The TIGER Corpus Navigator

Sebastian Hellmann Jörg Unbehauen Christian Chiarcos Axel-Cyrille Ngonga Ngomo
University of Leipzig University of Potsdam
[hellmann|unbehauen|ngonga]@informatik.uni-leipzig.de
chiarcos@uni-potsdam.de

Abstract

Linguistically annotated corpora are a central resource in NLP. The extraction of formal knowledge from these corpora, however, is a tedious process. We introduce the Tiger Corpus Navigator, a Semantic Web system which aids users to classify and retrieve sentences from linguistic corpora – here, the TIGER corpus – on the basis of abstract linguistic concepts.

These linguistic concepts are specified extensionally, thus, independent from the underlying annotation: The user provides a small set of pre-classified sentences that represent positive or negative examples for the corresponding concept, and the system automatically acquires a formal OWL-DL specification of the underlying concept using an Active Machine Learning approach.

1 Introduction

A large number of annotated corpora have become available over the past years. Still, the retrieval of dedicated linguistic knowledge for given applications or research questions out of these corpora remains a tedious process. An expert in linguistics might have a very precise idea of the concepts she would like to retrieve from a corpus. Yet, she faces a number of challenges when trying to retrieve corresponding examples out of a particular corpus:

access she needs a tool that is able to process the format of the corpus, that is easy to deploy, and that provides an intuitive user interface

documentation she needs to be familiar with the annotations and the query language

representation she needs a representation of the results so that these can be studied more closely or that they can be processed further with other NLP tools.

In this paper, we describe a novel approach to this problem that starts from the premise that linguistic annotations can be represented by means of existing standards developed in the Semantic Web community: RDF and OWL[1] are well-suited for data integration, and they allow to represent different corpora and tagsets in a uniform way.

We present the TIGER Corpus Navigator, an Active Machine Learning tool that allows for the extraction of formal definitions of user-defined concepts and the corresponding examples out of annotated corpora. Based on initial examples given by the user, the navigator learns a formal OWL Class Definition of the concept that the user is interested in. This definition is converted into a SPARQL query[2] and passed to a triple store database with reasoning capabilities. The results are gathered and presented to the user to choose more examples, to refine the query, and to improve the formal definition. The data basis for the navigator is an OWL/RDF representation of the Tiger Corpus[3] and a set of ontologies that represent its linguistic annotations.

Our tool, available at `http://tigernavigator.nlp2rdf.org`, addresses and circumvents the barriers to the acquisition of knowledge out of corpora presented above: (1) it does not need any deployment and provides a user interface in a familiar surrounding, the browser, (2) the meaning of the identifiers used in the background corpus is made explicit by the tool, and, finally, (3) the navigator uses OWL; the results are thus represented in a readable, portable

[1] `http://www.w3.org/TR/rdf-concepts/`, `http://www.w3.org/TR/owl-ref/`

[2] `http://www.w3.org/TR/rdf-sparql-query/`

[3] `http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERCorpus/`

and sustainable way.

2 Tools and Resources

Several categories of tools and resources need to be integrated to enable the implementation of the goals presented above: We employ the **DL-Learner** (Lehmann, 2009) to learn class definitions for linguistic concepts; **NLP2RDF** (Hellmann, 2010) is applied for the conversion and ontological enrichment of corpus data; and the **OLiA ontologies** (Chiarcos, 2008) provide linguistic knowledge about the annotations in the corpus.

2.1 DL-Learner

The DL-Learner extends Inductive Logic Programming to Descriptions Logics, OWL and the Semantic Web; it provides a DL/OWL-based machine learning tool to solve supervised learning tasks and support knowledge engineers in constructing knowledge. The induced classes are short and readable and can be stored in OWL and reused for classification. OWL Class definitions form a subsumption hierarchy that is traversed by DL-Learner starting from the top element (*owl:Thing*) with the help of a refinement operator and an algorithm that searches in the space of generated classes. An example of such a refinement chain is (in Manchester OWL Syntax): *(Sentence)* $\rightsquigarrow$
(Sentence and hasToken some Thing) $\rightsquigarrow$
(Sentence and hasToken some VVPP) $\rightsquigarrow$
(Sentence and hasToken some VVPP and hasToken some (stts:AuxiliaryVerb and hasLemma value "werden")) . The last class can easily be paraphrased into: A sentence that has (at least) one Token, which is a past participle (*VVPP*), and another Token, which is an AuxiliaryVerb with the lemma *werden* (passive auxiliary, lit. 'to become'). Detailed information can be found in (Lehmann, 2009) and on the DL-Learner project site.[4]

