Integrating Information to Bootstrap Information Extraction from Web Sites

Fabio Ciravegna, Alexiei Dingli, David Guthrie and Yorick Wilks
Department of Computer Science, University of Sheffield
Regent Court, 211 Portobello Street, S1 4DP Sheffield, UK

Abstract

In this paper we propose a methodology to learn to extract domain-specific information from large repositories (e.g. the Web) with minimum user intervention. Learning is seeded by integrating information from structured sources (e.g. databases and digital libraries). Retrieved information is then used to bootstrap learning for simple Information Extraction (IE) methodologies, which in turn will produce more annotation to train more complex IE engines. All the corpora for training the IE engines are produced automatically by integrating information from different sources such as available corpora and services (e.g. databases or digital libraries, etc.). User intervention is limited to providing an initial URL and adding information missed by the different modules when the computation has finished. The information added or delete by the user can then be reused providing further training and therefore getting more information (recall) and/or more precision. We are currently applying this methodology to mining web sites of Computer Science departments.

1 Introduction

Enabling large scale information extraction from the Web with limited user intervention is a relevant issue for the future of the Internet, especially in a Semantic Web perspective. The Semantic Web is based on the idea of semantically-based document annotation to be performed in order to allow both better document retrieval and empower semantically-aware agents. Most of the current technology is based on human centered annotation and is very often completely manual [10]. Manual annotation is difficult, time consuming and expensive. Convincing users to annotate documents for the Web (e.g. using ontologies) is difficult and requires a world-wide action of uncertain outcome. Moreover static annotation associated to a document can: (1) be incomplete or incorrect when the creator is not skilled enough; (2) become obsolete, i.e. not be aligned with pages updates; (3) be irrelevant for some users: a page in a pet shop web site can be annotated with shop-related annotations, but some users would rather prefer to find annotation related to animals. Therefore, producing methodologies for automatic annotation of pages becomes important. The initial annotation associated with the document loses its importance because at any time it is possible to automatically reannotate the document.

Information Extraction from text (IE) is able to produce automatic annotation, but porting to new application domains in fairly unconstrained domains such as the Web is out of reach of the current technology. The association to domain-specific ontology limits the domain and makes the application feasible. For example IE is currently used to reduce the annotation burden in some annotation tools [15] [9] [3]. Most of this technology is based on supervised learning, i.e. they require user-defined annotated corpora. When the task is complex and the documents to cope with present high variability in type (e.g. free texts are mixed with more or less rigidly structured pages), the amount of annotated material grows and supervised learning become unfeasible.

In this paper we propose a methodology to learn how to extract information from texts by integrating information from different sources. Information is extracted by starting from highly reliable/easy-to-mine sources such as databases and digital libraries. The extracted information is then used to bootstrap more complex modules such as wrappers which will collect more information which in turn will be used to train more sophisticated IE engines. All the corpora for training the IE engines are produced automatically by integrating information from different sources such as available corpora and services (e.g. databases or digital libraries, etc.). The natural application of such methodology is the Web, but large companies' information systems are also an option. In this paper we will focus on the Web, and in particular in mining web sites.

The first step to train an IE system is bootstrapping learning. In case of supervised learners, an annotated corpus is needed. The key feature of the Web that we exploit to enable bootstrapping is the *Redundancy* of information. Redundancy is given by the presence of multiple citations of the same information in different contexts and in different superficial formats. The redundancy is currently used for improving question answering systems [7]. When known information is present in different sources, it is possible to use its multiple occurrences to bootstrap recognisers that, when generalised, will retrieve other pieces of information, producing in turn more (generic) recognisers [2]. Information can be present in different formats on the Web: in documents, in repositories (e.g. databases or digital libraries), via agents able to integrate different information sources, etc. From them or their output it is possible to extract information with different reliability. Systems such as databases generally contain structured data and can be queried using an API. In case the API is not available (e.g. the database has a web front end and the output is textual), wrappers can be induced to extract such information. Wrapper Induction methodologies are able to model rigidly structured Web pages such as those produced by databases [11] [14]. When the information is contained in textual documents, extracting information requires more sophisticated methodologies. Wrapper induction systems have been extended to cope with less rigidly structured pages [8], free texts and even a mixture of them [4]. There is an obvious increasing degree of complexity in the extraction task mentioned above. The more the task is difficult, the less reliable generally the extracted information is. For example wrapper induction systems generally reach 100% on rigidly structured documents, while IE systems reach some 70% on free texts. Also the more the complexity increases, the more the amount of data needed for training grows: wrappers can be trained with a handful of examples whereas full IE systems can require millions of words [12].

