Abstract
Definition extraction is the task of automatically identifying definitional sentences within texts. The task has proven useful in many research areas including ontology learning, relation extraction and question answering. However, current approaches – mostly focused on lexico-syntactic patterns – suffer from both low recall and precision, as definitional sentences occur in highly variable syntactic structures. In this paper, we propose Word-Class Lattices (WCLs), a generalization of word lattices that we use to model textual definitions. Lattices are learned from a dataset of definitions from Wikipedia. Our method is applied to the task of definition and hypernym extraction and compares favorably to other pattern generalization methods proposed in the literature.

1 Introduction
Textual definitions constitute a fundamental source to look up when the meaning of a term is sought. Definitions are usually collected in dictionaries and domain glossaries for consultation purposes. However, manually constructing and updating glossaries requires the cooperative effort of a team of domain experts. Further, in the presence of new words or usages, and – even worse – new domains, such resources are of no help. Nonetheless, terms are attested in texts and some (usually few) of the sentences in which a term occurs are typically definitional, that is they provide a formal explanation for the term of interest. While it is not feasible to manually search texts for definitions, this task can be automatized by means of Machine Learning (ML) and Natural Language Processing (NLP) techniques.

Automatic definition extraction is useful not only in the construction of glossaries, but also in many other NLP tasks. In ontology learning, definitions are used to create and enrich concepts with textual information (Gangemi et al., 2003), and extract taxonomic and non-taxonomic relations (Snow et al., 2004; Navigli and Velardi, 2006; Navigli, 2009a). Definitions are also harvested in Question Answering to deal with “what is” questions (Cui et al., 2007; Saggion, 2004). In eLearning, they are used to help students assimilate knowledge (Westerhout and Monachesi, 2007), etc.

Much of the current literature focuses on the use of lexico-syntactic patterns, inspired by Hearst’s (1992) seminal work. However, these methods suffer both from low recall and precision, as definitional sentences occur in highly variable syntactic structures, and because the most frequent definitional pattern – X is a Y – is inherently very noisy.

In this paper we propose a generalized form of word lattices, called Word-Class Lattices (WCLs), as an alternative to lexico-syntactic pattern learning. A lattice is a directed acyclic graph (DAG), a subclass of non-deterministic finite state automata (NFA). The lattice structure has the purpose of preserving the salient differences among distinct sequences, while eliminating redundant information. In computational linguistics, lattices have been used to model in a compact way many sequences of symbols, each representing an alternative hypothesis. Lattice-based methods differ in the types of nodes (words, phonemes, concepts), the interpretation of links (representing either a sequential or hierarchical ordering between nodes), their means of creation, and the scoring method used to extract the best consensus output from the lattice (Schroeder et al., 2009). In speech processing, phoneme or word lattices (Campbell et al., 2007; Mathias and Byrne, 2006; Collins et al., 2004) are used as an interface between speech recognition and understanding. Lat-
tices are adopted also in Chinese word segmentation (Jiang et al., 2008), decompounding in German (Dyer, 2009), and to represent classes of translation models in machine translation (Dyer et al., 2008; Schroeder et al., 2009). In more complex text processing tasks, such as information retrieval, information extraction and summarization, the use of word lattices has been postulated but is considered unrealistic because of the dimension of the hypothesis space.

To reduce this problem, concept lattices have been proposed (Carpineto and Romano, 2005; Klein, 2008; Zhong et al., 2008). Here links represent hierarchical relations, rather than the sequential order of symbols like in word/phoneme lattices, and nodes are clusters of salient words aggregated using synonymy, similarity, or subtrees of a thesaurus. However, salient word selection and aggregation is non-obvious and furthermore it falls into word sense disambiguation, a notoriously AI-hard problem (Navigli, 2009b).

