Knowledge Capturing Tools for Domain Experts

Exploiting Named Entity Recognition and n-ary Relation Discovery for Knowledge Capturing in E-Science

Lars Bröcker
Fraunhofer IAIS
Schloss Birlinghoven
53754 Sankt Augustin,
Germany
lars.broecker@iais.fhg.de

Marc Rössler
Computational Linguistics
University of Duisburg-Essen
47048 Duisburg, Germany
marc.roessler@uni-due.de

Andreas Wagner
Computational Linguistics
University of Duisburg-Essen
47048 Duisburg, Germany
andreas.wagner@uni-due.de

ABSTRACT

The success of the Semantic Web depends on the availability of content marked up using its description languages. Although the idea has been around for nearly a decade, the amount of Semantic Web content available is still fairly small. This is despite the existence of many digital archives containing lots of high quality collections which would, appropriately marked up, greatly enhance the reach of the Semantic Web. The archives themselves would benefit as well, by improved opportunities for semantic search, navigation and interconnection with other archives.

The main challenge lies in the fact that ontology creation at the moment is a very detailed and complicated process. It mostly requires the service of an ontology engineer, who designs the ontology in accordance with domain experts. The software tools available, be it from the text engineering or the ontology creation disciplines, reflect this: they are built for engineers, not for domain experts. In order to really tap the potential of the digital collections, tools are needed that support the domain experts in marking up the content they understand better than anyone else.

This paper presents an integrated approach to knowledge capturing and subsequent ontology creation, called WIKINGER, that aims at empowering domain experts to prepare their content for inclusion into the Semantic Web. This is done by largely automating the process through the use of named entity recognition and relation discovery.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*linguistic processing*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*semantic networks*; I.2.6 [**Artificial Intelligence**]: Learning—*knowledge acquisition, concept learning*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*; I.5.3 [**Pattern Recognition**]: Clustering

General Terms

Algorithms

Keywords

Named Entity Recognition, Relation Discovery, Semantic Networks, Wiki Systems

1. INTRODUCTION

The Semantic Web can only flourish if enough content providers adopt it for the presentation of their content. This lack of adoption is the Achilles heel of the vision of the data web where humans and software agents can work side by side. The main reason for this lies right at the base of the Semantic Web: the creation of ontologies. The process needed to get to a working representation of a domain is too difficult for domain experts to do it on their own - a debilitating factor on the way to widespread adoption: the WWW did flourish simply due to the ease of marking up knowledge in HTML. This does not hold true for OWL or even RDF.

There are tools that deliver support in the process of creating an ontology, both from the domain of text engineering as well as from ontology engineering. But these tools are made for a selected audience: ontology engineers. This in itself is nothing bad, but it reduces the amount of growth of the Semantic Web to the availability (and affordability) of said engineers. Tools are needed that allow domain experts themselves to design and create ontologies tailored for their needs and domain corpora, if the Semantic Web is to come about on a grand scale.

But what is needed to create an ontology from a text corpus? First of all, an ontology can be seen as a graph structure, a semantic network. The nodes of this graph are the entities, i.e. the actors, topics and objects of the ontology, while the edges of the graph are the relations that exist between the entities. The task of automatically creating an ontology can be broken down into the following steps: first named entity recognition (NER) and second the detection of relations existing between those entities.

The detection and classification of proper names into predefined categories is called Named Entity Recognition (NER). The recognition of the categories PERSON, LOCATION and ORGANIZATION within the newspaper domain is especially well-studied as a part of the MUC-campaigns (Message Understanding Conferences) and can be conducted automatically with a performance beyond 0.9 F-measure for English texts [4]. The detection of relations between the entities of a corpus is a younger discipline, usually concerned with binary relations. Experiments on English newspapers show performance around 0.75 F-measure [8]. These advances facilitate a largely automated processing of text corpora into domain ontologies. This paper introduces an integrated web service-based framework called WIKINGER that does just that.

