

GlossExtractor: A Web Application to Automatically Create a Domain Glossary

Roberto Navigli and Paola Velardi

Dipartimento di Informatica, Università di Roma “La Sapienza”,
via Salaria 113, Roma
{navigli,velardi}@di.uniroma1.it

Abstract. We describe a web application, *GlossExtractor*, that receives in input the output of a terminology extraction web application, *TermExtractor*, or a user-provided terminology, and then searches several repositories (on-line glossaries, web documents, user-specified web pages) for sentences that are candidate definitions for each of the input terms. Candidate definitions are then filtered using statistical indicators and machine-learned regular patterns. Finally, the user can inspect the acquired definitions and perform an individual or group validation. The validated glossary is then downloaded in one of several formats.

1 Introduction

Navigli and Velardi (2004) presented a technique, named *OntoLearn*, to automatically learn a domain ontology from the documents shared by the members of a web community. This technique is based on three learning steps, each followed by manual validation: terminology extraction, glossary extraction, and finally, ontology enrichment. The *OntoLearn* methodology has been enhanced, and experimented in the context of a European project, INTEROP (Velardi et al. 2007). Recently, we started to develop web applications to make freely available each of the steps of the *OntoLearn* methodology. The first web application, *TermExtractor* (Sclano and Velardi, 2007), was made available on <http://icl.uniroma1.it> about one year ago. We here describe *GlossExtractor*, a tool that receives in input a terminology T , i.e. a list of relevant domain terms, and automatically extracts from web documents one or more definitions for each term in T .

2 Summary of the Gloss Extraction Algorithm

Figure 1 shows the basic steps of the glossary extraction algorithm. The input to the system a list T of terms, for which a glossary has to be learned. Possibly, this list is the result of a previous terminology extraction process.

The first phase is candidate extraction: for each term, definition sentences are searched first, in on-line glossaries, then, in on-line documents. Simple, manually defined regular expressions are used to extract the candidate definition sentences.

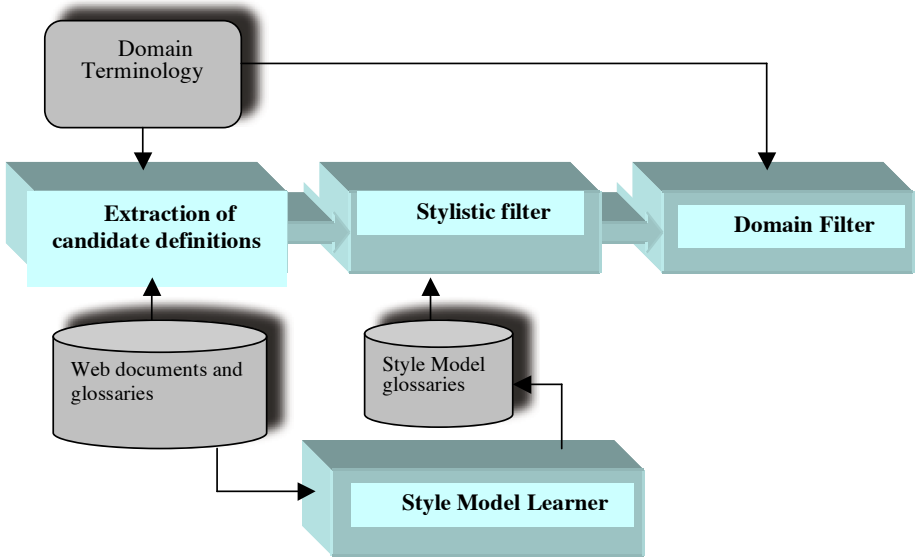


Fig. 1. The glossary extraction algorithm

Figure 2 shows an example of html page of a web glossary including definitions. To extract the relevant information (highlighted with arrows in Figure 2) it is necessary to perform layout and syntactic analysis of the html pages, in order to prune the noise. Similarly, Figure 3 shows an example of definition embedded in a web document. Here, not only graphical and irrelevant information has to be pruned, but also some simple regular expression must be defined to determine whether the sentence including a term $t \in T$ is a candidate definition. Examples of simple filtering patterns are: (“<term> is a ”)| (“<term> are the ”) (“<term> defines ”)| (“<term> refers to ”) (“<term> concerns ”)| (“<term> is the ”) (“<term> is any ”)| (“<term> is an ”) (“<term> is a kind of ”) (“<term> is defined (to)as ”) etc. These patterns (inspired by Hearst (1992) and subsequent works) are intentionally very simple, to reduce search time over the web.