In definition extraction, the variability of patterns is higher than for “traditional” applications of lattices, such as translation and speech, however not as high as in unconstrained sentences. The methodology that we propose to align patterns is based on the use of star (wildcard *) characters to facilitate sentence clustering. Each cluster of sentences is then generalized to a lattice of word classes (each class being either a frequent word or a part of speech). A key feature of our approach is its inherent ability to both identify definitions and extract hypernyms. The method is tested on an annotated corpus of Wikipedia sentences and a large Web corpus, in order to demonstrate the independence of the method from the annotated dataset. WCLs are shown to generalize over lexico-syntactic patterns, and outperform well-known approaches to definition and hypernym extraction.

The paper is organized as follows: Section 2 discusses related work, WCLs are introduced in Section 3 and illustrated by means of an example in Section 4, experiments are presented in Section 5. We conclude the paper in Section 6.

2 Related Work

Definition Extraction. A great deal of work is concerned with definition extraction in several languages (Klavans and Muresan, 2001; Storrer and Wellinghoff, 2006; Gaudio and Branco, 2007; Iftene et al., 2007; Westerhout and Monachesi, 2007; Przepiórkowski et al., 2007; Degorski et al., 2008). The majority of these approaches use symbolic methods that depend on lexico-syntactic patterns or features, which are manually crafted or semi-automatically learned (Zhang and Jiang, 2009; Hovy et al., 2003; Fahmi and Bouma, 2006; Westerhout, 2009). Patterns are either very simple sequences of words (e.g. “refers to”, “is defined as”, “is a”) or more complex sequences of words, parts of speech and chunks. A fully automated method is instead proposed by Borg et al. (2009): they use genetic programming to learn simple features to distinguish between definitions and non-definitions, and then they apply a genetic algorithm to learn individual weights of features. However, rules are learned for only one category of patterns, namely “is” patterns. As we already remarked, most methods suffer from both low recall and precision, because definitional sentences occur in highly variable and potentially noisy syntactic structures. Higher performance (around 60-70% F₁-measure) is obtained only for specific domains (e.g., an ICT corpus) and patterns (Borg et al., 2009).

Only few papers try to cope with the generality of patterns and domains in real-world corpora (like the Web). In the GlossExtractor web-based system (Velardi et al., 2008), to improve precision while keeping pattern generality, candidates are pruned using more refined stylistic patterns and lexical filters. Cui et al. (2007) propose the use of probabilistic lexico-semantic patterns, called soft patterns, for definitional question answering in the TREC contest. The authors describe two soft matching models: one is based on an n-gram language model (with the Expectation Maximization algorithm used to estimate the model parameter), the other on Profile Hidden Markov Models (PHMM). Soft patterns generalize over lexico-syntactic “hard” patterns in that they allow a partial matching by calculating a generative degree of match probability between the test instance and the set of training instances. Thanks to its generalization power, this method is the most closely related to our work, however the task of definitional question answering to which it is applied is slightly different from that of definition extraction, so a direct performance comparison is not possi-
able. In fact, the TREC evaluation datasets cannot be considered true definitions, but rather text fragments providing some relevant fact about a target term. For example, sentences like: “Bollywood is a Bombay-based film industry” and “700 or more films produced by India with 200 or more from Bollywood” are both “vital” answers for the question “Bollywood”, according to TREC classification, but the second sentence is not a definition.

Hypernym Extraction. The literature on hypernym extraction offers a higher variability of methods, from simple lexical patterns (Hearst, 1992; Oakes, 2005) to statistical and machine learning techniques (Agirre et al., 2000; Caraballo, 1999; Dolan et al., 1993; Sanfilippo and Poznański, 1992; Ritter et al., 2009). One of the highest-coverage methods is proposed by Snow et al. (2004). They first search sentences that contain two terms which are known to be in a taxonomic relation (term pairs are taken from WordNet (Miller et al., 1990)); then they parse the sentences, and automatically extract patterns from the parse trees. Finally, they train a hypernym classifier based on these features. Lexico-syntactic patterns are generated for each sentence relating a term to its hypernym, and a dependency parser is used to represent them.