This paper is structured as follows: Section 2 gives an overview of the WIKINGER framework, sections 3 and 4 describe our work on named entity extraction, while section 5 describes the relation discovery part of the process. After that, section 6 highlights relevant related work, and we close with remarks on future works and the conclusion in sections 7 and 8.

2. WIKINGER - THE BIG PICTURE

WIKINGER[3], short for *Wiki Next Generation Enhanced Repositories*, aims at developing collaborative knowledge platforms for scientific communities. The collaboration is facilitated by selecting a Wiki as a presentation layer, and the knowledge contained can be organized via semantic relations. The resulting semantic Wiki can be extended, reorganized and commented on by all (registered) members of the particular scientific community. To setup and maintain the semantic network, NER-techniques are applied to the available domain-relevant documents (see section 3). The resulting annotations are the potential nodes of the semantic network that is constructed in a semi-automatic manner. The relations are proposed based on clusters of co-occurring entities (see section 5).

Figure 1 shows a view of the components that are part of the WIKINGER framework. It is built following a service-oriented architecture, its modules are loosely coupled, which allows need-driven reconfiguration of the system. The system itself uses a linked set of data repositories to perform its duties. The resource layer at the bottom of fig. 1 shows a drastically simplified view of the outside world: it contains arbitrary data sources that can be imported into the first of the repositories, i.e. the document repository. This repository provides the other services of the system with a versioned corpus of documents to work on. The processing services (e.g. for NER, relation discovery and creation of the ontology) use this repository as a source only. They feed their results into the metadata repository. It is linked to the document repository to uphold references to the original and it also provides versioned storage of the data. This ensures that the original corpus remains unchanged. The final repository contains the semantic model of the corpus. It makes use of both the document repository as well as the metadata repository. At the moment, the application layer takes the form of a wiki system, but other applications can easily be envisioned.

The architecture of WIKINGER is motivated by the assumption that many nodes of a domain specific semantic network occur in domain relevant texts and that these occurrences are proper names or expressions which can be extracted with NER-techniques.

The pilot domain of WIKINGER is contemporary history with a focus on the history of Catholicism in Germany. For that domain, the traditional NER categories PERSON, LOCATION, ORGANIZATION, and TIME/DATE expressions obviously carry crucial nodes for a domain specific semantic network. However, the domain experts desired additional categories, such as HISTORICAL-EVENT, BIOGRAPHIC-EVENT or ROLE. A ROLE is a function or a position a person holds (e.g. "bishop", "professor of theology") and is often part of a BIOGRAPHIC-EVENT, which may contain additional annotations such as LOCATION and TIME/DATE, as the following example shows:

```
<BIO-EVENT>
   <DATE>1936</DATE>
   <ROLE>archbishop</ROLE> of
   <LOC>Cologne</LOC>
</BIO-EVENT>
```

The HISTORICAL-EVENT describes events significant to the domain experts, such as the "Wall Street Crash of 1929", also called "Black Thursday". This category may contain embedded categories, too. The two event categories of the pilot domain are beyond the traditional NER task: Depending on the perspective, they either involve relation extraction or embedded categories. The corpus to annotate currently consists of approximately 150 monographs within a book series. The books were scanned and the text was OCR-extracted. The annotations of the resulting corpus will be used as potential nodes of the semantic network to be created.

Since the book series has a consistent layout structure, it was possible to preserve some layout information, such as the distinction between footnotes and other text. This distinction is helpful in order to detect a text unit specific to the texts of our domain called a "biogram". A biogram usually is a footnote that is provided the first time a person is mentioned in the text and comprises a short biography. These biographies usually are short and concise and tend to follow a predetermined structure. For instance, most of the biograms start with the name of the person, and some biograms present the single pieces of information separated by a particular delimiter such as semicolon or comma.

