{I} > P(\text{token}_i) = \text{B} \\ \text{B} & \text{if } P(\text{token}_i) = \text{O} < \lambda \wedge \text{token}_i^{Synt} = \text{PRD} \end{cases}$$

Where token$_i^{Synt}$ refers to the syntactic function of the word token$_i$, and where θ and λ are constants empirically set to .35 and .8 respectively after experimenting with several thresholds and inspecting manually the resulting classification.

These heuristics contribute to increase F-Score in feature sets 1 and 2 when considering [-1,1] contexts. Likewise, F-Score also improves after this step in feature sets 1, 2 and 3 when considering [-2,2] and [-3,3] contexts. In many configurations, recall improves almost 10 points, and while in strict comparison against gold standard the drop in precision affects negatively the overall F-Score in the majority of feature sets considered, we found that in some cases our greedier approach detected a better hypernym than the one manually annotated in the gold standard. Let us look at the following sample definition:

> “An abzyme (from antibody and enzyme), also called catmab (from catalytic monoclonal antibody), is a monoclonal antibody with catalytic activiy"

In the manually annotated dataset, the hypernym is “antibody”, and in the majority of our experiments our algorithm identifies “monoclonal antibody”, thus producing a false positive in our word-level evaluation. However, it is not clear that “antibody” is a better hypernym for “abzyme” than “monoclonal antibody”. In fact, there is a Wikipedia entry for “monoclonal antibody”[4], but not for “important antibody”, for instance, which suggests that the prediction of our algorithm is correct since “monoclonal” is not a property of “antibody” but rather defines a monosemic type of antibody.

4 Evaluation

4.1 Results and Discussion

We evaluated at token-level in terms of Precision, Recall and F-Measure by adding one feature at a time to the CRF-trained model. These results are shown in Table 2. Four main conclusions can be drawn: (1) Word-level morphosyntactic features are highly informative in the encyclopedic genre (see the boost in performance after these features are added to the model), which reinforces our intuition that syntactic structures do follow certain patterns and show regularities that can be exploited; (2) The best-performing model (highest F-Score) is *FeatSet8*, which includes all linguistic features, definitional information, and page-rank; (3) Unsurprisingly, the best performing models for each feature set are those including the largest context window ([-3,3]); and (4) Recall-Boosting post-classification rules increase F-Score only in the most basic feature sets. We provide further discussion on feature relevance in Section 4.2.

Finally, we compared our best-performing model with existing state-of-the-art systems reported in the literature. Firstly, the Word-Class Lattices algorithm [22], and secondly an approach conceptually similar to ours that also modelled the problem in terms of syntactic dependencies [15] (Table 3).

4.2 Information Gain

Information Gain measures the decrease in entropy when the feature is present vs. absent [28]. We rank our features according to their $score(f, ctx, i)$, where f_i is a token-level feature, *ctx* refers to the context window to which it is applied, and *i* is the index of the current token (i.e. its current iteration). We use the machine-learning toolkit Weka

[4] http://en.wikipedia.org/wiki/Monoclonal_antibody

Table 2. Performance of our CRF-trained model at three different context windows ([1:1], [2:2] and [3:3]). We include results before applying the post-classification-heuristic (DefConf) and after (Boosted). We observe the best performance when only linguistic and definitional information is considered.

		DefConf-1:1	**DefConf-2:2**	**DefConf-3:3**	**Boosted-1:1**	**Boosted-2:2**	**Boosted-3:3**
*FeatSet*1	P	48.51	65.22	70.33	30.35	40.22	46.46
	R	31.96	41.45	48.34	65.44	72.06	75.23
	F	38.49	50.64	57.25	41.43	51.6	57.41
*FeatSet*2	P	49.36	61.87	66.55	32.12	41.77	47.84
	R	33.92	44.33	51.13	64.52	71.26	74.27
	F	40.17	51.58	57.79	42.85	52.66	58.18
*FeatSet*3	P	64.93	67.58	72.65	41.98	49.38	55.32
	R	33.17	47.23	56.62	64.68	71.34	75.36
	F	43.85	55.54	63.31	50.86	58.34	63.79
*FeatSet*4	P	70.32	72.41	74.32	48.05	53.2	58.47
	R	44.98	55.37	60.87	70.07	74.63	76.37
	F	54.8	62.71	66.89	56.99	62.1	66.22
*FeatSet*5	P	76.04	75.85	76.17	56.03	58.67	62.05
	R	54.33	61.52	64.73	74.68	76.86	78.49
	F	63.34	67.88	69.94	64.01	66.51	69.31
*FeatSet*6	P	80.19	82.99	**84.22**	62.44	68.14	73.08
	R	63.26	72.04	75.69	79.85	82.42	**84.99**
	F	70.68	77.12	79.71	70.04	74.59	78.58
*FeatSet*7	P	80.08	83.05	84.15	62	68.43	73.25
	R	63.15	72.04	75.51	79.57	82.47	84.96
	F	70.57	77.13	79.58	69.66	74.77	78.67
*FeatSet*8	P	80.11	82.56	84.01	62.67	68.34	72.59
	R	63.47	72.02	76.12	79.68	82.27	84.82
	F	70.79	76.91	**79.85**	70.13	74.64	78.22
*FeatSet*9	P	79.94	82.31	83.82	62.01	68.04	72.44
	R	63.68	72.06	75.94	79.58	82.26	84.64
	F	70.86	76.82	79.66	69.67	74.46	78.06
*FeatSet*10	P	79.6	81.86	83.6	62.4	68.64	72.71
	R	63.86	71.35	75.74	79.02	81.69	84.51
	F	70.85	76.23	79.47	69.7	74.59	78.15
*FeatSet*11	P	79.72	81.87	83.43	62.69	68.7	73.1
	R	64.48	71.62	75.36	79.22	82.13	84.16
	F	71.28	76.03	79.17	69.94	74.81	78.22