3 Word-Class Lattices

3.1 Preliminaries

Notion of definition. In our work, we rely on a formal notion of textual definition. Specifically, given a definition, e.g.: “In computer science, a closure is a first-class function with free variables that are bound in the lexical environment”, we assume that it contains the following fields (Storrer and Wellinghoff, 2006):

- The Definiens field (GF): it includes the
  - The Definiendum field (DF): this part of
  - The Definitor field (VF): it includes the
  - The Genus phrase (usually including the hypernym, e.g., “a first-class function”);
  - The Rest field (RF): it includes additional

Further examples of definitional sentences annotated with the above fields are shown in Table 1. For each sentence, the definiendum (that is, the word being defined) and its hypernym are marked in bold and italic, respectively. Given the lexico-syntactic nature of the definition extraction models we experiment with, training and test sentences are part-of-speech tagged with the TreeTagger system, a part-of-speech tagger available for many languages (Schmid, 1995).

Word Classes and Generalized Sentences. We now introduce our notion of word class, on which our learning model is based. Let $T$ be the set of training sentences, manually bracketed with the DF, VF, GF and RF fields. We first determine the set $F$ of words in $T$ whose frequency is above a threshold $\theta$ (e.g., the, a, is, of, refer, etc.). In our training sentences, we replace the term being defined with (TARGET), thus this frequent token is also included in $F$.

We use the set of frequent words $F$ to generalize words to “word classes”. We define a word class as either a word itself or its part of speech. Given a sentence $s = w_1, w_2, \ldots, w_s$, where $w_i$ is the $i$-th word of $s$, we generalize its words $w_i$ to word classes $\omega_i$ as follows:

$$\omega_i = \begin{cases} w_i & \text{if } w_i \in F \\ \text{POS}(w_i) & \text{otherwise} \end{cases}$$

that is, a word $w_i$ is left unchanged if it occurs frequently in the training corpus (i.e., $w_i \in F$) or is transformed to its part of speech (POS($w_i$)) otherwise. As a result, we obtain a generalized sentence $s' = \omega_1, \omega_2, \ldots, \omega_s$. For instance, given the first sentence in Table 1, we obtain the corresponding generalized sentence: “In NN, a (TARGET) is a JJ NN”, where NN and JJ indicate the noun and adjective classes, respectively.

3.2 Algorithm

We now describe our learning algorithm based on Word-Class Lattices. The algorithm consists of three steps:

2 In the paper, a 55% recall and 34% precision is achieved with the best experiment on TREC-13 data. Furthermore, the classifier of Cui et al. (2007) is based on soft patterns but also on a bag-of-word relevance heuristic. However, the relative influence of the two methods on the final performance is not discussed.
[In arts, a **chiaroscuro**]_{DF} [is]_{VF} [a monochrome **picture**]_{GF}. [In mathematics, a **graph**]_{DF} [is]_{VF} [a data structure]_{GF} [that consists of ...]_{REST}. [In computer science, a **pixel**]_{DF} [is]_{VF} [a dot]_{GF} [that is part of a computer image]_{REST}.

Table 1: Example definitions (defined terms are marked in bold face, their hypernyms in italic).

- **Star patterns**: each sentence in the training set is pre-processed and generalized to a star pattern. For instance, “In arts, a chiaroscuro is a monochrome picture” is transformed to “In *, a (TARGET) is a *” (Section 3.2.1);
- **Sentence clustering**: the training sentences are then clustered based on the star patterns to which they belong (Section 3.2.2);
- **Word-Class Lattice construction**: for each sentence cluster, a WCL is created by means of a greedy alignment algorithm (Section 3.2.3).

We present two variants of our WCL model, dealing either globally with the entire sentence or separately with its definition fields (Section 3.2.4). The WCL models can then be used to classify any input sentence of interest (Section 3.2.5).