The web-search step described above usually produces a large number of hits, including a considerable amount of noise. Noise is generated by two factors:

First, a sentence matching one of the above simple regular expressions is likely not to be a definition, e.g.: “**Knowledge management** is a contradiction in terms, being a hangover from an industrial era when control modes of thinking.”

Second, the sentence could indeed be a definition, but not pertinent to the domain, e.g. A **model** is a person who acts as a human prop for purposes of art, fashion, advertising, etc. that would not be pertinent to, e.g. a “software engineering” domain, but rather to a “fashion” domain. The two subsequent steps of the glossary extraction methodology are conceived in order to eliminate these two sources of noise.

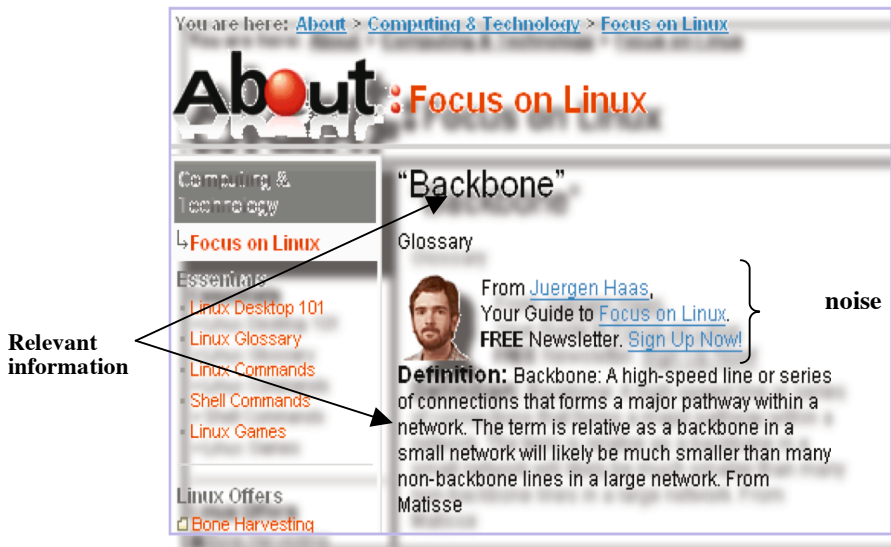


Fig. 2. Example of a definition in a web glossary page

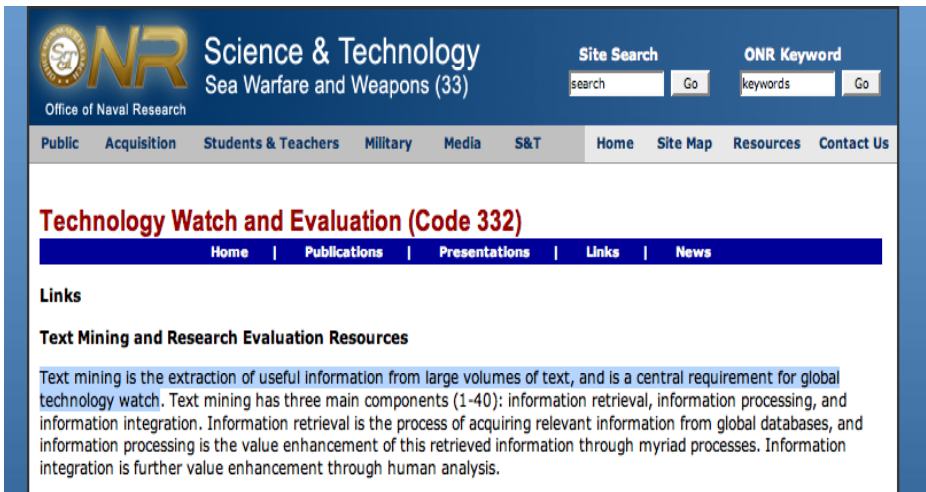


Fig. 3. Example of a definition embedded in a web document

2.1 Application of a Stylistic Filter

The objective of the stylistic filter is to select “*well-formed*” definitions, i.e. definitions expressed in term of *genus* (the *kind* a concept belongs to) and *differentia*