Thus, in most cases the person named at the beginning of a biogram is the one that the other annotations in that biogram relate to. While some of the information items also belong to persons that are related to the person described in the biogram (e.g. "his father was a <ROLE>prime minister</ROLE>") this assumption nevertheless holds true for the largest part of the corpus. This is very important for the relation discovery step, since all relations discovered in a specific biogram are linked implicitly to said person, although its participation in most of the relations is not readily apparent from their local contexts. Accordingly, they need to be

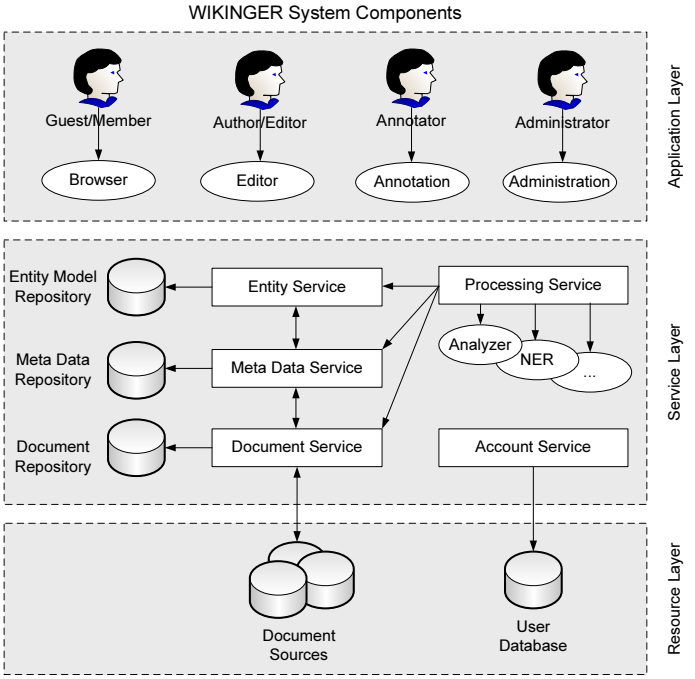

Figure 1: The WIKINGER Framework: Component View

associated with the person discussed in the biogram, which in turn has implications for the creation of the semantic network from the anntoation and relation data discovered in the course of the process.

Processing these biograms results in a semantic network in OWL which contains any information that could be harvested automatically from all the biograms within the 150 monographs. This knowledge base constitutes a biographical database for the scientific domain, which, according to the historians working within the WIKINGER project, is a long time desideratum for the domain of contemporary history of Catholics in Germany.

However, the tasks described are not limited to the pilot application of WIKINGER. Indeed, it has many features in common with a series of annotation tasks found in other domains as well. Our research within the WIKINGER project focuses on the application-oriented generalization of these challenges.

3. NER

It is highly desirable to generalize successful NER approaches described in section 1 to a broader variety of semantic markup at phrase level (i.e. apart from "standard" categories such as PERSON, ORGANIZATION, or LOCATION) in order to support other NLP applications. However, this requires annotation components that can be extended to new categories and adapted to new domains and new languages. These tasks may have different characteristics than the classical MUC task: First, they may lack the clue of the distinctive capitalization for some semantic classes and some languages, such as German. Second, the categories of interest may neither be obvious nor easily understandable due to a highly specialized domain and language.

A well-known example for such a task is the recognition of biomedical entities such as genes, proteins or cell tissue [6, 9]. It is almost impossible for a non-expert in the biomedical domain to judge about the correctness of an annotation or even to figure out a definition of the classes to recognize. Additionally, capitalization is not a distinctive feature of the entities to detect. Furthermore, biomedical entities are no proper names in the linguistic sense since a mention of a particular protein refers to all instances of that protein and not to a particular instance.

The annotation task within WIKINGER has similar characteristics: the documents to be processed are specialized texts, thus the definition of the annotation categories has to be provided by the domain experts. Also, most of the texts are in German, so the capitalization is not a reliable clue to detect proper names. Furthermore, discussions with the domain experts have shown that some of the annotation tasks amount to information extraction in a more general sense, in particular involving relation extraction, even though on a local level. For example, the BIO-EVENT provided in section 2 establishes a relation between the person the respective biogram deals with, a role occupied by that person, a certain time, and a location. Although these annotation tasks significantly expand the annotation of proper names, we still consider them as a sophisticated form of NER. In other words, we basically employ approaches which have been successfully applied to NER.

