An Environment for Relational Annotation of Political Debates

André Blessing[1], **Nico Blokker**[2], **Sebastian Haunss**[2],
Jonas Kuhn[1], **Gabriella Lapesa**[1], and **Sebastian Padó**[1]

[1]IMS, University of Stuttgart, Germany
[2]SOCIUM, University of Bremen, Germany

Abstract

This paper describes the MARDY corpus annotation environment developed for a collaboration between political science and computational linguistics. The tool realizes the complete workflow necessary for annotating a large newspaper text collection with rich information about *claims* (demands) raised by politicians and other actors, including claim and actor spans, relations, and polarities. In addition to the annotation GUI, the tool supports the identification of relevant documents, text pre-processing, user management, integration of external knowledge bases, annotation comparison and merging, statistical analysis, and the incorporation of machine learning models as "pseudo-annotators".

1 Introduction

Scalable text analysis techniques can open corpora to new questions in computational social sciences and digital humanities. This goal can be greatly facilitated with an environment for cross-disciplinary corpus access that supports the design and refinement of analysis categories and models – equally well at the conceptual and the natural language processing (NLP) level. It thus also invites a mixed-methods approach (Kuhn, to appear) towards more far-reaching research questions – combining the strengths of scalable computational models and the expert view on contextualized text instances.

This paper describes the MARDY tool, an interactive annotation environment for *political claims analysis* in computational political science (see Padó et al. (2019) for a task analysis and initial modeling results). The term claim is operationalized as a textual span containing a demand, proposal, criticism, or a decision made by actors active in the respective field (Koopmans and Statham, 1999). For example, a commentator may put forward the claim that the voting age be lowered to

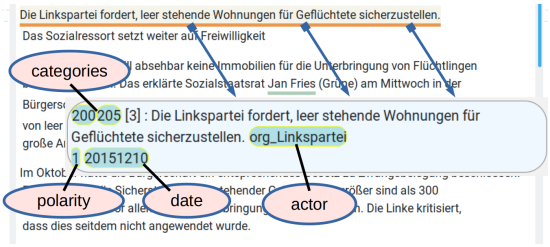

Figure 1: Annotation for the text span: "The Left Party demands that vacant flats be secured for refugees." which includes claim category 205 (forced occupancy), the actor *Linkspartei* (Left Party), and positive polarity.

16; a political party may propose that a government help (or deter) refugees. Figure 1 shows a typical example of a claim from a German domestic politics debate on immigration: A sentence is identified as containing a claim, the claim is categorized according to an annotation scheme, and assigned an actor, a polarity, and a date.

Political Science Background. Understanding the structure and evolution of political debates is central to understanding democratic decision making (Haunss and Hofmann, 2015). Therefore, an important research strand in political science aims at modeling and analyzing the exact mechanisms of political discourse, such as the formation of *discourse coalitions* out of *actors* (Hajer, 1993).

Discourse network analysis (Leifeld, 2016) builds on top of claims analysis (Koopmans and Statham, 1999), representing debates as graphs and analyzing their structure and dynamics. Actors and claims are represented as the two classes of nodes in a bipartite *affiliation network*. In Figure 2, actors are circles, claims are squares, and they are linked by edges that indicate support (green) or opposition (orange). A discourse coalition is the projection of the affiliation network on the actor side (dotted edges), while the projection on the concept side yields argumentative clusters. Affiliation networks open up a systematic view on conjectured discur-

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics-System Demonstrations, pages 105–110
Florence, Italy, July 28 - August 2, 2019.

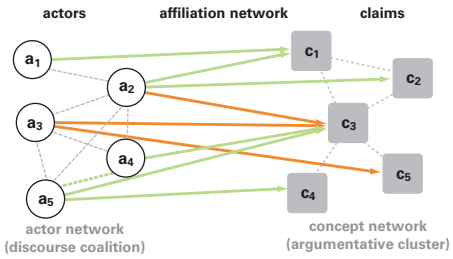

Figure 2: Actor, affiliation, and concept networks

sive patterns, e.g. regarding the stability/variability of coalition configurations in response to external events. To warrant a sufficiently focused analysis of discursive patterns, it is common to restrict attention to the debate in a given topical issue field. In the issue field of internal security policy, e.g., a recurring pattern could be hypothesized as follows: whenever a terrorist network is on the news, claims are made that police should receive more funding.