(what specializes the concept with respect to its kind), e.g. “*enterprise information integration is **the process** of integrating structured data from any relevant source for the purpose of presenting an intelligent, real-time view of the business to a business analyst or an operational application.*” In this definition, the phrase that identifies the genus is marked in bold. Not all definitions are well-formed in the above mentioned sense, e.g. “*component integration is obtained by composing the component's refinement structures together, resulting in (larger) refinement structures which can be further used as components*”, and many sentences being not well-formed are non-definitions, e.g. “*component integration has been recently proposed to provide a solution for those issues*”

Stylistic filtering is a novel criterion with respect to related literature on definition extraction, and has several advantages: i) to prefer definitions adhering to a uniform style, commonly adopted by professional lexicographers; ii) to distinguish definitions from non-definitions (especially when candidate definitions are extracted from free texts, rather than glossaries); iii) to be able to extract from definitions a *kind-of* information, used to arrange terms taxonomically.

To verify well-formedness, we use regular expressions that impose constraints on a sentence structure at the lexical, part-of-speech and syntactic level. Part of speech and syntactic elements are identified using an available parser, the TreeTagger¹. Figure 4 shows an example of sentence tagged with part of speech (POS) and segmented (chunked) according to syntactic categories. For example, DT and NN are POS for determiners (e.g. *the*) and nouns (e.g. *process*), respectively, while the sentence chunk “*The process*” is tagged with NC (noun phrase).

“Style” regular expressions have been automatically learned using a decision tree machine learning algorithm. We used the J48 algorithm from the *weka* machine learning web site (www.cs.waikato.ac.nz/ml/weka/). As input features for the algorithm, we used the first 5 POS/chunk tags pairs. Figure 5 shows an example of learned decision tree.

The *training set* (TS) used to learn style filters includes positive example of definitions from several on-line sources, and a set of manually extracted negative examples. Table 1 illustrates the training set composition. Notice that most negative examples come from evaluation experiments performed in the context of the already mentioned INTEROP project, during which our terminology and glossary extraction tools have been used to create an interoperability glossary (Velardi et al. 2007).

```
<NC> The DT process NN </NC><PC> of IN <VC> achieving VVG </VC></PC><NC> the DT
objectives NNS </NC><PC> of IN <NC> the DT business NN organization NN </NC></PC><PC> by IN
<VC> bringing VVG </VC></PC><ADVC> together RB </ADVC><NC> some DT resources NNS
</NC>. SENT
```

Fig. 4. Example of a sentence annotated with part of speech and syntactic tags (TreeTagger)

¹ TreeTagger is available at <http://www.ims.unistuttgart.de/projekte/corplex/TreeTagger/Decision/TreeTagger.html>

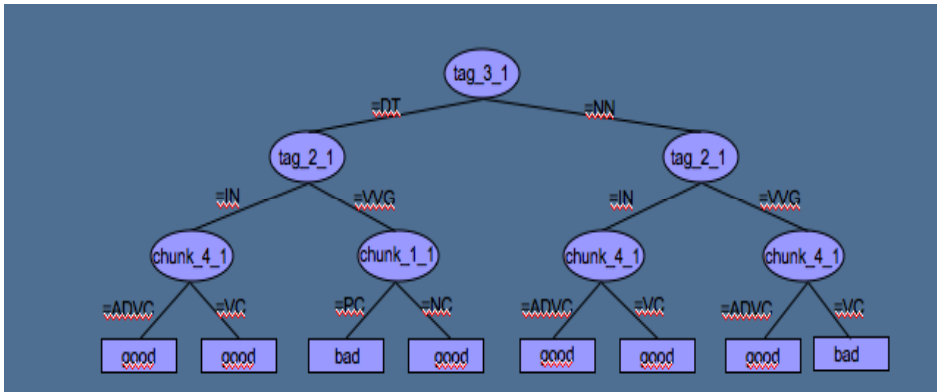


Fig. 5. A style decision tree learned with J48 machine learning algorithm

Table 1. Training Set used to learn a style filter for definition sentences

Examples	Art (AT&T ² , Tourism WordNet ³)	Computer Network (WordNet+STB ⁴)	Interoperability (Interop glossary)	
Positive	310+415	270+270	450	215+1220
Negative	80	60	50	2032