In principle, two major kinds of NER approaches have been proposed in the literature: rule-based and machine learning (ML) approaches. Rule-based approaches employ a handcrafted set of rules which is fine-tuned to the particular application domain. The adaptation of such a rather complex rule set to new domains and/or languages brings about ex-

tensive modification and maintenance efforts and requires therefore comprehensive knowledge about both the new domain and the proper design of the linguistic rule set. This means that domain experts need extensive support by computational linguists in order to port such a system to their domain. In contrast, adapting machine learning approaches to a new application domain requires the creation of domain-specific training data, i.e. manual annotation of domain-specific documents. Since this essentially requires domain (rather than linguistic) expertise, domain professionals need much less support by computational linguists (if any at all). Our experience within the WIKINGER project has shown that such support is necessary primarily for the initial task of defining a suitable set of semantic categories. During this definition stage, the communication between domain experts and linguists in essence consists in exchanging annotated examples. We believe that this *example-based communication* significantly facilitates portability, since concrete examples are much easier to create and understand than the explicit formulation of more or less complex and abstract (sub-)regularities. The same holds true for the annotation of the training data itself, which can be regarded as example-based communication between domain experts and machine learning algorithms.

Consequently, in order to minimize the amount of "external help" specialists needed to set up the WIKINGER system for their domain, we decided to employ ML approaches for NER. In our current experiments, we are using Maximum Entropy modeling and support vector machines. (As implementations, we employ openNLP[1] and SVMstruct[2], respectively.) However, we aim at providing a variety of ML algorithms which can either be employed independently or in combination to maximize performance. Regarding portability, it is crucial that the learning approaches employ domain-independent features and resources that can be easily adapted to a new domain or a new NER task. Furthermore, these methods have to be applied in a way that allows the acquisition of embedded annotations. "Standard" ML classifiers assign one class (in our case, a semantic category) to each instance to classify (in our case, a token)[3]. In embedded annotations, (parts of) entities may receive multiple classes simultaneously (e.g. in the example in section 2, "1936" is at the same time a DATE and part of a BIO-EVENT). To achieve such kind of concurrent classification, we run multiple classifiers, each one assigning different classes, and unify the results. For ML approaches which are restricted to binary classification (e.g. SVM), one classifier is required for each category. For ML approaches without this restriction (e.g. MaxEnt), classifiers assigning multiple classes can be built and combined in a more flexible way. Our experiments with MaxEnt models have shown that combining classifiers each of which assigns all categories except one, i.e. each of which "ignores" one particular class, yields higher performance than employing binary classifiers. In these experiments, we got F-measures (at token level) of up to 84.6% for persons, 87,1% for organizations, 94,8% for geographic-political entities, and 92,8% for roles.

[1]http://maxent.sourceforge.net/

[2]http://svmlight.joachims.org/svm_struct.html

[3]Multiword NEs are recognized as a sequence of tokens receiving the same class.

4. WALU

A prerequisite for enabling domain experts to create training data and control the process of training and (semi-)automatic semantic markup is the availability of a powerful and convenient tool. On the one hand, such a tool has to provide the necessary functionalities, i.e. manual annotation of documents, configuration and initiation of the training process, application of automatic annotation components, as well as inspection and correction of the resulting annotations. On the other hand, intuitive interfaces and convenient facilities supporting these functionalities while encapsulating their complexity are crucial to ensure usability for professionals of any domain. In addition, this tool has to be integrated into the overall WIKINGER infrastructure sketched in section 2. Currently there is no tool available that meets all these requirements (see section 6), at least not to our knowledge. Therefore, we are developing such a tool, which we call WALU (WIKINGER Annotations- und Lern-Umgebung = WIKINGER annotation and learning environment, see [16]).