### 3.2.1 Star Patterns

Let $T$ be the set of training sentences. In this step, we associate a star pattern $\sigma(s)$ with each sentence $s \in T$. To do so, let $s \in T$ be a sentence such that $s = w_1, w_2, \ldots, w_{|s|}$, where $w_i$ is its $i$-th word. Given the set $F$ of most frequent words in $T$ (cf. Section 3.1), the star pattern $\sigma(s)$ associated with $s$ is obtained by replacing with * all the words $w_i \notin F$, that is all the tokens that are non-frequent words. For instance, given the sentence “In arts, a chiaroscuro is a monochrome picture”, the corresponding star pattern is “In *, a (TARGET) is a *”, where (TARGET) is the defined term.

Note that, here and in what follows, we discard the sentence fragments tagged with the REST field, which is used only to delimit the core part of definitional sentences.

### 3.2.2 Sentence Clustering

In the second step, we cluster the sentences in our training set $T$ based on their star patterns. Formally, let $\Sigma = (\sigma_1, \ldots, \sigma_m)$ be the set of star patterns associated with the sentences in $T$. We create a clustering $C = (C_1, \ldots, C_m)$ such that $C_i = \{s \in T : \sigma(s) = \sigma_i\}$, that is $C_i$ contains all the sentences whose star pattern is $\sigma_i$.

As an example, assume $\sigma_3$ = “In *, a (TARGET) is a *”. The sentences reported in Table 1 are all grouped into cluster $C_3$. We note that each cluster $C_i$ contains sentences whose degree of variability is generally much lower than for any pair of sentences in $T$ belonging to two different clusters.

### 3.2.3 Word-Class Lattice Construction

Finally, the third step consists of the construction of a Word-Class Lattice for each sentence cluster. Given such a cluster $C_i \in C$, we apply a greedy algorithm that iteratively constructs the WCL.

Let $C_i = \{s_1, s_2, \ldots, s_{|C_i|}\}$ and consider its first sentence $s_1 = w_1^1, w_2^1, \ldots, w_{|s_1|}^1$ ($w_i^1$ denotes the $i$-th token of the $j$-th sentence).

We first produce the corresponding generalized sentence $s_1^1 = \omega_1^1, \omega_2^1, \ldots, \omega_{|s_1|}^1$. (cf. Section 3.1). We then create a directed graph $G = (V, E)$ such that $V = \{\omega_1^1, \ldots, \omega_{|s_1|}^1\}$ and $E = \{(\omega_1^1, \omega_2^1), (\omega_2^1, \omega_3^1), \ldots, (\omega_{|s_1|-1}^1, \omega_{|s_1|}^1)\}$.

Next, for the subsequent sentences in $C_i$, that is, for each $j = 2, \ldots, |C_i|$, we determine the alignment between the sentence $s_j$ and each sentence $s_k \in C_i$ such that $k < j$ based on the following dynamic programming formulation (Cormen et al., 1990, pp. 314-319):

$$M_{a,b} = \max \{M_{a-1,b-1} + S_{a,b}, M_{a,b-1}, M_{a-1,b}\}$$

where $a \in \{1, \ldots, |s_k|\}$ and $b \in \{1, \ldots, |s_j|\}$. $S_{a,b}$ is a score of the matching between the $a$-th token of $s_k$ and the $b$-th token of $s_j$, and $M_{0,0}$, $M_{0,b}$ and $M_{a,0}$ are initially set to 0 for all $a$ and $b$.

The matching score $S_{a,b}$ is calculated on the generalized sentences $s_k'$ of $s_k$ and $s_j'$ of $s_j$ as follows:

$$S_{a,b} = \begin{cases} 1 & \text{if } \omega_a^k = \omega_b^j \\ 0 & \text{otherwise} \end{cases}$$

where $\omega_a^k$ and $\omega_b^j$ are the $a$-th and $b$-th word classes of $s_k'$ and $s_j'$, respectively. In other words, the matching score equals 1 if the $a$-th and the $b$-th tokens of the two original sentences have the same word class.

Finally, the alignment score between $s_k$ and $s_j$ is given by $M_{|s_k|,|s_j|}$, which calculates the mini-
In arts science mathematics, a pixel graph chiaroscuro is a monochrome structure picture dot data. We also add to the sentence sequences. We repeat this calculation for each mal number of misalignments between the two tobeside the corresponding node.