2.2 Application of a Domain Filter

The domain filter is used to prune candidate definitions that are not pertinent with the domain. A probabilistic model of the domain is obtained by analyzing the domain terminology. As explained in the introduction, the input to the glossary extraction algorithm is a terminology T , e.g. a flat list of single and multi-word terms.

From the set of word *components* forming the terminology T , a *probabilistic model of the domain* is learned, assigning a probability of occurrence to each word component. More precisely, let T be the lexicon of extracted terms, LT the set of singleton word components appearing in T , and let :

$$E(P(w)) = \frac{freq(w)}{\sum_{w_i \in LT} freq(w_i)}$$

be the estimated probability of w in D , where $w \in LT$ and the frequencies are computed in T . For example, if $T = [distributed\ system\ integration, integration\ method]$ then $LT = [distributed, system, integration, method]$ and $E(P(integration)) = 2/5$, since over 5 singleton words in T , integration appears twice.

² www.getty.edu/research/conducting_research/vocabularies/aat/

³ wordnet.princeton.com

⁴ <http://app.stb.com.sg/asp/tou/tou08.asp>

⁵ www.geek.com/glossary/

We then define and measure the domain pertinence of each extracted definition, as follows: let W_t be the set of words in $\text{def}(t)$, a candidate definition of t . Let $W'_t \subseteq W_t$ be the subset of words in $\text{def}(t)$ belonging to LT . Compute:

$$\text{weight}(\text{def}(t)) = \sum_{w \in W'_t, w \in LT} E(P(w)) \log(N_t / n_t^w) + \alpha \sum_{w \in LT, w \in t} E(P(w))$$

where N_t is the number of definitions extracted for the term t , and n_t^w is the number of such definitions including the word w . The log factor, called *inverse document frequency* in information retrieval literature, reduces the weight of words that have a very high probability of occurrence in any definition, regardless of the domain (e.g. “system”). The additional sum in this formula assigns a higher weight to those sentences including some of the components of the term t to be defined, e.g. “*Schema integration* is [the process by which *schemata* from heterogeneous databases are conceptually *integrated* into a single cohesive *schema*.]”

The domain pertinence is applied over definitions that are classified as such by the decision-tree classifier mentioned in previous section. An adjustable threshold ϑ is applied to every definition, in order to select only definitions for which $\text{weight}(\text{def}) \geq \vartheta$.

2.3 Evaluation of the Glossary Extraction Algorithm

We defined a novel validation policy: we extracted a set of non-definition sentences (yet matching the simple regular patterns of section 2) and we immersed definitions from on-line glossaries (not used during the style-learning phase) into the set of non-definition. We then computed precision and recall of the glossary filtering methodology. First, we created a test set of glossary and web definitions, as detailed in Table 2. To obtain examples of “good” definitions, we first extracted definitions from professional glossaries on the web (partially from the same sources as for the learning set, but different definitions), then, we searched the web for definitions of the same terms as in the glossaries, and finally we compared the glossary definitions with those extracted from web documents, to decide whether they were good definitions or not. The bad definitions were used as negative examples.

Since “good” definitions are professionally created, the test set can be considered what is usually called a “*gold standard*”. Notice that, in the literature, bad and good definitions are evaluated using the “*2-3 judges with adjudication*” policy, a technique that suffers from subjectivity: it is not very robust, and the judges often are not expert lexicographers (usually, the authors of the publication).

Table 2 shows that the test set included definitions also from domains that were not in the training set (compare with Table 1). This was decided to better test the generality of the style filter.