WALU supports manual annotation with a GUI that is easy to use. It offers a comfortable navigation through the annotations, and simple but effective annotation support such as the automatic adjustment of markup boundaries or a dynamic markup dictionary. This dictionary is created during the annotation process and is used to propose markup labels for text passages corresponding to dictionary entries. Using a context-sensitive menu, the annotator confirms or rejects these proposals and/or removes the entry from the dictionary. In our experience the immediate feedback of the dynamic markup dictionary also helps the domain experts to clarify the task of string-based identification of domain-relevant concepts. Additionally, WALU also provides an automatic annotator for strings referring to the category DATE which is based on regular expressions. This is a simple prototype of a series of automatic mechanisms that will be used to annotate all the available documents. Except a few annotators based on regular expressions to classify entities with unique patterns (such as email addresses and URLs), most of these annotators are based on machine learning algorithms that will be accessible via WALU.

Training the ML facilities mentioned in section 3 as well as their annotation of new text can be initiated via the WALU GUI. The annotation results can be displayed and manually corrected. Automatic annotations are displayed in a distinct way (only the lower half of the annotated tokens are marked) so that they can be discovered immediately by the user.

WALU is designed both as a part of the WIKINGER infrastructure and as a stand-alone tool. Web-service-based communication facilities allow WALU to load documents from the WIKINGER document repository and load/store corresponding annotations from/to the metadata repository. As a stand-alone tool, WALU currently is able to import text documents (other import formats will be captured later) and to export annotated documents in a straightforward XML standoff format. The transfer between the various different data formats is achieved via a special internal format we call 'WaRP (WALU Rich Paragraph) stream', which is also processed by the automatic annotation components.

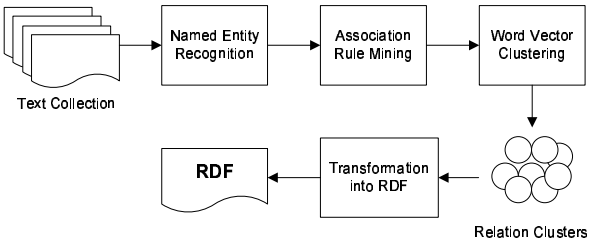

Figure 2: Workflow of the algorithm

5. SEMIAUTOMATIC RELATION DISCOVERY

The algorithms and tools described in the preceding sections provide named entities for a variety of project-dependent concept classes. They will become the nodes of the semantic network that is to be built. The remaining part is the provision of edges connecting these nodes, which will be explained in this section. The common approach to this problem is to let domain experts come up with a small number of relations and then to model them in an ontology editor. This requires knowledge of both ontology creation and ontology editors, which tends to be a too high hurdle for domain experts. Instead, we propose to do it based on the content of the corpus in question. With the named entities given by the preceding steps, relation discovery applying statistical methods becomes feasible.

5.1 Algorithm

Figure 2 shows the workflow of our approach. The first step, NER, has been covered already. The next step consists of the application of an association rule mining algorithm on the annotated corpus that has been segmented on the sentence-level. Only those sentences containing at least two entities are kept. Each sentence is represented by the set of entity classes appearing in it. These item sets serve as input for the *apriori* algorithm[1], that generates a set of association rules of the form $a \rightarrow b$. Each rule carries two parameters, support (the amount of observations supporting it), and confidence (in our case $\frac{\#(a \rightarrow b)}{\#a}$). Thresholds for these parameters can be used to influence the result of the algorithm.

The association rules can be ranked according to the two parameters. High support promises higher coverage, high confidence hints at a tighter correlation between the entity classes involved. Rules with more than one succedent tend to be more specialized, as evidenced by a higher confidence, and thus offer a higher potential information gain and they tend to be forgotten by the domain experts, when asked to come up with possible relations.

The next step is a clustering phase. It takes an association rule as input. The sentences of the rule are preprocessed, i.e. the named entities are replaced with their respective classes. This is done to receive generalized patterns of the relations in the sentences. Only the part between the outermost named entities is taken and transformed into word vectors. These weights of the vectors are created using tf*idf.