Annotating debates in corpora. To establish an empirical basis for research into the dynamics of political discourse, a systematic annotation methodology (*coding*, in the social science terminology) needs to be defined. The overarching goal is to identify and label claims brought forward by specific actors in a corpus covering public discourse, of the news coverage thereof, from a predefined time span. Within this scope, it is important to come as close as possible to discovering and annotating all claims made during the researched period. The granularity of distinct claim types adopted in the analysis has to be carefully chosen to ensure that different formulations of the same claim (claims with the same *substance*) are aggregated, while related claims have to be differentiated when this is relevant for the evolution of discourse dynamics. A debate-independent inventory of claim categories is hence impossible, and the development of a so-called *codebook* specifying annotation guidelines for relevant claim types is a crucial part of every issue-specific study (typically going through a cycle of revisions before freezing the claim types).

Manual vs. automatic annotation. To ensure reliability in the face of complex statements and of ambiguities only resolvable with world knowledge, annotation for political claims analysis has traditionally proceeded almost exclusively manually. There is however considerable potential for computer-aided annotation approaches that may increase the speed and consistency of annotations. This contribution demonstrates a technical architecture supporting researchers from political science in the full cycle of corpus selection, codebook development, (parallel) annotation, annotation adjudication and consistency checking. Although various tools for annotation-related subtasks in corpus linguistics and NLP development exist, the specific interleaving of the various workflow steps in corpus-based social science and digital humanities calls for an integrated architecture.

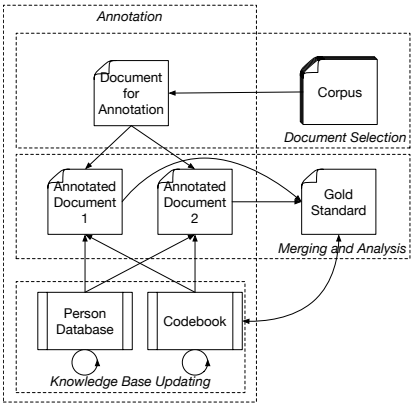

Figure 3: Workflow for Political Claims Annotation

The MARDY tool we present is a general environment for annotating of articles for political claims analysis. It extends ideas and components from earlier computational social science and digital humanities projects.[1] In a usability study on German newspaper texts (see Section 4), users reported improved guidance over traditional annotation procedures, and the integration of Machine Learning (ML) predictions as (pseudo-)annotators provides a unified interface for experiments with manual and automatic annotation.

2 Annotation Workflow Requirements

Figure 3 shows a typical annotation workflow that applies to political claims analysis as well as to related annotation tasks (see Section 5). The four dashed boxes with labels in italics show the major tasks involved, each of which comes with a number of desiderata.

Document Selection. We assume that annotation is performed on the basis of a potentially large overall corpus where full annotation of all documents is not feasible or desired. Thus, the first task is the *selection of relevant documents* for annotation. This is essentially an information retrieval task, where keyword-based approaches face the typical prob-

[1] Specifically, *e-Identity* (Blessing et al., 2015) and CRETA (Blessing et al., 2017).

lem of resulting in either high recall–low precision scenarios (too few keywords) or low recall–high precision scenarios (too many keywords).

Annotation. Since projects typically involve several annotators, the environment should not just support *annotation* proper, but also *user administration* (user management, task assignment).

Assuming that we annotate relations between actors and their claims, the annotation links markables to external knowledge bases, specifically the actors to a database of persons and other relevant entities (parties, companies, geopolitical entities etc.) and the claims to an ontology of claim categories (the *codebook*). The environment should support the integration of external knowledge bases for this purpose as seamlessly as possible.

Merging and Evaluation. Following best practice in both political science and NLP, we carry out double annotation of the relevant documents. These independent annotations need to be combined into a gold standard and merged by an expert adjudicator where they diverge. Our merging system also allows the integration of automatic generated annotations, which is particularly useful to counteract oversights, thus improving recall (see Section 4).

Knowledge Base Update. To support an evolutionary process of the analytical categories, both the actor and the claims knowledge bases (KBs) must be modifiable. The actor KB is initialized with resources like Wikidata (Vrandečić and Krötzsch, 2014), but allows for manual extension to cover references to less known people. Similarly, some claims categories typically need to be refined or coarsened in the initial phase of annotating a new topic of debate. The dynamic nature of the KBs make it necessary for the environment to provide functionality for *data and error analysis* and for *versioning of KBs and data*.