On this test set, we ran several experiments, to evaluate:

1. The ability of the system at correctly classifying good and bad definitions, when definitions are extracted only from glossaries;
2. The ability of the system at correctly classifying good and bad definitions, when these are extracted only from web documents;
3. The ability of the system at pruning out definitions which are good, but not pertinent with the selected domain (domain filter).

Table 2. Sources used to create the test sets

Domain	Glossary definitions	Web Definitions
<i>Art</i>	AT&T	-
<i>Economy</i>	Michigan University glossary ⁶ and Wikipedia ⁷	Web definitions for the same terms
<i>Medicine</i>	University of Maryland glossary ⁸	Web definitions for the same terms
<i>Interoperability</i>	Interop. glossary	-
<i>Computer networks</i>	-	Web definitions for the same terms
<i>Tourism</i>	WordNet and STB	-

Table 3 shows the result of experiment 1, and table 4 the results of experiment 2. Both have been obtained using *10-fold cross validation* technique. The evaluation measures are *accuracy*, *precision*, *recall* and *f-measure*, which are standard measures in the *Information Retrieval* and *Machine Learning* literature.

Table 3. Performance of the algorithm searching only web glossaries and Google's define

Set	#	Accuracy	Precision	Recall	F-Measure
Training (TR)	2105	0,863	0,831	0,935	0,880
Test (TS)	1978	0,862	0,897	0,889	0,893
TR+TS	4083	0,874	0,880	0,911	0,895

Table 4. Performance of the algorithm searching only web documents

Set	#	Accuracy	Precision	Recall	F-Measure
Training (TR)	402	0,864	0,854	0,859	0,857
Test (TS)	359	0,851	0,925	0,810	0,864
TR+TS	4371	0,874	0,876	0,860	0,868

Both tables highlight good results, even in comparison with the few available data in literature. The only performance data available in literature concern a task similar to glossary extraction, i.e. in the Question Answering TREC context, the sub-task "*answering what-is questions*".

To evaluate the performance of the domain filter in isolation (experiment 3), we created a test set composed by 1000 economy definitions and 250 medicine definitions (extracted both from web documents and glossaries). We then ordered the set of definitions, based only on the domain pertinence, computed on the economy

⁶ www.umich.edu/alandear/glossary

⁷ www.wikipedia.org

⁸ www.umm.edu/glossary

terminology. The first non-pertinent definition is found in position 571, and only 72 economy definitions appear in positions from 571 to 1000. Therefore the domain filter seems to be rather effective. Of course, with an appropriate selection of the threshold, it is possible to balance precision and recall at best: we stress that, in certain new domains, a high recall could be preferable to high precision.

Table 3 and 4 provide an “objective” evaluation of the system, since “good” examples come from professional glossaries, or have been compared with professional definitions. However, the experiments are, in a sense, “canned”, since the system is asked to analyze a pre-defined set of candidate definitions, and to classify them as good or bad using the style and domain filters.

To obtain an evaluation more close to the reality of system’s intended use, we performed another experiment, in which the validation is performed manually by the contributors of this paper, using their intuition. To exploit the evaluator’s experience, we used the interoperability domain. We repeated the experiment on a medical domain, since expertise in this domain was also available. Table 5 shows the results.

Notice that in this experiment the system was provided only with a set of terms, and all returned definitions were actually found on the web, either in glossaries or in documents. The table shows that, overall, the performance of the system is similar to that measured on the predefined test set (Tables 3 and 4). However, in this case we could not measure the “real” Recall, but only the Recall over the total number of extracted candidates, before pruning with the style and domain filters.

Notice that performance is similar across the two domains, but the coverage is considerably lower for interoperability terms, as expected: for new, or relatively recent domains, it is more difficult to find definitions, both in glossaries or in web documents.

Table 5. Performance of the extraction algorithm when using a “live” search with post-evaluation

	Interoperability	Medicine
Total number of submitted terms (T)	100	100
Total number of extracted sentences (from all sources) (E)	774	1137
Sentences over the threshold ϑ^9 (C)	517	948
Accepted by evaluators(A)	448	802
Precision (A/C)	81,27%	80,73%
Recall on positive (A/E)	86,65%	84,59%
F-measure	83,87	82,61
Terms with at least one positive definition (N)	51	96
Coverage (N/T)	51%	96%


⁹ The threshold is selected computing the average difference between the two consecutive definitions, when definitions are ordered by weight.