The goal of the clustering phase is to receive relation clusters, i.e. clusters in which every vector symbolizes the same relation. Since the amount of relation clusters is not known beforehand, agglomerative clustering is applied. In this algorithm, every vector starts as its own cluster. Clusters are then merged, given they fulfill a certain clustering criterion that is defined on a distance measure. We use standard Cosine similarity as distance and allow both single and complete linkage as criteria. Given two clusters A and B and a distance threshold t, this translates to:

$$Single\ Linkage : \exists \alpha \in A, \beta \in B : min(dist(\alpha, \beta)) < t$$

$$Complete\ Linkage : \exists \alpha \in A, \beta \in B : max(dist(\alpha, \beta)) < t$$

Which method will be used depends on the corpus in question. Terse texts show better results with complete linkage, normal text performs better with single linkage.

The result of this step is a set of relation clusters for each association rule. User interaction is needed at this point, in order to review the results and to provide meaningful labels for the relations. They are not generated automatically at the moment, but schemes employing parts-of-speech analysis (e.g. using the verbs) are feasible.

The last step of the algorithm is the transformation of the entities and their relations into an ontology language. The transformation process is a straight-forward affair for entities, classes and binary relations, since those can be handled by corresponding constructs in RDF. The transformation of n-ary relations is slightly more complex, since it involves blank nodes that act as a hub for the attachment of binary relations to the various members of the relation. The resulting RDF represents the ontology for the domain corpus.

In the use-case of our project, we have to deal with a dynamic corpus, since the articles from the wiki are fed back into the system to be analyzed. This continually updates the semantic network and keeps it on par with the wiki. But an additional step is required: relation classification. The relation clusters that have been committed in the initialization phase of the system are used for this task. New instances of sentences are marked up with named entities and are then transformed into word vectors which can be classified against the relation clusters, and subsequently transformed into RDF. Since the provenance of each triple in the ontology is known, exchanges can be restricted to those triples that are affected.

Preliminary evaluation results of the algorithm show F-measures ($F_1 = \frac{2 * Recall * Precision}{Recall + Precision}$) between 70% and 75% for clusters representing binary as well as n-ary relations. The algorithm usually creates more relation clusters than a human would, since humans tend to generalize the relations rather than to have a multitude of minuscule distinctions in their relation set. We have performed an evaluation of the performance of the algorithm against a part of the corpus relevant for the pilot application in the WIKINGER project. More details can be found in [2].

5.2 User interface

In order to provide the domain experts with an interface that facilitates directing the relation discovery process, the Wikinger Relation Discovery GUI, short WiReD, has been developed. It allows to view the results of the different steps of the algorithms and to experiment with different settings for them. This encompasses the association rules generated by the apriori algorithm as well as the composition of the relation clusters generated by the clustering phase.

Association rules can be selected manually for clustering, clusters can be post-processed (merged with others, deleted, renamed) and finally selected for inclusion into the semantic network. The parameters for each algorithmic step are preset with reasonable defaults, but can be changed directly from within WiReD, thus allowing experiments on the data set. This may sound intimidating at first reading, but in practice there are never more than two parameters per step in the processing chain, four parameters in total.

When the experts have come to a final result, i.e. they have agreed upon a set of relations they want to see included in the ontology, the relation information is fed back into the WIKINGER framework. Here it is used for different purposes. First of all it can be used to transform the information associated with it - the entities and their relations - into the ontology format of choice. If the corpus is static, this concludes the work needed for the ontology. In the case of dynamic corpora, e.g. wiki systems, the relation information approved by the experts is used to automatically classify new patterns that enter the system. These basically follow the same steps of the algorithm, only now in a fully automated mode. The experts can change the relation set anytime they want using the WiReD GUI which results in a total recalculation of the ontology to reflect their desire for change.

6. RELATED WORK

This section highlights related work in the areas touched by the work described in the sections above. We concentrate on annotation tools rather than individual NER algorithms, since the tools mentioned all encompass different approaches to NER. Following that, ontology learning environments are discussed, with a special regard to their use of relation discovery. Finally, algorithms partial to the discipline of relation discovery are discussed.