Comparison to other annotation environments. In the NLP community, BRAT[2] and WebAnno[3] are the most prominent tools for web-based annotation. Both tools focus on annotation of linguistic items (tagging, named entities). They are not designed for the annotation of complete document collections, nor do they provide interfaces to integrate complex and dynamic codebooks. The same holds if we consider more general frameworks like UIMA[4] or

[2] https://brat.nlplab.org/
[3] https://webanno.github.io/webanno/
[4] https://uima.apache.org/index.html

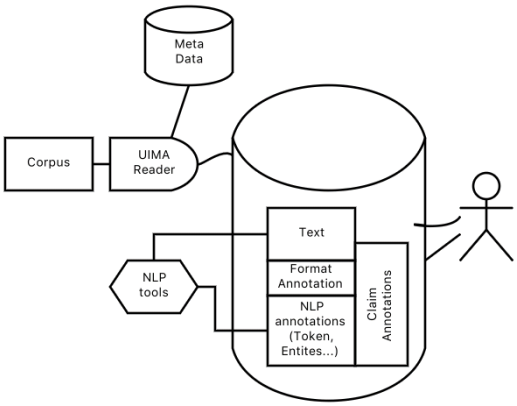

Figure 4: Technical structure of environment

GATE[5]: they cannot be used directly for complex annotation tasks, but need to be adapted (as we did for UIMA, cf. Section 3).

Established tools for qualitative data analysis (QDA) in political science are MAXQDA, NVivo and Atlas.ti (Friese, 2019; Rädiker and Kuckartz, 2019). These applications allow researchers to comfortably annotate a wide variety of textual data. However, their unit of analysis is always the text and not the annotated text segment. As a result, they do not retain the relational aspects of annotations: A text segment in which actor A makes claim X and actor B makes claim Y becomes indistinguishable from an annotation in which B claims X and A claims Y. Another tool, the discourse network analyzer (DNA) by Leifeld (2009), was developed specifically for the purpose of Discourse Network Analysis and solves this issue by focusing on the concept/actor relation as the unit of analysis. However, this application offers only very basic support for multiple annotators and does not enable parallel annotation of text. All annotations are always visible to the entire team, and there is no functionality to compare and merge annotations.

3 Design and Implementation

Figure 4 shows the technical structure and the frameworks used for the environment which we have developed to realize the workflow and desiderata from Section 2. The web page https://mardy-spp.github.io/ contains a demo video, information concerning demo access, as well as a docker image for the annotation environment.

Document Preprocessing and Selection. Our environment builds on existing pipelines and web services using UIMA as data exchange formalism.

[5] https://gate.ac.uk/

The input documents (in this case newspaper articles provided by the publisher) are encoded in a proprietary XML format. We developed a reader which parses and transforms the source articles into the UIMA representation defined by our type systems. That involves three tasks: i) identifying all relevant textual parts of the articles (e.g., embedded image captions have to be removed); ii) extracting meta data (e.g., date, author, title); iii) transforming format information (e.g., headings, footnotes). Afterwards, the text is passed through a generic pre-processing pipeline which calls several CLARIN webservices, namely tokenization and sentence boundary detection[6], POS tagging[7], and named entity recognition[8]. The analyzed documents are stored in an UIMA repository and an Elastic stack (https://www.elastic.co/).

Selecting a good sample has an high impact on the annotation process. MARDY starts off by using the Elastic stack's built-in keyword-based search to select relevant documents. This approach is complemented by a document classifier which can be trained as soon as some documents have been confirmed for annotation and some others rejected.

Annotation Frontend. Our core system is based on a client-server architecture. The Java server component interacts with the Elastic stack and the UIMA repository to store and retrieve annotated documents. The front-end is implemented with AngularJS (https://angularjs.org/).

The GUI for article view and annotation is implemented in Annotatorjs (http://annotatorjs.org/), a framework which enables text span annotation. Once the annotator has selected the relevant textual span, an input widget (Figure 5) displays claim categories (as defined by the annotation schema) as well as a suggestion list of potential actors, dates, and polarity (see below for details). This reduces annotation in most cases to selection among a small number of options and is crucial to increase annotation speed. Annotations are stored in a JSON-based format which is integrated by our back-end into the UIMA standoff format.