In our model, learning of complex modules is bootstrapped by using information coming from simple reliable sources of information. This information is then used to annotate documents to train more complex modules. The redundancy of the Web generally allows us to do this. For example a simple wrapper can be used to extract information from a web page produced by a database containing papers from a computer science department (e.g. http://ciir.cs.umass.edu/publications/index.html). But the page must be annotated to induce the wrapper. There are two possibilities: (1) manually annotating some results of the database in order to train a wrapper; here user intervention is needed; if a dozen databases are to be coped with it is necessary to manually annotate examples for all of them; (2) look for a known database which contains some examples that can be found on each site to use to annotate the produced pages. For example in the case of the computer science department it is possible to use Citeseer (www.citeseer.com). Citeseer is a large database of papers in computer science. It is largely incomplete, but it is the first place where many scientists look for a paper. By wrapping Citeseer, it is possible to extract a number of examples that hopefully will be relevant to each of the databases. For example by querying both Citeseer and the CIIR bibliography using names of the UMass CS department, two pages containing papers lists will be produced. The one from Citeseer has a known format and the information can be extracted. Then, using simple information integration techniques, it is possible to automatically annotate some of the information in the CIIR page (e.g. for paper title generally it is necessary just an intelligent string matching). On these examples it is possible to induce wrappers that, given the high regularity of the information in the CIIR page, will be able to extract papers from CIIR. Considering that training a wrapper generally requires just a handful of examples, it is possible to focus only on those examples where the match is very clear and reliable, discarding examples that are more questionable, therefore producing a highly reliable wrapper. This is just an example of the idea. The more the task becomes complex, the more information is needed for training, the more reliable input data becomes difficult to identify.

In this paper we will present how we are using this idea to mine Web sites of Computer Science Departments. All the process is based on integrating information from different sources to provide annotations which will bootstrap learning which in turn will provide more annotation and so on. Also the process starts with simple methodologies which require limited annotation to produce further annotation to train more complex modules. In the next section we will describe the CS Department task and how the information from different sources is integrated to extract the desired information. Then we will discuss the generic architecture that we have used to build the application. Finally we will discuss some challenges that our experience highlights for both Information Extraction from text and for information integration.

2 The Computer Science Department Task

The application that will be used to describe the methodology is mining websites of Computer Science Departments. The goal is to discover who works in a specific department (name, position, home page, email address, telephone number) and extract the list of projects (including involved people) and to trace communities of practice, i.e. who works with whom and in what period. Moreover we want to extract for each person a list of published papers.

Finding People Names

In order to recognize people's names, a Named Entity Recognizer (NER) such as Annie (www.gate.ac.uk) is the most natural option. Unfortunately classic NER tend to be quite slow if launched on large sites (e.g. the 1,600 pages of the CS department at the University of Southampton) and can be quite imprecise on Web pages, as they are generally defined for newspaper-like articles. A two step strategy is used instead: initially a short list of seed names is found. The these seeds are used to bootstrap learning for finding further names.

Finding Seed Names

To find seed names, a number of weak strategies are combined that integrate information from different sources. First of all, the web site is crawled looking for strings that are potential names of people (e.g. using a gazetteer of first names and a regular expression such as ¡first-name¿$^+$(capitalized word)$^+$.). Then the following web services are queried:

- Citeseer (www.citeseer.com): Input: the potential name; Output: list of papers and a home page URL (if any);
- The CS bibliography at Unitrier (http://www.informatik.uni-trier.de/ ley/db/): Input: the potential name: Output: a list of papers (if any);
- HomePageSearch (http://hpsearch.uni-trier.de/): Input: the potential name; Output: a home page URL (if any);

- Annie (www.gate.ac.uk): Input: the potential name and the text surrounding it; Output: True/False;
- Google (www.google.co.uk) Input: the potential name and the site URL in order to restrict search; Output: Relevant Pages that are hopefully home pages;

