COW14 Tokenization, Tagging (and Lemmatization)

Roland Schäfer
Linguistic Web Characterization (DFG)
Freie Universität Berlin
roland.schaefer@fu-berlin.de

Felix Bildhauer
Grammar Department
Institut für Deutsche Sprache Mannheim
bildhauer@ids-mannheim.de

Abstract

In this paper, we describe the COW tokenization and part-of-speech tagging pipeline with which we participated in the EmpiriST 2015 shared task. We briely discuss the original design goals for the tool chain and the minimal changes we made for the shared task. It should be noticed that we did not expect our system to perform competitively, especially not on CMC data and in the POS tagging track, and that we consequently viewed our system as an improved baseline system.

1 Original Design Goals

This section describes the original design goals of our toolchain.[1] Since we participtated with a production system only minimally adapted for the EmpiriST 2015 shared task, it is important to keep in mind the intended use and users of the system when evaluating the performance of and the errors made by the system.

The system with which we participated is the production system implemented for the construction of the COW and CommonCOW corpora that are described in Schäfer and Bildhauer (2012), Schäfer (2015), Schäfer (2016 to appear), but also (in comparison to other initiatives) in Biemann et al. (2013) and (implicitly) in textbook form in Schäfer and Bildhauer (2013).[2] Instead of developing custom tools for linguistic annotation—as we did for the non-linguistic preprocessing in the form of the *texrex* web page processor (Schäfer and Bildhauer, 2012; Schäfer, 2015; Schäfer, submitted)—we wrapped available tools and models (tokenizers, POS taggers and lemmatizers, morphological analyzers, named-entity recognizers, and dependency parsers). Section 2 describes technical details, including some improvements we achieved through custom pre- and postprocessing.

However, since the COW toolchain is a production system, it is vital to understand who our primary customers are. We have a background in theoretical linguistics, especially morpho-syntax and semantics. For empirical work in these fields, corpora have become a major source of data.

From log analyses of the queries made by our users at `https://webcorpora.org`, we know that POS tag specifications are used in only 14.8% (DECOW-only 13.7%), lemma information in 13.39% (DECOW-only 14.13%), and simple token specifications in 89.2% (DECOW-only 89.2%) of all queries. This illustrates that most linguists who use (our) corpora do not profit from high quality POS tagging—let alone other types of annotation—either because they are unaware of what such annotations can do for them, or because they cannot risk publishing corpus studies where material was not taken into account because the tagger made crucial errors, and queries did not return all the relevant targets. In other words, while low precision for corpus queries just means that corpus linguists have to filter their concordances by hand more thoroughly, below 100% recall runs the danger of invalidating corpus studies.

As an example, the named entity annotation in DECOW14 was added because some productive DECOW users work on the morphosyntax of German person names. After the real-life accuracy of the Stanford Named Entity Recognizer with the models from Faruqui and Padó (2010) was evaluated as inacceptable (Helmers, 2013), carefully designed heuristics and lists of names created by hand were used instead of the automatically generated annotation (Ackermann, to appear). The en-

[1] The tools have been freely available since 2014. They are distributed under a permissive 2-clasue BSD license on GitHub (`https://github.com/rsling/cow`).

[2] `http://corporafromtheweb.org`

tire automatic annotation was essentially useless, at least for the intended use.

Furthermore, users usually do not ask for more fine-grained POS annotations, but for more coarse-grained ones.

2 Implementation

Since we use our university's SLURM-based high-performance cluster and SLURM is best controlled from Bash scripts, all tools are wrapped in Bash scripts.

As corpus creators, we would like to mention our main woe is that, even in 2016, NLP tools generally do not come with the ability to process text in XML, which—in the simplest and totally sufficient case—means skipping over anything in <> and treating the five canonical XML entities as their literal counterparts. This is clearly the point where we lose most time and resources. The problem exists across the board with Ucto and TreeTagger discussed below, but also with all other tools we use, such as FreeLing (Carreras et al., 2004), the Stanford Named Entity Recognizer (Faruqui and Padó, 2010), mate-tools (Bohnet and Nivre, 2012), and Marmot (Mueller et al., 2013). Such problems are not usually tackled in shared tasks where accuracy is, of course, the only metric of interest, and usability in large production systems is less relevant.[3] That said, we now proceed to the technical details.

2.1 Tokenizer

We currently use the rule-based Ucto tokenizer for tokenization and heuristic sentence splitting (van Gompel et al., 2012). It is wrapped in a Bash script which also performs some pre- and post-processing. Unfortunately, much of this processing goes into keeping Ucto from separating material that should not be separated. In general, we find it highly difficult to write clean rule sets for Ucto without triggering completely unpredictable side-effects.[4] Also, Ucto's added functionality of being able to discern different types of tokens (numbers, dates, etc.), while an interesting (yet little documented) feature, makes writing rule sets complicated and unpredictable. As a consequence, we are currently designing our own rule-based tokenizer and are planning to move our entire rule set from Ucto to the new system.

Pre-processing includes:

- converting XML entities to literals because Ucto cannot deal with XML entities
- marking certain kinds of strings in a way that they are not broken up by Ucto, for example double names written as *Kay-M.*, file names, DOIs, ISBNs, content-type declarations, dates and numbers with periods (otherwise often detected as sentence ends)
- pre-processing quotes to make sure they are always treated as separate tokens
- pre-processing obfuscated email addresses with *[at]* instead of @.