2.2 NLP2RDF

NLP2RDF[5] is a framework that integrates multiple NLP tools in order to assess the semantic meaning of the annotated text by means of RDF/OWL descriptions: Natural language (a character sequence) is converted into a more expressive formalism – in this case OWL-DL – that grasps the underlying meaning and serves as input for (high-level) algorithms and applications.

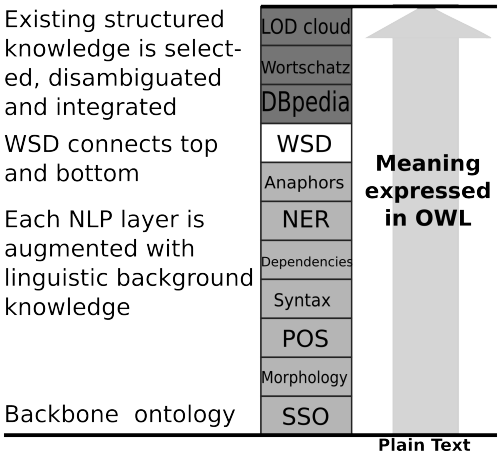

Figure 1: NLP2RDF stack

In a first step, sentences are tokenized and aggregated in a *Structured Sentence ontology (SSO)*. The SSO consists of a minimal vocabulary that denotes the basic structure of the sentence such as tokens and relative position of a token in a sentence.

The SSO (Figure 1 bottom) serves as the backbone model, which is then augmented by (1) features from NLP approaches (in dark grey), (2) rich linguistic ontologies for these features (combined in a *parser-ontology pair*), (3) background knowledge from the Web of Data[6] (in light grey), and (4) additional knowledge (either derived from steps 1-3 or created by this navigator).

2.3 Linguistic Ontologies

The Ontologies of Linguistic Annotations (Chiarcos, 2008, OLiA) represent an architecture of modular OWL-DL ontologies that formalize several intermediate steps of the mapping between concrete annotations, a Reference Model and external terminology repositories, such as GOLD[7]. The Reference Model provides the integrating terminology for different annotation schemes (OLiA Annotation Models). For the TIGER corpus navigator, we focused on the STTS Annotation Model[8] that covers the morphosyntactic annotations in the TIGER corpus. The usage of OLiA combined with NLP2RDF offers two major advantages: OLiA provides a growing collection of

[4] `http://dl-learner.org`

[5] Website `http://nlp2rdf.org`, Download (open source) `http://code.google.com/p/nlp2rdf/`

[6] e.g. Linking Open Data (LOD) Cloud `http://richard.cyganiak.de/2007/10/lod/`

[7] `http://linguistics-ontology.org/`

[8] `http://nachhalt.sfb632.uni-potsdam.de/owl/stts.owl`

more than 10 annotation models for more than 35 languages, that are interlinked with the OLiA reference model. The adaption of the navigator to other corpora and other languages is thus easily possible. The interlinking further allows to reuse learned classes on other corpora and even to learn on a combination of corporas.

3 The TIGER Corpus Navigator

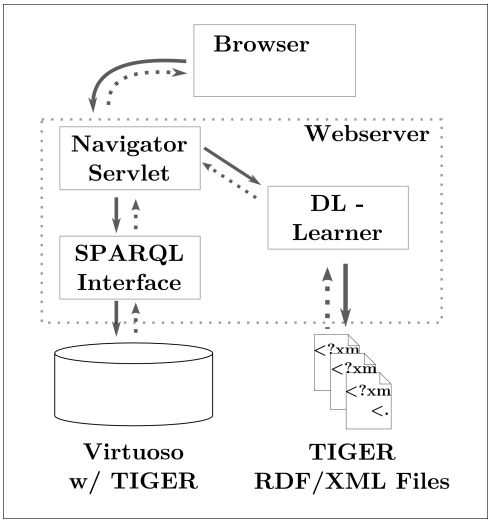

Figure 2: Technical architecture

Figure 2 shows the architecture of the Corpus Navigator: The Virtuoso triple store contains the whole corpus in RDF and allows queries over the complete data for correct retrieval, the data used by DL-Learner consists of one file for the OWL schema and 50,474 RDF/XML files (one per sentence), which it loads on demand according to the given examples.