3 The Web Application

We here describe the web application that encapsulates the methodology described so far, named GlossExtractor, available on <http://cl.uniroma1.it/glossextractor>.

The user login and either accepts the default options or selects the Options button. In the option window, the user can first select the sources from which a glossary has to be extracted: i) web glossaries searched by the system or suggested by the user itself; ii) the Google's "define" feature, and iii) documents on the web, searched by GlossExtractor as described in previous sections. The user can also set the relevance threshold ϑ and select a single-user validation or group validation. In the group validation, a coordinator selects the initial and final date of the validation campaign, and then (s)he selects the list of user's e-mail (all users must –freely– subscribe to use the application). The other members of the validation team receive an e-mail to announce starting and ending dates.

Figure 6 is a screen-dump showing the subsequent steps of the workflow. In step 2, the user uploads the terminology T, or (s)he can run a demo session, where only one term is specified. In the demo session, the user can select the domain from a list of existing terminologies, but, if the term does not belong to any of these domains, (s)he will be presented with a list of selected definitions where only the style filter has been applied.

glossextractor 

GlossExtractor is a web tool for the automated acquisition of a glossary for an input terminology. Starting from domain terms, the software extracts relevant glosses from a number of resources (dictionary and glossary definitions, definitions within texts, etc.).

Step 1 - Options

Set the **GlossExtractor** options or just use the default options. From the options page you can upload text resources to input to GlossExtractor, set the relevance threshold for selecting glosses, and many other options.

Step 2 - Upload terminology or specify single term

This step allows you to specify the set of terms for which you need to extract glosses. Click [here](#) for information about the allowed format.

Terminology:

Alternatively, you can try a demo version of **GlossExtractor** by specifying a single term:

Term:

Step 3 - Assign a name

Please assign a name to the glossary. This name will be used to refer to the glossary in subsequent steps.

Glossary Name:

Step 4 - Process started

The extraction process of glossary "Interoperability" has been started. Due to our limited hardware resources, the process could request some time. An e-mail will be sent to you when the process is finished. The e-mail will contain a link through which you will be able to perform step four, namely the validation of the extracted glosses. In step four you will be able to accept or reject extracted glosses, and download the validated glossary.

Fig. 6. A Screen-dump of a GlossExtractor session

Finally, in step 3 a name is assigned to the glossary extraction task, and in step 4 the user is disconnected.

When the glossary extraction process is terminated, the user receives an e-mail and is directed to the validation page. Figure 7 shows a screen-dump of a multiple users validation page. For each definition, the page shows the computed weight and the source type from which the definition has been extracted (web glossary, Google’s define feature, web document). By clicking on the pencil icon to the left of each gloss, the user can modify the text of a definition if the definition is judged good, but not fully satisfactory. Manual changes are tracked by the system. The coordinator has a different view, in which he can inspect the global votes received by each definition.