Table 3. Comparative Evaluation between our best performing model (FeatureSet8 with no post-classification heuristics) and the results reported in [22] and [15]

	Precision	Recall	F-Score
N&V WCL-1	77	42.09	54.42
N&V WCL-3	78.58	60.74	68.56
B&DiC	83.05	68.64	75.16
Our Approach	**84.01**	**76.12**	**79.85**

Table 4. Selected best features for Hypernym Extraction. Each feature reads as follows: $featureName$Position=value, where Position refers to the context in which appears at the current iteration. For instance, Position=-1 refers to one word before the word at the current iteration.

Rank	**Feature**	**InfoGain**
1	deprelPosition0=PRD	0.0682345
2	posPosition0=nn	0.0538957
3	deprelPosition-1=NMOD	0.0517277
4	defnodefPositiond0=def	0.0349189
5	defnodefPosition0=nodef	0.0349189
6	defnodefPosition1=def	0.0349189
7	headIDPosition-1	0.0320474
8	deprelPosition-2=ROOT	0.0315236
9	defnodefPosition+1=nodef	0.0300525
10	defnodefPosition-3=nodef	0.0300255
24	chainsPosition0=dt_NMOD&nnp_SBJ	0.0182301

[29]. Looking at the best features in our model (Table 4), we can conclude the following[5]: (1) Hypernym extraction algorithms improve by a huge margin if provided with syntactic information; (2) Previous work has demonstrated improvement in the task of Definition Extraction by informing the classifier with terminological information [23]. This seems to hold the other way round as well; (3) We also observe an interesting set of features clumped together with the same value and the same Information Gain score. These are *no_value* feature scores, which means that the context specified (e.g. $i = -1$)

[5] The full set of features and their Information Gain rank can be accessed at: `https://www.dropbox.com/s/d8er9jvgjz2dqo8/infogain_syntsal.txt?dl=0`. There are 2111 features with non-zero Information Gain score.

is null due to the current iteration being at the beginning or end of the sentence. This might point to hypernyms being consistently mentioned at a certain position in a sentence; (4) the discretization of our numeric values might have been too coarse-grained for being discriminative enough in a classification task. Finally, (5) After looking at the last row in Table 4, we observe the highest graph-based ranking feature (in position 24) referring to the fact that a word has a child with NNP part-of-speech and dependency relation SBJ.

5 Conclusions and Future Work

We have described a set of experiments on hypernym extraction from textual definitions in the WCL dataset. We experimented with linguistic, definitional and graph-based features which operated over the sentence parse tree. Our best model achieves competitive results in comparison with existing approaches on the same dataset. The experiments carried out also showed that linguistic and definitional information are by far the most important features in our configuration, and only few exceptions among the graph-based features can be considered informative.

Our main conclusions can be summarized as follows: (1) Hypernym extraction from textual definitions benefits significantly from syntactic and definitional information; (2) Recall-boosting heuristics contribute to increase the overall F-Score in configurations that considered smaller context windows; and (3) Graph-based features have limited discriminative power for this task.

The approach presented in this paper to hypernym extraction in textual definitions opens several avenues for future work. For example, we would like to draw statistics to measure accurately how many of the false positives in which our approach incurred after applying the Recall-Boosting heuristics could be correct hypernyms by looking at generic encyclopedias or domain-specific knowledge bases. Also, since the contribution of graph-based features was very limited, we would like to explore with finer-grained discretization heuristics as well as with the raw numeric values. Finally, it would be interesting to test our approach on other large datasets, such as WiBi [10] or the Linked Hypernyms Dataset [30].

Acknowledgments. We would like to express our gratitude to the anonymous reviewers for their helpful comments. This work is partially funded by the SKATER project, TIN2012-38584-C06-03, Ministerio de Economía y Competitividad, Secretaría de Estado de Investigación, Desarrollo e Innovación, España; and Dr. Inventor (FP7-ICT-2013.8.1 611383), programa Ramón y Cajal 2009 (RYC-2009-04291).