Figure 1: The Word-Class Lattice for the sentences in Table 1. The support of each word class is reported beside the corresponding node.

3.2.4 Variants of the WCL Model
So far, we have assumed that our WCL model learns lattices from the training sentences in their entirety (we call this model WCL-1). We now propose a second model that learns separate WCLs for each field of the definition, namely: the DEFINIENDUM (DF), DEFINITOR (VF) and DEFINIENS (GF) fields (see Section 3.1). We refer to this latter model as WCL-3. Rather than applying the WCL algorithm to the entire sentence, the very same method is applied to the sentence fragments tagged with one of the three definition fields. The reason for introducing the WCL-3 model is that, while definitional patterns are highly variable, DF, VF and GF individually exhibit a lower variability, thus WCL-3 should improve the generalization power.

3.2.5 Classification
Once the learning process is over, a set of WCLs is produced. Given a test sentence s, the classification phase for the WCL-1 model consists of determining whether it exists a lattice that matches s. In the case of WCL-3, we consider any combination of DEFINIENDUM, DEFINITOR and DEFINIENS lattices. While WCL-1 is applied as a yes-no classifier as there is a single WCL that can possibly match the input sentence, WCL-3 selects, if any, the combination of the three WCLs that best fits the sentence. In fact, choosing the most appropriate combination of lattices impacts the performance of hypernym extraction. The best combination of WCLs is selected by maximizing the following confidence score:

\[ \text{score}(s, l_{DF}, l_{VF}, l_{GF}) = \text{coverage} \cdot \log(\text{support}) \]

where s is the candidate sentence, \( l_{DF}, l_{VF} \) and \( l_{GF} \) are three lattices one for each definition field, coverage is the fraction of words of the input sentence covered by the three lattices, and support is the sum of the number of sentences in the star patterns corresponding to the three lattices.

Finally, when a sentence is classified as a definition, its hypernym is extracted by selecting the words in the input sentence that are marked as “hypernyms” in the WCL-1 lattice (or in the WCL-3 GF lattice).

4 Example
As an example, consider the definitions in Table 1. As illustrated in Section 3.2.2, their star pattern is “In *, a (TARGET) is a *”. The corresponding WCL is built as follows: the first part-of-speech tagged sentence, “In/IN arts/NN, a/DT (TARGET)/NN is/VBZ a/DT monochrome/JJ picture/NN”, is considered. The corresponding generalized sentence is “In NN, a (TARGET) is a JJ NN”. The initially empty graph is thus populated with one node for each word class and one edge for each pair of consecutive tokens, as shown in Figure 1 (the central sequence of nodes in the graph). Note that we draw the hypernym token \( \text{NN}_2 \) with a rectangle shape. We also add to the
graph a start node - and an end node - , and connect them to the corresponding initial and final sentence tokens. Next, the second sentence, “In mathematics, a graph is a data structure that consists of...”, is aligned to the first sentence. The alignment of the generalized sentence is perfect, apart from the node corresponding to “data”. The node is added to the graph together with the edges - and - . Finally, the third sentence in Table 1, “In computer science, a pixel is a dot that is part of a computer image”, is generalized as “In , a is a NN”. Thus, a new node NN is added, corresponding to “computer” and new edges are added: - and - . Figure 1 shows the resulting WCL-1 lattice.

5 Experiments

5.1 Experimental Setup

Datasets. We conducted experiments on two different datasets:

- A corpus of 4,619 Wikipedia sentences, that contains 1,908 definitional and 2,711 non-definitional sentences. The former were obtained from a random selection of the first sentences of Wikipedia articles. The defined terms belong to different Wikipedia domain categories, so as to capture a representative and cross-domain sample of lexical and syntactic patterns for definitions. These sentences were manually annotated with , and fields by an expert annotator, who also marked the hypernyms. The associated set of negative examples (“syntactically plausible” false definitions) was obtained by extracting from the same Wikipedia articles sentences in which the page title occurs.