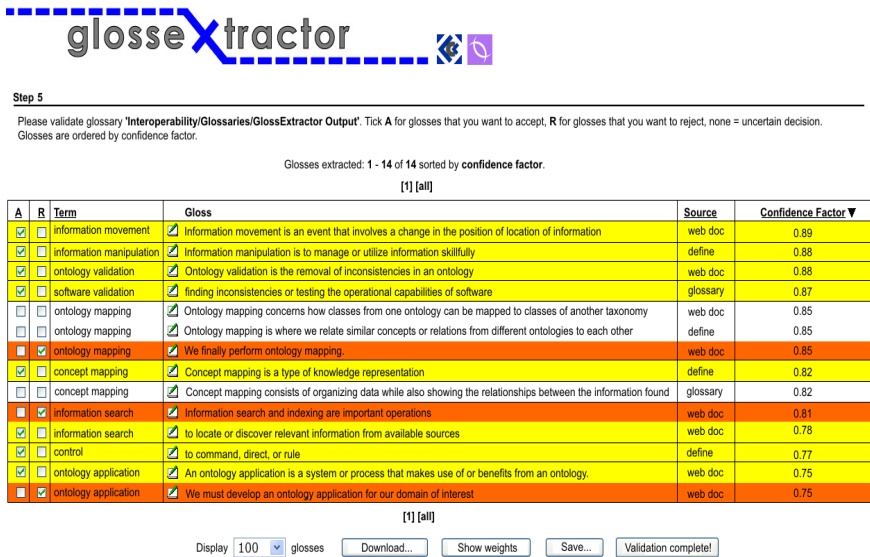


Fig. 7. Group Validation session

4 Related Work

Recently, significant progress has been made in using text mining methods to extract information from the web, for a variety of applications that rely on document meaning. The literature on automatic glossary extraction however is not very rich. In (Klavans and Muresan, 2001), and in other subsequent works by the same authors, it is described the DEFINDER system, a text mining method to extract embedded definitions in on-line texts. The system is based on pattern matching at the lexical level, guided by cue phrases as “is called” “is defined as” etc.. However, the application of lexical patterns on unrestricted documents (e.g. the web) may produce poor results in terms of precision and recall, especially the relevant but noisy “is a” pattern. A problem closely related to glossary extraction is that of answering “*what is x?*” questions in open domain question answering (QA). Many approaches presented

in QA literature, especially those evaluated in TREC conferences¹⁰, require the availability of training data, e.g. large collection of sentences tagged as “definitions” or “non-definitions”. The majority of methods are fully supervised e.g. (Miliaraki and Androutsopoulos, 2004) and (Ng et al. 2001). In (Androutsopoulos and Galanis, 2005) a weakly supervised approach is proposed, in which feature vectors associated to each candidate definition are augmented with automatically learned patterns. Patterns are sequences of n words (n -grams) occurring before or after the term for which a definition has to be found. This approach is more realistic, but the application of contextual patterns only at the lexical (word) level might not suffice to identify definitions in texts. In TREC conferences, the target is to mediate at best between precision and recall, whereas when the objective is to typify an emerging domain, recall is the most relevant performance figure. For certain novel concepts very few or possibly just one definition might be available, and the target is to capture the majority of them.

With reference to the literature, the work described in this paper has several novel features: i) the evaluation is rather more objective than standard three-judges with adjudication; ii) the extraction process is more sophisticated and fully general, since the supervised learning phase is non-domain dependent; iii) it has been fully implemented as a web application, which allows to extract not only individual definitions, but a complete domain glossary, and furthermore it supports a group validation. Finally, the method has been applied and validated with success in the “real” context of a research community on enterprise interoperability, as discussed in (Velardi et al. 2007).

References

- Androutsopoulos, I., Galanis, D.: A practically unsupervised learning method to identify single-snippets answers to definition questions on the web. In: HLT-EMNLP 2005. Proc. of the Human Language Technology Conference, Vancouver, Canada (2005)
- Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics, Nantes, France (1992)
- Klavans, J., Muresan, S.: Text Mining Techniques for fully automatic Glossary Construction. In: Proc. of the HTL2001 Conference, San Diego, CA (2001)
- Miliaraki, S., Androutsopoulos, I.: Learning to identify single-snippet answers to definition questions. In: Proceedings of COLING-2004, pp. 1360–1366 (2004)
- Ng, H.T., Kwan, J.L.P., Xia, Y.: Question answering using a large text database: A machine learning approach. In: Proceedings of EMNLP-2001, Pittsburgh, PA, pp. 67–73 (2001)
- Navigli, R., Velardi, P.: Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites. *Computational Linguistics* 50(2) (2004)
- Sciano, F., Velardi, P.: TermExtractor: a Web Application to Learn the Shared Terminology of Emergent Web Communities. In: Proc. of 3rd I-ESA '07, Madeira, March 28-30 (2007)
- Velardi, P., Cucchiarelli, A., Pétit, M.: A Taxonomy Learning Method and its Application to Characterize a Scientific Web Community. *IEEE Transaction on Data and Knowledge Engineering (TDKE)* 19(2), 180–191 (2007)

¹⁰ The Question Answering track page of TREC is <http://trec.nist.gov/data/qa.html>