The information returned by the digital libraries (Citeseer and Unitrier) is used to confirm or deny the name identity of the string. If they return reasonable results for a specific name (i.e. not too few and not too many), this name is retained as potential name. Defining what a reasonable result for a digital library is crucial here. If a string is a valid name, a number of papers are returned, otherwise the output is either empty or with unlikely features. For example when querying Citeseer with the term ”Smith” more than 400 papers are returned. This is the indication of a potential anomaly: the probability that a person writes so many papers is quite low and the name can be discarded. Equally, when looking for a non name (e.g. the words ”Fortune Teller”), no papers are returned. We tend to use quite restrictive criteria for keeping reliability high (e.g. more than 5 papers and less than 50 returned for Citeseer). The redundancy of information allows one to bootstrap learning using just a limited amount of information, as already noted by Brin [2]. The results of the digital libraries are integrated with those of the classic Named Entity Recognizer run on a window of words around the candidate (so to avoid the problem of slow processing). At this point a number of names of people are available. They are in principle of three types: (1) correct (they are people working for this department); (2) wrong (they are not people: they are false positives); (3) people who do not work at this site, but that are cited because, for example, they have co-authored papers with some of the researchers of the department. For this reason, Citeseer, Google and HomepageSearch are used to look for a personal web page in the site. If such a page is not found, the names are discarded. ¿From the Google’s results personal web pages are recognised with simple heuristics such as looking for the name in the title or in ”$< H1 >$” tags. The process mentioned above is meant to determine a small, highly reliable list of seed names to enable learning. Each of the strategies is, per se, weak, as they all report high recall, low precision. Their combination is good enough to produce data with high accuracy.

Learning Further Names

All the occurrences of potential names are then annotated on the site’s documents. Learning is performed initially only on documents where a reasonable quantity of known names are organised in XML structures such as lists and tables. Such structures generally have an intrinsic semantic: lists generally contain elements of the same type (e.g. names of people), while the semantics in tables is generally related to the position either in its rows or columns (e.g. all the elements of the first column are people, the second column represents addresses, etc.). When some elements (at least four or five in our case) are identified in a list or table, we try to train a weak classifier able to cover a large part of these examples and to account of the HTML structure (e.g. names are always the first element in each row). If we succeed, we are able to reliably recognize other names in the structure. Every department generally has one or more pages listing their staff in some kind of lists. These are the lists that we are mainly looking for, but also tables assigning supervisors and students are useful, provided that students and teachers can be discriminated. Each time new examples are identified, the site is further annotated and more patterns can potentially be learnt. New names can be cross-checked on the resources used to identify the seed list: we now have more evidence that these names are real names. In our experiments this is enough to discover all the staff of an average CS website with very limited noise, even using a strategy of multiple cross-evidence. We are currently using a combination of the following evidence to accept a learnt name: (1) the name was recognised as seed; (2) the name is included in an XML structure where other known occurrences are found (3) there is a hyperlink internal to the site that wraps the whole name; (4) there is evidence from generic patterns (as derived by recognizing people on other sites) that this is a person. The latter strategy was inspired by [13].

Extracting Personal Data

To extract personal data (email address, telephone number, position, etc.) it is necessary to identify a dedicated web page (e.g. a personal web page). Again we combine information from Citeseer, HomepageSearch and Google to check if the person has a known page in the current web site. Otherwise we look for occurrences in the site in which the name is completely included in a hyperlink pointing internally to the site. It is then possible to extract personal data from the home page using a named entity recognizer (e.g. Annie) to easily identify them. In case some of the personal data are not found, subpages linked in the home page are inspected. Only pages with an address under the same path are inspected (e.g. www.aaa.edu/ãdomine/index.html and www.aaa.edu/ãdomine/contact.html refer to the same subdirectory. This is unfortunately a very delicate step for which at the moment the strategy is very weak and a other strategies need to be found.

Discovering Papers Citations

Discovering what papers are written by the departmental staff is much more difficult than recognizing names and personal data. Here we focus on authors and title only. Authors are names in particular contexts (a paper citation and the author position; they must not be confused with editors of collections in which the paper can be published) A title is generally a random sequence of words (e.g. the tile of [7]) and cannot be characterized in any way (i.e. we cannot write generic patterns for identifying candidate strings as we did for people). Moreover paper titles must not be confused with titles of collections in which they are published. Nearly each department and each member of staff will provide a list of publications. Moreover papers are co-authored, so it is very possible that every paper is cited more than one time in a specific site. In rare cases personal lists of papers are produced using a departmental database (i.e. all the publication pages are formatted in the same way), but in most cases each person writes the list using a personal format; very often the style is quite irregular as the list is compiled manually in different moments of times. This is a typical case in which a wrapper-like approach does not work because it requires manually annotating at least some examples for each page for each member of staff. Also irregularities in style produce noisy data and classic wrappers are not able to cope with noise.