- A subset of the ukWaC Web corpus (Ferraresi et al., 2008), a large corpus of the English language constructed by crawling the .uk domain of the Web. The subset includes over 300,000 sentences in which occur any of 239 terms selected from the terminology of four different domains (computer science, astronomy, cardiology, aviation).

The reason for using the ukWaC corpus is that, unlike the “clean” Wikipedia dataset, in which relatively simple patterns can achieve good results, ukWaC represents a real-world test, with many complex cases. For example, there are sentences that should be classified as definitional according to Section 3.1 but are rather uninformative, like “dynamic programming was the brainchild of an american mathematician”, as well as informative sentences that are not definitional (e.g., they do not have a hypernym), like “cubism was characterised by muted colours and fragmented images”. Even more frequently, the dataset includes sentences which are not definitions but have a definitional pattern (“A Pacific Northwest tribe’s saga refers to a young woman who [...]”), or sentences with very complex definitional patterns (“white body cells are the body’s clean up squad” and “joule is also an expression of electric energy”). These cases can be correctly handled only with fine-grained patterns. Additional details on the corpus and a more thorough linguistic analysis of complex cases can be found in Navigli et al. (2010).

Systems. For definition extraction, we experiment with the following systems:

- WCL-1 and WCL-3: these two classifiers are based on our Word-Class Lattice model. WCL-1 learns from the training set a lattice for each cluster of sentences, whereas WCL-3 identifies clusters (and lattices) separately for each sentence field ( , and ) and classifies a sentence as a definition if any combination from the three sets of lattices matches (cf. Section 3.2.4, the best combination is selected).

- Star patterns: a simple classifier based on the patterns learned as a result of step 1 of our WCL learning algorithm (cf. Section 3.2.1): a sentence is classified as a definition if it matches any of the star patterns in the model.

- Bigrams: an implementation of the bigram classifier for soft pattern matching proposed by Cui et al. (2007). The classifier selects as definitions all the sentences whose probability is above a specific threshold. The probability is calculated as a mixture of bigram and
unigram probabilities, with Laplace smoothing on the latter. We use the very same settings of Cui et al. (2007), including threshold values. While the authors propose a second soft-pattern approach based on Profile HMM (cf. Section 2), their results do not show significant improvements over the bigram language model.

For hypernym extraction, we compared WCL-1 and WCL-3 with Hearst's patterns, a system that extracts hypernyms from sentences based on the lexico-syntactic patterns specified in Hearst’s seminal work (1992). These include (hypernym in italic): “such *NP* as *NP*, {or and} *NP*, "*NP* {or and} *NP*, "*NP* {or and} *NP*, especially {or and} *NP*, and variants thereof. However, it should be noted that hypernym extraction methods in the literature do not extract hypernyms from definitional sentences, like we do, but rather from specific patterns like “*X* such as *Y*”. Therefore a direct comparison with these methods is not possible. Nonetheless, we decided to implement Hearst’s patterns for the sake of completeness. We could not replicate the more refined approach by Snow et al. (2004) because it requires the annotation of a possibly very large dataset of sentence fragments. In any case Snow et al. (2004) reported the following performance figures on a corpus of dimension and complexity comparable with ukWaC: the recall-precision graph indicates precision 85% at recall 10% and precision 25% at recall 30% for the hypernym classifier. A variant of the classifier that includes evidence from coordinate terms (terms with a common ancestor in a taxonomy) obtains an increased precision of 35% at recall 30%. We see no reasons why these figures should vary dramatically on the ukWaC.

Finally, we compare all systems with the random baseline, that classifies a sentence as a definition with probability 1/2.

### 5.2 Results and Discussion

#### Definition Extraction.

In Table 2 we report the results of definition extraction systems on the Wikipedia dataset. Given this dataset is also used for training, experiments are performed with 10-fold cross validation. The results show very high precision for WCL-1, WCL-3 (around 99%) and star patterns (86%). As expected, bigrams and star patterns exhibit a higher recall (82% and 66%, respectively). The lower recall of WCL-1 is due to its limited ability to generalize compared to WCL-3 and the other methods. In terms of $F_1$-measure, star patterns and WCL-3 achieve 75%, and are thus the best systems. Similar performance is observed when we also account for negative sentences – that is we calculate accuracy (with WCL-3 performing better). All the systems perform significantly better than the random baseline.