In the Ucto rule set, we have rules (some copied from the generic Ucto profile for German) which recognize, for example,

- email addresses and URLs
- dates and numbers
- various special abbreviation-like tokens such as *H&M* and *C++*
- number-letter combinations that should not be split such as *90-fach* (*90-fold*)
- over 250 custom abbreviations (plus some regexes which detect abbreviations heuristically)

The post-processing mainly deals with restoring material that was protected from separation in pre-processing.

Virtually all of this was there before the EmpiriST 2015 shared task. We merely changed the way some tokens are treated. For example, we made the decision to keep single-word strings with asterisks like **freu** as one token but to split up similar multi-word strings such **total freu** as ** total freu ** (4 tokens). Also, we previously tokenized dates as one token (*2016-05-01*), simply because our users are not usually interested in dates at all, and the best we could do is keep the token count low for such strings.

2.2 POS Tagger

For POS tagging, we use the TreeTagger (Schmid, 1994) with the standard models. The Bash wrapper's main function is to set up a correct piping between different TreeTagger instances for tagging

[3] Please note that in many production systems, using UIMA wrappers is not feasible technically. The same goes for Python NLTK.

[4] Most side effects, we think, are due to the fact that Ucto compiles the rules into complex regular expressions for the ICU library. For example, we observed cases where the scopes of matched groups and replacement operators were obviously mangled, leading to unsolicited replacements.

and chunking with other scripts in between. The only improvements we implemented in the wrapper concern regular expression-based recognition of smileys and other emoticons as well as some *blank*-tokens inserted by our web page processing system *texrex*. This applies after TreeTagger. It cannot be implemented by pre-tagging, simply because introducing new POS tags for smileys would require an extended tag set and consequently a retrained tagger model.

Other than that we only improved lemmatization and POS tagging by amending the tagger lexicon. We sorted the frequency list of tokens lemmatized as *unknown* in a large subset of DECOW14A and manually created a 3,800 entries long lexicon addition for TreeTagger with POS and lemmas in order to take care of the most frequent unknown words.[5] While this necessarily also improves the quality of the POS tagging in our full corpus, it most likely did not improve the results in the EmpiriST 2015 shared task, simply because the sample is so small that the added words do not occur in it with high enough frequency.

3 Discussion of the Results

4 Summary and Outlook

As we pointed out, designers of production systems are faced with problems that are not within the scope of shared tasks, such as compatibility of tools with input formats. However, we suggest that discussions of changes or extensions to tag sets—a problem well within the scope of efforts like EmpiriST 2015—should involve the largest possible group of users because we see linguistic annotation purely as a service we provide to make our corpora more usable for our users. Therefore, we are planning to conduct a survey aong the users of our corpora and ask them how they use the linguistic annotation provided by us, and in which directions they would like to see them improve

Acknowledgments

Roland Schäfer's contribution to this paper was funded by the German Research Council (Deutsche Forschungsgemeinschaft, DFG) through the personal grant SCHA1916/1-1 *Linguistic Web Characterization*.

We would like to thank the high performance computing (HPC) unit of ZEDAT data center at Freie Universität Berlin for the provided computing time.

[5] We stopped at the point where the list contained less than one fixable *unknown* token in a window of thirty tokens. The remaning unknowns are productively formed compounds and noise.

References

Tanja Ackermann. to appear. From genitive inflective to possessive marker? – the development of german possessive -s with personal names. In Tanja Ackermann, Horst Simon, and Christian Zimmer, editors, *Germanic Genitives*.

Chris Biemann, Felix Bildhauer, Stefan Evert, Dirk Goldhahn, Uwe Quasthoff, Roland Schäfer, Johannes Simon, Leonard Swiezinski, and Torsten Zesch. 2013. Scalable construction of high-quality web corpora. *Journal for Language Technology and Computational Linguistics*, 28(2):23–60.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea, July. Association for Computational Linguistics.

Xavier Carreras, Isaac Chao, Lluís Padró, and Muntsa Padró. 2004. Freeling: An open-source suite of language analyzers. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC '04)*, pages 239–242.

Manaal Faruqui and Sebastian Padó. 2010. Training and evaluating a german named entity recognizer with semantic generalization. In *Proceedings of KONVENS 2010*, Saarbrücken, Germany.

Lea Arianna Helmers. 2013. Eigennamenerkenung in Web-Korpora des Deutschen - eine Herausforderung für die (Computer)linguistik. BA Thesis, Humboldt Universität, Berlin.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332, Seattle, Washington, USA, October. Association for Computational Linguistics.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, September.

Roland Schäfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight*

International Conference on Language Resources and Evaluation (LREC'12), pages 486–493, Istanbul. ELRA.

Roland Schäfer and Felix Bildhauer. 2013. *Web Corpus Construction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, San Francisco.

Roland Schäfer. 2015. Processing and querying large web corpora with the COW14 architecture. In Piotr Bański, Hanno Biber, Evelyn Breiteneder, Marc Kupietz, Harald Lüngen, and Andreas Witt, editors, *Proceedings of Challenges in the Management of Large Corpora 3 (CMLC-3)*, Lancaster. UCREL.

Roland Schäfer. 2016, to appear. CommonCOW: massively huge web corpora from CommonCrawl data and a method to distribute them freely under restrictive EU copyright laws. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC '16)*.

Roland Schäfer. submitted. Accurate and efficient general-purpose boilerplate detection for crawled web corpora. Accepted with revisions by Language Resources and Evaluation Journal.

Maarten van Gompel, Ko van der Sloot, and Antal van den Bosch. 2012. Ucto: Unicode tokeniser. version 0.5.3. reference guide. ILK Technical Report ILK 12-05, Induction of Linguistic Knowledge Research Group, Tilburg Centre for Cognition and Communication, Tilburg University, Tilburg, November.