In order to bootstrap learning we query the digital libraries (Citeseer and UniTrier) using as keywords staff names we have discovered. The output for each name is hopefully a list of papers for each of them. Such lists will be incomplete because the digital libraries are largely incomplete. The titles in the list are then used to query a search engine to retrieve pages containing multiple paper citations. Again we focus on lists and tables where at least two papers are found. We use titles because they tend to be unique identifier. Again we are looking for seed examples, so we can discard titles which report too many hits (to avoid titles which are very common strings such as ”Lost”). As for discovering new names, the seed examples are annotated and the adaptive IE system is used to develop page-specific patterns. Again we favour examples contained in XML structures such as lists and tables for which we have multiple evidence. Please note however that the structure of the citation is not very structured internally. For example

```
Fabio Ciravegna, Alexiei Dingli,
David Guthrie and Yorick Wilks: Mining Web Sites using
Unsupervised Adaptive Information Extraction, in Proceedings of
the 10th Conference of the European Chapter of the Association for
Computational Linguistics, Budapest, Hungary, April 2003.
```

Simple wrappers relying on the XML structure only cannot be used. More sophisticated wrappers (such as [8] and [5])are needed. Using a cycle of annotation/learning/annotation we are able to discover a large number of new papers. Note that every time co-authorship among people is discovered in analysing the publication page of one specific authors, the paper is retained for annotation when the other names are considered (i.e. the redundancy is exploited again).

3 Generic Architecture for Website Mining

The CS website exercise is just one of the potential applications of the proposed technology. In order to make the technology scalable to a large number of cases, it is necessary to define a generic architecture portable in an easy way. We propose an architecture based on Web Services where each task is divided into subtasks. Each subtask is performed by a server which in turn will use other servers for implementing parts of the subtask. Each server exposes a declaration of input and output, plus a set of working parameters. Servers are reusable in different contexts and applications. For example one server in the CS department task will return all papers written by a person by accessing Citeseer. Another one will do the same on another digital library . The named entity recogniser server (whose role is to decide if a string is a name) will invoke these servers and integrate the evidence returned and decide if such evidence is enough to conclude that the candidate string represents a person.

Facilities for defining wrappers are provided in our architecture by Amilcare (nlp.shef.ac.uk/amilcare/), an adaptive IE system based on a wrapper induction methodology able to cope with a whole range of documents from rigidly structured documents to free texts [6]. Amilcare can be trained to work on rigid documents (e.g. Citeseer or Google output) by providing a handful of manually annotated examples, while it needs some hundreds of examples for more sophisticated cases [5]. All the servers are defined in a resource pool and can be inserted in a user-defined architecture to perform some specific tasks. New servers can be defined and added to the pool by wrapping them in a standard format. In the CS website task wrappers are defined for all the resources described in Section 2. The CS application works in the following way: a user submits a URL. The system returns a database populated with people’s names, personal details, papers, projects, etc. The defined architecture works as a ”Glass Box”. All the steps performed by the system are shown to the user together with their input and output. The user can check the intermediate results and manually modify their output, or change their strategy (if possible, such as in the case of modules who integrate information). For example if a person name is missed by the system, it can be manually added by the user. The modules that receive as input the output of that name finder will then be re-run and further information will hopefully be retrieved. In this way the user is able both to check the results of each step and to improve the results of the system by manually providing some contributions (additions, corrections, deletion).

4 Preliminary Evaluation

The architecture mentioned is fully implemented and we are currently experimenting extensively on a number of web sites. Currently, we have tested the different modules mainly in isolation. We experimented on the Computer Science Department of the University of Sheffield (www.dcs.shef.ac.uk) and separately on the site of the NLP group of the same department (www.nlp.shef.ac.uk). Experimental results are encouraging. Names of people can be found with a high reliability: in the case of the NLP group, all the member’s names were found. Only a limited number of spurious names were found (2/80[1]). Concerning paper discovery, evaluation is much more difficult, as a large amount of data must be checked manually. We have randomly checked the results on papers recognition for three researchers to perform a very preliminary evaluation. We focused on recognition of titles from three personal web pages. Recognizing titles, as mentioned, is very difficult, as a title is in principle a random sequence of words. We extracted from Citeseer 6 papers for each researcher and checked the ability to find new papers by mining the web site. We used only 6 papers for each person in order to be able to appreciate how the system behaved in case of limited available annotation. Of course the more information is available, the more the accuracy (as precision and recall) increases. The results are shown in Table 1.