With the Navigator user interface (Fig. 3), the user starts his research by searching for sentences with certain lemmas or words. The retrieved sentences are presented on the left side. They can be moved to the right panel and classified as positive or negative examples. Upon pressing the *Learn* button, they are sent to the DL-Learner and the learned OWL Class Definition is displayed (right top). The *Matching* button triggers the retrieval of matching sentences. The user can choose more positive and negative examples from the classified instances and iterate the procedure until the learned definition has an acceptable quality.

To aid the user during this process, the accuracy of the definition on the training data is given below the definition. Additionally, a count of matching sentences is displayed (in this case 5,299, ≈10%). Hovering over a named class in the concept description presents a tooltip explaining the meaning of the construct as specified by the OLiA Annotation Model. This allows to quickly gain insight into the annotations of the corpus and judge whether the learning result matches the conception of the user.

4 Evaluation

In this section, we evaluate recall and precision of automatically acquired concepts for passive identification in German. We describe two problems (with 4 experiments each), in which we variate several configuration options: training set size (how many examples a user needs to choose), learning time and usage of lemmas.

```
// root node
#root:[cat=/VP|S/] &
// root dominates "werden"
#root >* #werden:[lemma="werden"] &
// root dominates participle
#root >* #partizip:[pos=/V.PP/] &
// finite sentence
(#root >HD #werden |
// infinitive with zu
 (#root >HD #VZ:[cat="VZ"] & #VZ >HD #werden)) &
// root dominates participle directly
(#root >OC #partizip |
// participle is dominated by VP
 (#VP:[cat="VP"] >HD #partizip &
// root dominates VP directly
   (#root >OC #VP |
// or indirectly over a CVP
     (#root >OC #CVP:[cat="CVP"] & #CVP >CJ #VP))))
```

Figure 4: Rule for passive sentences in the Tiger Query Language (König and Lezius, 2003)

4.1 Experimental Setup

We consider the German passive that is formed by the auxiliary *werden* and a past participle (Schoenthal, 1975). The task is to distinguish passive clauses from other auxiliary constructions, given only information from the SSO and morphosyntactic annotations (parts of speech, pos). In the TIGER corpus, neither pos nor the linguistic surface structure alone are sufficient to distinguish passive from active clauses, so that information from both sources has to be combined.

For our experiment, the DL-Learner was trained on pos and lemmas. Syntax annotation, where passive is unambiguously marked was used only to identify target classifications (Fig. 4).

We distinguish three sets of sentences:

1. finite passive (finite auxiliary *werden*,

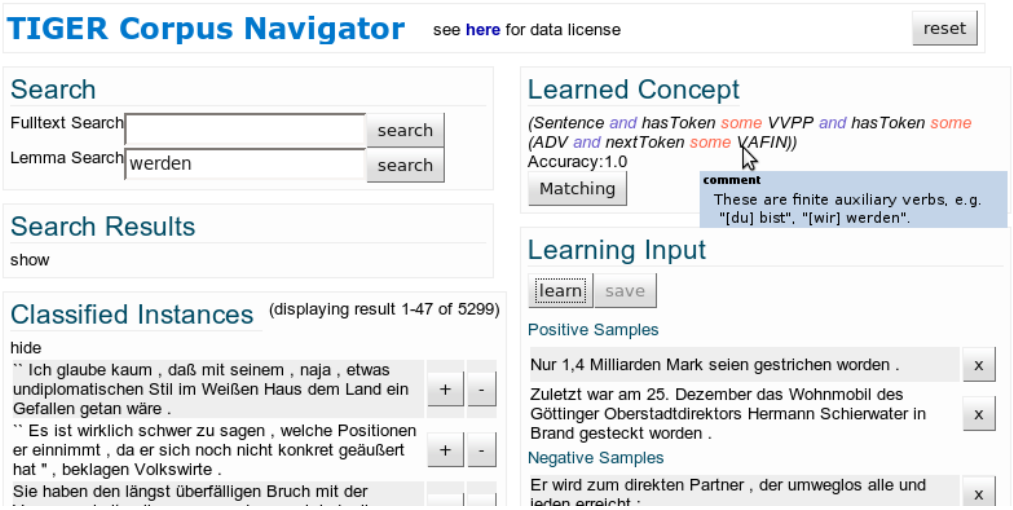

Figure 3: Screenshot of the TIGER Corpus Navigator (`http://tigernavigator.nlp2rdf.org`)

6,333 sentences, condition `#root >HD #werden`)

2. infinite passive with particle *zu* (lit. 'to') (37 sentences, condition `#root >HD #VZ`)

3. active (44,099 sentences that do not match the query)

From these sets we identified two learning problems to measure how well our approach can separate these sets from each another: (i) learn an OWL class that *covers* all finite passives (set 1) and the remainder (sets 2, 3), and (ii) distinguish between infinite passives (set 2) and the remainder. The second problem is especially difficult, as the number of correct sentences (37) is less than 0.07% of the total sentences.

We split the data for each problem into training and test data (both positive and negative) and removed 5 overlapping sentences.

As BASELINE, we randomly drew 5 positive (5p) and 5 negative (5n) sentences from the training data. In the experiments, we performed 4 iterations, starting with 5p+5n initial examples, and adding 5p+5n examples in every iteration. Precision and recall were measured on the intersection of retrieved sentences with the target classification.

As configuration variations for the first problem, we (1) adapted the max. execution time to three times the number of examples (ADAPT, 30s, 60s, 90s, 120s),[9] (2) we reduced the number of initial examples to 2p+2n and added 2p+2n for each iteration (REDUCE, total 4,8,12,16), and (3) we deactivated the inclusion of *owl:hasValue* (lemmas) in the classes (NO_LEMMA).

For the second problem, we (1) added 10 additional negative examples (ADD_10, total 20, 40, 60, 80), (2) also added 10n but adapted the runtime to 3 times the example size (ADD_10_X3, 60s, 120s, 180s, 240s), and (3) we used again the baseline (BASELINE) with no lemmas (NO_LEMMA).

For the first problem, we conducted a stratified leave-one-out 10-fold cross validation. As it was impossible to create 10 folds for the second set, we used a randomized 70%-30% split averaged over 10 runs (28 sentences for training, 11 for testing).

4.2 Results

Our results (summarized in Fig. 5) show that the TIGER Corpus Navigator is capable of acquiring concepts that involve multiple knowledge sources, here, the SSO (lemma) and the OLiA ontologies (for pos annotation) with a high recall and with reasonable speed.

The observed high recall is inherent in the learning algorithm: When exploring the search spaces, it automatically discards all classes that do not cover all positive examples, so it produces very general results. High precision, however, can only be achieved after a certain number of iterations or

[9] DL-Learner is an anytime algorithm, it stops when finding a class with 100% accuracy or a given maximum execution time (default 30 sec) is reached and returns its (intermediate) results.