References

1. Fu, R., Guo, J., Qin, B., Che, W., Wang, H., Liu, T.: Learning semantic hierarchies via word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol 1: Long Papers. Association for Computational Linguistics, pp. 1199–1209 (2014)

2. Kazama, J., Torisawa, K.: Exploiting wikipedia as external knowledge for named entity recognition. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 698–707 (2007)
3. Chandramouli, K., Kliegr, T., Nemrava, J., Svátek, V., Izquierdo, E.: Query refinement and user relevance feedback for contextualized image retrieval. In: Proceedings of the 5th International Conference on Visual Information Engineering (2008)
4. Kliegr, T., Chandramouli, K., Nemrava, J., Svatek, V., Izquierdo, E.: Combining image captions and visual analysis for image concept classification. In: Proceedings of the 9th International Workshop on Multimedia Data Mining: Held in Conjunction with the ACM SIGKDD, pp. 8–17. ACM (2008)
5. Navigli, R., Velardi, P., Faralli, S.: A graph-based algorithm for inducing lexical taxonomies from scratch. In: IJCAI 2011, pp. 1872–1877 (2011)
6. Saggion, H., Gaizauskas, R.: Mining on-line sources for definition knowledge. In: 17th FLAIRS, Miami Bearch, Florida, pp. 45–52 (2004)
7. Muresan, A., Klavans, J.: A method for automatically building and evaluating dictionary resources. In: Proceedings of the Language Resources and Evaluation Conference, LREC. European Language Resources Association (2002)
8. Roller, S., Erk, K., Boleda, G.: Inclusive yet selective: Supervised distributional hypernymy detection. In: Proceedings of the Twenty Fifth International Conference on Computational Linguistics, COLING 2014, Dublin, Ireland, pp. 1025–1036 (2014)
9. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM 38, 39–41 (1995)
10. Flati, T., Vannella, D., Pasini, T., Navigli, R.: Two is bigger (and better) than one: the wikipedia bitaxonomy project. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol 1: Long Papers. Association for Computational Linguistics, pp. 945–955 (2014)
11. Navigli, R., Velardi, P., Ruiz-Martínez, 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. Language Resources Association (ELRA), Valletta (2010)
12. Hearst, M.A.: 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)
13. Snow, R., Jurafsky, D., Ng, A.Y.: Learning syntactic patterns for automatic hypernym discovery. Advances in Neural Information Processing Systems 17 (2004)
14. Herbelot, A., Copestake, A.: Acquiring ontological relationships from wikipedia using rmrs. In: Proceedings of Workshop on Web Content Mining with Human Language Technologies, ISWC 2006. Citeseer (2006)
15. Boella, G., Di Caro, L., Ruggeri, A., Robaldo, L.: Learning from syntax generalizations for automatic semantic annotation. Journal of Intelligent Information Systems, 1–16 (2014)
16. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: HLT-NAACL, pp. 746–751. Citeseer (2013)
17. Nivre, J.: Dependency grammar and dependency parsing. Technical reporut, Växjö University (2005)
18. Ivanova, A., Oepen, S., Dridan, R., Flickinger, D., Øvrelid, L.: On different approaches to syntactic analysis into bi-lexical dependencies an empirical comparison of direct, pcfg-based, and hpsg-based parsers. In: Proceedings of the 13th International Conference on Parsing Technologies, pp. 63–72 (2013)
19. Storrer, A., Wellinghoff, S.: Automated detection and annotation of term definitions in German text corpora. In: Conference on Language Resources and Evaluation, LREC (2006)

20. Bohnet, B.: Very high accuracy and fast dependency parsing is not a contradiction. In: Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010, pp. 89–97. Association for Computational Linguistics, Stroudsburg (2010)
21. Espinosa-Anke, L., Saggion, H.: Applying dependency relations to definition extraction. In: Métais, E., Roche, M., Teisseire, M. (eds.) Natural Language Processing and Information Systems. LNCS, vol. 8455, pp. 63–74. Springer, Heidelberg (2014)
22. 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, ACL 2010, pp. 1318–1327. Association for Computational Linguistics, Stroudsburg (2010)
23. Jin, Y., Kan, M.Y., Ng, J.P., He, X.: Mining scientific terms and their definitions: A study of the ACL anthology. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 780–790. Association for Computational Linguistics, Seattle (2013)
24. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using NetworkX. In: Proceedings of the 7th Python in Science Conference (SciPy 2008), Pasadena, CA, USA, pp. 11–15 (2008)
25. Hacioglu, K.: Semantic role labeling using dependency trees. In: International Conference on Computional Linguistics (COLING). Association for Computational Linguistics, Stroudsburg (2004)
26. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML 2001, pp. 282–289. Morgan Kaufmann Publishers Inc., San Francisco (2001)
27. Cai, P., Luo, H., Zhou, A.: Named entity recognition in italian using crf. In: Poster and Workshop Proceedings of the 11th Conference of the Italian Association for Artificial Intelligence, Reggio Emilia, Italy (2009)
28. Forman, G.: An extensive empirical study of feature selection metrics for text classification. The Journal of Machine Learning Research 3, 1289–1305 (2003)
29. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann Publishers Inc., San Francisco (2005)
30. Kliegr, T.: Linked hypernyms: Enriching dbpedia with targeted hypernym discovery. Web Semantics: Science, Services and Agents on the World Wide Web (2014)