For the first two researchers the results are excellent. For the third, the results are less good: The system was able to return 14 correct titles (including the seed ones) plus 2 wrong: in the latter cases the title of the paper was actually mistaken with that of the book in which the papers appeared. Recall was quite low because the personal publication page from

[1] There are currently 35 members in the group, but it is impossible to discriminate past and current members, as often their old home page is still available. The file date could be used as an indicator of people who have left.

	Seeds	Possible	Correct	Wrong	Precis	Recall
R1	6	59	47	0	100	79
R2	6	34	26	0	100	76
R3	6	40	14	2	85	35

Table 1: Examples found giving 6 seed examples for three researchers

which the papers were retrieved was highly irregular in its format, especially around the titles. Using more than 6 papers from Citeseer would have improved it. Citeseer actually returns 19 papers for this researcher, 1 duplicated and one wrong. We also did not use reseeding in the experiment, i.e. we did not use the information from one page to help recognizing information in another in case of coauthored paper, i.e. if a paper was coauthored by two of these researchers, the information returned for one person was not used to further annotate the publication pages of the second person. All in all in this experiment we just tested how much was possible to achieve with very little information, without exploiting most of the redundancy of information. We are currently performing more extensive experiments. They will be included in the final version of the paper.

5 Conclusion and future work

In this paper we have proposes a methodology to extract information from large repositories (e.g. the Web) with minimum user intervention. Information is extracted by starting from highly reliable/easy-to-mine sources such as databases and digital libraries. The extracted information is then used to bootstrap more complex modules such as wrappers which will collect more information which in turn will be used to train more sophisticated IE engines. All the corpora for training the IE engines are produced automatically by integrating information from different sources such as available corpora and services (e.g. databases or digital libraries, etc.). The user intervention is limited to provide an initial URL and to add information missed by the different modules when the computation is finished. The information added/delete by the user can then be reused for providing further training and therefore getting more information (recall) and/or more precision.

The natural application of such methodology is the Web, but large companies' information systems are also an option. In this paper we have focused on the use of the technology for mining web sites, an issue that can become very relevant for the Semantic Web, especially because annotation is provided largely without user intervention. It could potentially provide a partial solution to the outstanding problem of who is providing semantic annotation for the SW. The idea of using the redundancy of information to bootstrap IE learning is not new, having been already proposed by Brin [2] and Mitchell [13]. The difference with our approach is the way in which learning is bootstrapped. Brin uses user-defined examples, while Mitchell uses generic patterns that work independently from the place at hand (e.g. the site). We integrate information from different sources. The three approaches are not exclusive: in the CS application we also use the other two: projects names are bootstrapped using user-defined examples, generic patterns are used in the named entity recognizer. Integrating information from different sources is a further step in the direction of using the redundancy of information. In this respect our approach is - to our knowledge - unique. As noted by Brin, great care is needed in order to select only reliable information for annotation for learning. The integration of different knowledge sources multiplies the available information and therefore allows to use only information for which multiple evidence is found.

Challenges for IE

¿From the IE point of view there are a number of challenges in learning from automatic annotation, instead of using human annotation. On the one hand not all the annotation is reliable: the use of multiple strategies and combined evidence reduces the problem, but still there is a strong need for methodologies robust with respect to noise. On the other hand, many IE systems are able to learn from completely annotated documents only, so that all the annotated strings are considered positive examples and the rest of the text is used as a set of counterexamples. In our cycle of seed and learn, we generally produce partially annotated documents. This means that the system is presented with positive examples, but the rest of the texts can never be considered as a set of negative examples, because unannotated portions of text can contain instances that the system has to discover, not counterexamples. This is a challenge for the learner. At the moment we present the learner with just the annotated portion of the text plus a window of words of context, not with the whole document. This is enough to have the system learning correctly: the number of unannotated examples that become negative examples entering the training corpus is generally low enough to avoid problems. In the future we will have to focus on using machine learning methodologies that are able to learn from scattered annotation.