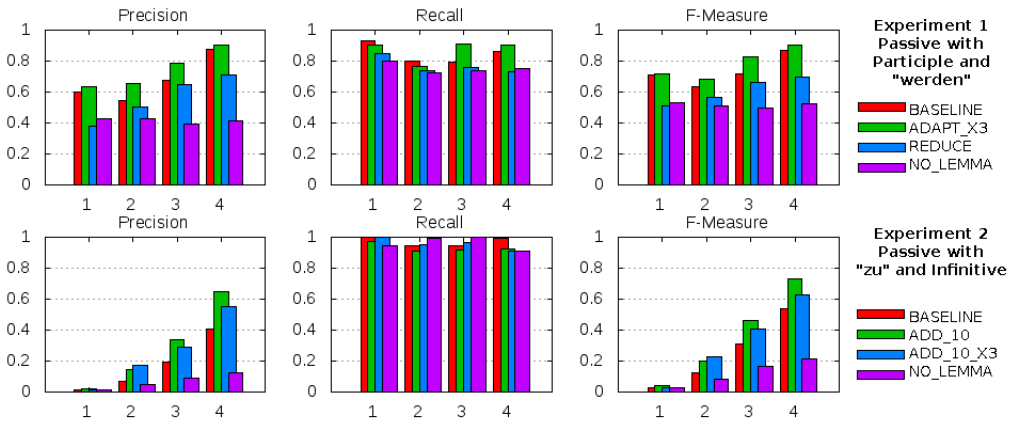

Figure 5: Evaluation results

by raising the *noise* parameter (zero in our experiments).

We found that our results are clearly dependent on lemmas, *owl:hasValue* inclusion yields better results. The selection of significant lemmas is done generically by DL-Learner according to a value frequency threshold, set equal to the number of positive example. Users could also wish to manually configure this parameter or give certain lemmas in advance.

The size of the training set greatly influenced the performance with about 20% lower F-Measure in iteration 4 (REDUCE vs. ADD_10 to BASELINE). We observed marginal effects by increasing the maximum learning time with a slight F-Measure gain of 3.5% (ADAPT_X3 vs. BASELINE) and even a loss of over 10% in the second experiment (ADD_10 vs. ADD_10_X3).

Although the second experiment amounts to a much lower F-Measure scores in iteration 4, achieved results are interpretable: 40 % precision and 99 % recall mean that the retrieved set of sentences was reduced to about 100 sentences of which 40 would be correct. Such a small sample would be suitable for manual inspection and post-processing.

Our implementation fulfills the speed requirements for a web scenario: Learning times for BASELINE were on average for the first experiment: 1.8 sec, 22.6 sec, 31.9 sec and 29.5 sec and for the second experiment 0.5 sec, 2.2 sec, 5.3 sec, 13.3 sec. The SPARQL queries needed 14.6 seconds on average and can be further improved by caching. The last example of the refinement chain in section 2.1 was one of the highest scoring learned classes.

5 Related Work and Outlook

Linguistic corpora can be accessed by several corpus tools, e.g., TGrep,[10] TIGER Search,[11] or GATE,[12] and newer tools also provide web interfaces, such as the Corpus Workbench[13], TrED,[14], or ANNIS.[15] All these tools, however, have in common that they operate on a formal, complex query language, that represents a considerable hurdle to their application by non-specialists.

The TIGER Corpus Navigator described in this paper represents an innovative approach to access corpus data that may complement such traditional corpus interfaces. The navigator provides access to the primary data of specific sentences on the basis of intuitively defined conceptual descriptions, and it is thus not even restricted to queries for concepts that are directly annotated as shown above for the passive concept in TIGER pos annotation.

So, the automatic acquisition of query concepts allows a relatively uninformed user to run queries against a database without being necessarily aware of the underlying data format. Thereby, our approach extends and generalizes approaches to access annotated corpora on the basis of abstract, ontology-based descriptions as described, for example, by Rehm et al. (2007) and Chiarcos et al. (2009). As opposed to these, however, the con-

[10] http://www.stanford.edu/dept/linguistics/corpora/cas-tut-tgrep.html

[11] http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERSearch/oldindex.shtml

[12] http://gate.ac.uk/

[13] http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/

[14] http://ufal.mff.cuni.cz/~pajas/tred/

[15] http://www.sfb632.uni-potsdam.de/~d1/annis

cepts are not pre-defined in our scenario, but rather acquired by the system itself.

In the introduction, we identified three elementary functions a corpus tool has to fulfill, i.e., to **access**, to **document** and to **represent** linguistic annotations. We presented the Tiger Corpus Navigator, which provides *access* via a an intuitve user interface over the Web. The paradigm of navigating a corpus based on example sentences rids the necessity of being familiar with the *documentation* beforehand. Even more so, only the necessary information is presented unobstrusively on-the-fly. Learned classes *represent* the results in a formal, yet easily understandable way and the evaluation has shown that it is possible to extract the desired information without much time or effort.