6.1 Annotation tools

As explained in section 4, the rationale behind WALU is its usability by professionals of any domain, in particular without computational or linguistic expertise. In this respect, WALU differs from other existing tools for semantic annotation, e.g. GATE [7], WordFreak [12], MMAX [13], or PALinkA [15]. These tools are primarily intended for users with a background in (computational) linguistics. Consequently, they are either tailored to different, more complex tasks than WALU (e.g. PALinkA for discourse annotation), or are designed as highly multifunctional tools (e.g. GATE, WordFreak, or MMAX). This multifunctionality allows their flexible application with regard to specific and complex needs. However, the price of this flexibility is that these tools require extensive configuration efforts which significantly affects usability for non-experts in computational linguistics. In this respect, WALU complements the range of existing tools.

6.2 Ontology learning environments

As has been pointed out above, ontology learning environments usually are built as supporting tools for ontology engineers. Their task differs from the one tackled by the approaches in this paper insofar as the ontology engineer has the process-knowledge necessary for building ontologies. He usually has access to different domain experts, and thus needs only marginal software support. Named entity recognition is employed sometimes to facilitate populating the ontology, whereas relation discovery is not used extensively, at least not to our knowledge.

Text-To-Onto[11] contains a module that calculates association rules to provide the engineer with an overview over possible interrelations between concept classes, but this approach is not followed further in the context of the application. Its successor, Text-2-Onto[5], employs a limited version of relation extraction, insofar as it searches for hyponym relation patterns (e.g. "x is a kind of y") in order to find additional instances of concept classes in a corpus. Relation discovery is not employed there.

6.3 Relation Discovery

Hasegawa et al [8] propose a system with a similar approach than the one presented here. They first perform NER on a text corpus, and then collect entity pairs from within sentences. These pairs are grouped by composition, the corresponding sentences are transformed into word vectors and a clustering step is performed on each of the groups. This results in a couple of relation clusters for each group. With some postprocessing (weeding out clusters below a certain size), they report F-measures of between 75% and 80% for selected clusters on a year of newspaper articles from *The New York Times*. In addition, they generate cluster labels by taking the words with the highest occurrence in each cluster. We believe that adding an association rule creation phase at the beginning helps in the selection of interesting combinations of relation candidates, even more so because we are not restricted to the detection of binary relations.

There are other approaches besides this one, that exploit syntactic structures and perform parts-of-speech analysis: Jiang et al. [10] analyze sentence grammar trees, model candidate relations in RDF in order to capture their direction and extract from the RDF a set of generalized relations. Navigli et al. [14] present an approach to ontology learning that exploits synsets from WordNet in order to disambiguate meaning and find relations that might hold between different entities from the sentences that explain the different synsets. But these approaches are dependent on deeper knowledge of the language of the text corpus. Approaches like Hasegawa's or ours only rely on statistics and the existence of annotated entities, thus they are language agnostic.

7. FUTURE WORK

Regarding NER, we will implement an interface to the Weka library [17], which comprises a number of machine learning algorithms. We will investigate combinations of different ML approaches either sequentially (i.e. the output of one

classifier is used as input to another one) or concurrently (i.e. several kinds of classifiers are run in parallel and a more-or-less sophisticated voting mechanism — which might involve a further ML approach — decides on the final classification).

Furthermore, we plan to provide an interface to the UIMA framework[4]. This way, further facilities for learning and preprocessing (e.g. morphological or syntactic analysis, which can provide useful information for semantic annotation as well as relation discovery) will become available to our framework. Since units from the UIMA framework can be provided as web services they can be added to complement the WIKINGER framework as needed.

Regarding relation discovery, we intend to apply our approach to other data sets, especially from the newspaper domain, in order to evaluate its performance on data sets that cover a wide range of topics, and to enhance the algorithm with a stage that extracts suitable labels for the relations and their members automatically.

The WIKINGER framework will be developed further, we intend to use it as a base platform for a variety of future projects.