From our Wikipedia corpus, we learned over 1,000 lattices (and star patterns). Using WCL-3, we learned 381 DF, 252 VF and 395 GF lattices, that then we used to extract definitions from WCL-3.
Table 4: Precision in hypernym extraction on the Wikipedia dataset

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Full</th>
<th>Substring</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCL-1</td>
<td>42.75</td>
<td>77.00</td>
</tr>
<tr>
<td>WCL-3</td>
<td>40.73</td>
<td>78.58</td>
</tr>
</tbody>
</table>

Table 5: Precision in hypernym extraction on the ukWaC dataset (number of hypernyms in parentheses).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Full</th>
<th>Substring</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCL-1</td>
<td>86.19 (206)</td>
<td>96.23 (230)</td>
</tr>
<tr>
<td>WCL-3</td>
<td>89.27 (383)</td>
<td>96.27 (413)</td>
</tr>
<tr>
<td>Hearst</td>
<td>65.26 (62)</td>
<td>88.42 (84)</td>
</tr>
</tbody>
</table>

For the above reasons it is difficult to achieve high performance in capturing the correct hypernym (e.g. 40.73% with WCL-3 on Wikipedia). However, our performance of identifying a substring of the correct hypernym is much higher (around 78.58%). In Table 4 we do not report the precision of Hearst’s patterns, as only one hypernym was found, due to the inherently low coverage of the method.

On the ukWaC dataset, the hypernyms returned by the three systems were manually validated and precision was calculated. Both WCL-1 and WCL-3 obtained a very high precision (86-89% and 96% in identifying the exact hypernym and a substring of it, respectively). Both WCL models are thus equally robust in identifying hypernyms, whereas WCL-1 suffers from a lack of generalization in definition extraction (cf. Tables 2 and 3). Also, given that the ukWaC dataset contains sentences in which any of 239 domain terms occur, WCL-3 extracts on average 1.6 and 1.7 full and substring hypernyms per term, respectively. Hearst’s patterns also obtain high precision, especially when substrings are taken into account. However, the number of hypernyms returned by this method is much lower, due to the specificity of the patterns (62 vs. 383 hypernyms returned by WCL-3).

6 Conclusions

In this paper, we have presented a lattice-based approach to definition and hypernym extraction. The novelty of our approach is:

1. The use of a lattice structure to generalize over lexico-syntactic definitional patterns;
2. The ability of the system to jointly identify definitions and extract hypernyms;
3. The generality of the method, which applies to generic Web documents in any domain and style, and needs no parameter tuning;

---

5WCL has only one threshold value $\theta$ to be set for determining frequent words (cf. Section 3.1). However, no tuning was made for choosing the best value of $\theta$.

6We had to re-tune the system parameters on ukWaC, since with the original settings of Cui et al. (2007) performance was much lower.
4. The high performance as compared with the best-known methods for both definition and hypernym extraction. Our approach outperforms the other systems particularly where the task is more complex, as in real-world documents (i.e., the ukWaC corpus).

Even though definitional patterns are learned from a manually annotated dataset, the dimension and heterogeneity of the training dataset ensures that training needs not to be repeated for specific domains, as demonstrated by the cross-domain evaluation on the ukWaC corpus.

The datasets used in our experiments are available from http://lcl.uniroma1.it/wcl. We also plan to release our system to the research community. In the near future, we aim to apply the output of our classifiers to the task of automated taxonomy building, and to test the WCL approach on other information extraction tasks, like hypernym extraction from generic sentence fragments, as in Snow et al. (2004).

References


7Of course, it would need some additional work if applied to languages other than English. However, the approach does not need to be adapted to the language of interest.