As for exchange and representation formats, the linguistic community still struggles to define its own standards, and, thus, several concurrent proposals are currently in use, e.g., NITE XML (Carletta et al., 2003), LAF/GrAF (Ide, 2007), or PAULA (Chiarcos et al., 2009). Here, standards from the Semantic Web community are applied, RDF and OWL, that are maintained by a large community and by a high number of tools. So far, only few NLP tools working with OWL are available, e.g., Aguado de Cea et al. (2008), but a number of linguistic resources has already been transformed to OWL-DL (Scheffczyk et al., 2006; Burchardt et al., 2008), or linked with ontologies (Hovy et al., 2006). Also, existing ontologies have been extended with concepts and properties for linguistic features (Buitelaar et al., 2006; Davis et al., 2008). The Navigator represents another step in this development of convergence of ontological and NLP resources.

Future Work includes the ability to save learned OWL classes. They can be collaboratively reused and extended by multiple users (Web2.0). Furthermore, they can be utilized to classify previously untagged text, converted by NLP2RDF in the same manner as here and thus extend the discovery of matching sentences beyond the initial corpora. With a matching parser-ontology pair it is even possible to replace the initial fulltext search by entering any example sentences.

References

G. Aguado de Cea, J. Puch, and J.. Ramos. 2008. Tagging Spanish texts: The problem of "se". In *LREC 2008*, Marrakech, Morocco, May.

P. Buitelaar, T. Declerck, A. Frank, S. Racioppa, M. Kiesel, M. Sintek, R. Engel, M. Romanelli, D. Sonntag, B. Loos, V. Micelli, R. Porzel, and P. Cimiano. 2006. LingInfo: Design and applications of a model for the integration of linguistic information in ontologies. In *LREC 2006*, Genoa, Italy, May.

A. Burchardt, S. Padó, D. Spohr, A. Frank, and U. Heid. 2008. Formalising Multi-layer Corpora in OWL/DL – Lexicon Modelling, Querying and Consistency Control. In *IJCNLP 2008*, Hyderabad, January.

J. Carletta, S. Evert, U. Heid, J. Kilgour, J. Robertson, and H. Voormann. 2003. The NITE XML Toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers*, 35(3):353–363.

C. Chiarcos, S. Dipper, M. Götze, U. Leser, A. Lüdeling, J. Ritz, and M. Stede. 2009. A Flexible Framework for Integrating Annotations from Different Tools and Tagsets. *TAL (Traitement automatique des langues)*, 49(2).

C. Chiarcos. 2008. An ontology of linguistic annotations. *LDV Forum*, 23(1):1–16.

B. Davis, S. Handschuh, A. Troussov, J. Judge, and M. Sogrin. 2008. Linguistically light lexical extensions for ontologies. In *LREC 2008*, Marrakech, Morocco, May.

S. Hellmann. 2010. The Semantic Gap of Formalized Meaning. In *ESWC PhD Symposium (submitted)*.

E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. 2006. Ontonotes: the 90% solution. In *HLT-NAACL 2006*, pages 57–60, New York, June.

N. Ide. 2007. GrAF: A graph-based format for linguistic annotations. In *Proceedings of the LAW Workshop at ACL 2007*, Prague.

E. König and W. Lezius. 2003. The TIGER language - a description language for syntax graphs, Formal definition. Technical report.

J. Lehmann. 2009. DL-Learner: learning concepts in description logics. *JMLR 2009*.

G. Rehm, R. Eckart, and C. Chiarcos. 2007. An OWL- and XQuery-based mechanism for the retrieval of linguistic patterns from XML-corpora. In *RANLP 2007*, Borovets, Bulgaria, September.

J. Scheffczyk, A. Pease, and M. Ellsworth. 2006. Linking FrameNet to the suggested upper merged ontology. *Frontiers in Artificial Intelligence and Applications*, 150.

Gisela Schoenthal. 1975. *Das Passiv in der deutschen Standardsprache. Darstellung in der neueren Grammatiktheorie und Verwendung in gesprochener Sprache*. Hueber, München.