Integrating Information from Different Sources

The proposed methodology is based on using the redundancy of information. Information is extracted from different sources (databases, digital libraries, documents, etc.), therefore the classic problems of integrating information arise. Information can be represented in different ways in different sources from both a syntactic and a semantic point of view. The syntactic variation is coped with in the definition architecture definition step: when two modules are connected, a canonical form of the information is defined, e.g. the classic problem of recognising film titles as ”The big chill” and ”Big chill, the” can be addressed. More complex tasks are to be addressed, though. For example a person name can be cited in different ways: N. Weaver, Nick Weaver and Nicholas Weaver are potential variation of the same name. But do they identify the same person as well? When a large quantity of information is available (e.g. authors names in Citeseer) this becomes an important issue [1]. This problem intersects with that of intra- and inter-document coreference resolution. We are currently focusing on mining websites, because this allows us to apply some heuristics that very often solve these problems in a satisfying way. For example the probability that N. Weaver, Nick Weaver and Nicholas Weaver are not the same person in a CS website is very low and therefore it is possible to hypothesize coreference. Different is the case of ambiguity in the external resources (e.g. in the digital libraries). Here the problem is more pervasive. Querying with very common names (e.g. ”John Smith”) gives disappointing results because papers by different people are mixed up. This is not a problem in our approach because the information returned is used to annotate the site. Papers from people in other departments or universities will not introduce any annotations and therefore will not cause any problems. The same applies in case multiple home pages are returned: if they do not have an address local to the current site, the page is not used. In the generic case, though, this is a problem. We are currently using this strategy to recognize named entities from Reuters news and to find more information about a specific name. In this case we must know if the Ken Russell cited in a specific news article is the famous director or an MTI researcher. We are currently experimenting with a strategy that integrates evidence from lexical chains extracted from generic ontologies. The idea is that an MTI researcher and a director should produce different lexical chains (one concerning computers, the other concerning films).

Acknowledgements

This work was carried out within the AKT project (http://www.aktors.org), sponsored by the UK Engineering and Physical Sciences Research Council (grant GR/N15764/01) and involving the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University. Objectives are to develop advanced technologies for knowledge management and the Semantic Web.

References

[1] H. Alani, S. Dasmahapatra, N. Gibbins, H. Glaser, S. Harris, Y. Kalfoglou, K. O’Hara, and N. Shadbolt. Managing reference: Ensuring referential integrity of ontologies for the semantic web. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02*. Springer Verlag, 2002.

[2] Sergey Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, EDBT’98*, 1998.

[3] F. Ciravegna, Alexiei Dingli, Daniela Petrelli, and Yorick Wilks. User-system cooperation in document annotation based on information extraction. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02*. Springer Verlag, 2002.

[4] Fabio Ciravegna. Adaptive information extraction from text by rule induction and generalisation. In *Proceedings of 17th International Joint Conference on Artificial Intelligence (IJCAI)*, 2001. Seattle.

[5] Fabio Ciravegna. (LP)^2, an adaptive algorithm for information extraction from web-related texts. In *Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining held in conjunction with the 17th International Joint Conference on Artificial Intelligence*, 2001. Seattle, http://www.smi.ucd.ie/ATEM2001/.

[6] Fabio Ciravegna. Designing adaptive information extraction for the semantic web in amilcare. In S. Handschuh and S. Staab, editors, *Annotation for the Semantic Web*, Frontiers in Artificial Intelligence and Applications. IOS Press, Amsterdam, 2003.

[7] Susan Dumais, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. Web question answering: Is more always better? In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, Tampere, Finland, 2002.

[8] D. Freitag and N. Kushmerick. Boosted wrapper induction. In R. Basili, F. Ciravegna, and R. Gaizauskas, editors, *ECAI2000 Workshop on Machine Learning for Information Extraction*, 2000. www.dcs.shef.ac.uk/ fabio/ecai-workshop.html.

[9] S. Handschuh, S. Staab, and F. Ciravegna. S-CREAM - Semi-automatic CREAtion of Metadata. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02*. Springer Verlag, 2002.

[10] S. Handschuh, S. Staab, and A. Maedche. CREAM — Creating relational metadata with a component-based, ontology driven framework. In *In Proceedings of K-Cap 2001*, Victoria, BC, Canada, October 2001.

[11] N. Kushmerick, D. Weld, and R. Doorenbos. Wrapper induction for information extraction. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1997.*, 1997.

[12] S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schwartz, R. Stone, and R. Weischedel. Bbn: Description of the sift system as used for MUC7. In *Proceedings of the 7th Message Understanding Conference*, 1998. $www.itl.nist.gov/iaui/894.02/related_projects/muc/$.

[13] Tom Mitchell. Extracting targeted data from the web. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, San Francisco, California, 2001.

[14] I. Muslea, S. Minton, and C. Knoblock. Wrapper induction for semistructured web-based information sources. In *Proceedings of the Conference on Automated Learning and Discovery (CONALD), 1998.*, 1998.

[15] M. Vargas-Vera, Enrico Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. MnM: Ontology driven semi-automatic or automatic support for semantic markup. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02*. Springer Verlag, 2002.