8. CONCLUSIONS

This paper described a new approach to semi-automatic knowledge capturing from large text corpora. The goal is to empower domain experts to create domain ontologies themselves, without being dependent on the availability of ontology engineers. This is to be achieved by automating the process to a high degree, by employing named entity recognition (NER) and relation discovery. Domain experts are involved at those stages which require a substantial knowledge of the domain in question. Two software tools aiding in the process have been introduced that aid the domain experts in the task, WALU and WiReD. The former is a workbench for example-based NER, while the latter is a tool aiding in the relation discovery process.

Evaluation results for the different algorithmic solutions have been presented that show high values for F-measure for the automatic knowledge capturing methods.

All of this is part of a web service based architecture, the WIKINGER framework. It is used to create semantically enhanced collaborative knowledge platforms for scientific communities. The pilot application is a semantic wiki for the domain of contemporary history research regarding German catholicism.

9. ACKNOWLEDGMENTS

The work presented in this paper is being funded by the German Federal Ministry of Education and Research under research grant 01C5965. See http://wikinger-escience.de for further details regarding the project. The authors would like to thank Prof. Cremers from the University of Bonn and Prof. Hoeppner from the University of Duisburg-Essen for their helpful suggestions.

10. REFERENCES

[4]http://incubator.apache.org/uima/

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB conference*, pages 487–499, 1994.

[2] L. Bröcker. Semiautomatic Creation of Semantic Networks. In *Online-proceedings of PhD-symposium at ESWC 2007*, June 2007. no URL as of yet.

[3] L. Bröcker, M. Rössler, A. Wagner, et al. WIKINGER - Wiki Next Generation Enhanced Repositories. In *Online Proceedings of the German E-Science Conference*, 2007.

[4] N. A. Chinchor, editor. *Proceedings of the Seventh Message Understanding Conference*, Fairfax, VA, 1998.

[5] P. Cimiano and J. Völker. Text-2-Onto. In *Proceedings of NLDB 2005*, pages 227–238, 2005.

[6] N. Collier, P. Ruch, and A. Nazarenko, editors. *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004)*, Geneva, Switzerland, 2004.

[7] H. Cunningham. GATE, a General Architectur for Text Engineering. *Computers and the Humanities*, 36:223–254, 2002.

[8] T. Hasegawa, S. Sekine, and R. Grishman. Discovering Relations among Named Entities from Large Corpora. In *Proceedings of the Annual Meeting of Association of Computational Linguistics*, pages 415–422, 2004.

[9] L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6 (Supplement 1), 2005.

[10] T. Jiang, A. Tan, and K. Wang. Mining Generalized Associations of Semantic Relations from Textual Web Content. *IEEE Transactions on Knowledge and Data Engineering*, 1(2):164–179, 2007.

[11] A. Maedche. *The Text-To-Onto Environment*, chapter 7 in *Alexander Maedche: Ontology Learning for the Semantic Web*. Kluwer Academic Publishers, 2002.

[12] T. Morton and J. LaCivita. WordFreak: an open tool for linguistic annotation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, Edmonton, Canada, 2003.

[13] C. Müller and M. Strube. MMAX: A tool for the annotation of multi-modal corpora. In *Proceedings of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, Seattle, WA, 2001.

[14] R. Navigli, P. Velardi, and A. Gangemi. Ontology learning and its application to automated terminology translation. *IEEE Intelligent Systems*, 18(1):22–31, 2003.

[15] C. Orasan. PALinkA: A highly customisable tool for discourse annotation. In *Proceedings of the Fourth SIGdial Workshop on Discourse and Dialogue*, Sapporo, Japan, 2003.

[16] A. Wagner and M. Rössler. WALU — Eine Annotations- und Lern-Umgebung für semantisches Tagging. In G. Rehm, A. Witt, and L. Lemnitzer, editors, *Data Structures for Linguistic Resources and Applications*, pages 263–271. Gunter Narr Verlag, Tübingen, 2007.

[17] I. H. Witten and F. Eibe. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.