The tool supports user management for article assignment, as well as for checking the progress

[6] http://hdl.handle.net/11858/00-247C-0000-0007-3736-B

[7] http://hdl.handle.net/11858/00-247C-0000-0022-D906-1

[8] http://hdl.handle.net/11858/00-247C-0000-0022-DDA1-3

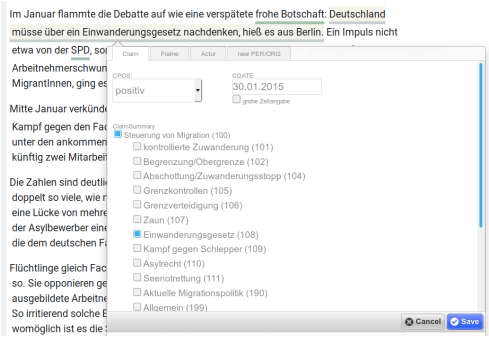

Figure 5: Annotation interface

of the annotation (documents are displayed as in progress, still awaiting annotation, or finished).

Merging. Figure 6 shows the view which enables experts to merge annotations into the gold standard. The experts decide for each row (same background color) if the detected text snippet is a claim and, if it is, then it is copied to the right column which shows the gold annotations. Each gold annotation can be further modified to adjust categories, actors and polarity of the claim. Usually, the annotations listed on the left are by human annotators, which are identified by IDs shown in square brackets ([7] in Figure 6). MARDY allows the users to integrate the predictions of a machine learning classifier which are then displayed in the merging view, marked as [AI] and with a lighter background color. Figure 6 illustrates an evaluation scenario for the [AI] pseudo-annotator. In the first row, [AI] has spotted a claim that the manual annotator had missed; moreover, it has identified the correct macro-category (700): the expert just needs to approve the claim and assign a finer-grained category. The [AI] annotator is, however, not perfect: the claim in the second row has been correctly identified only by the human annotator. In the third row, [AI] produced a false positive, probably because the candidate sentence contains the keyword *Lösung* (*solution*), which is a strong lexical cue for claims. In this case, though, only the need for a solution is stated, with no proposed action, leading the expert to reject the annotation.

Evaluation and Codebook Update. MARDY offers the possibility to evaluate annotation performance, thus providing valuable feedback to both annotators and specialists. The evaluation view reports performance (TP, FP, FN, precision, recall, and F1) aggregated per annotator and per category. The former is useful for training purposes, the latter particularly informative for the incremental re-

Figure 6: Left: user annotation or AI predictions; Right: approved annotations

finement of the codebook in the light of corpus evidence: categories that are often confounded are either ill-defined, or they stand in a systematic relationship (equivalence, subsumption, negation). Both cases may indicate the need for a redefinition of such categories. Versioning of data and codebook is available to later reconstruct how the annotation schema evolved during the project.

Actor Annotation and Update. In order to increase the speed and consistency of actor annotation, MARDY presents a set of plausible actors. This is achieved by attempting to link each named entity of type person or organization to a Wikidata entry. These entries provide canonical names (*Angela Merkel*) from which we derive variants (*A. Merkel, Frau Merkel*). We also exploit encyclopedic information to identify phrases that likely refer to these actors (e.g., *Kanzlerin (chancellor), CDU-Vorsitzende (CDU chairwoman)*). Given that Wikidata does not provide exhaustive coverage, the actor KB is extended whenever an annotator identifies an unknown actor.

4 Usability Study

In this section, we show how MARDY has been employed in a political science study targeting one of the major topics of German politics of 2015: the domestic debate on (im)migration policy. So far, we annotated 423 newspaper articles from *TAZ* (`http://www.taz.de/`), with a total of 982 claims (Padó et al., 2019).

The first step is **document selection**. The whole TAZ corpus contains more than 140.000 articles for the year 2015. The keyword-based search was used with a high recall objective in mind and resulted in 3112 articles. From the 423 annotated documents from this sample, approximately 58% was found to be off-topic. In the future, the second ML-based selection stage (cf. Section 3) should improve precision and thus reduce annotator load (fewer irrelevant articles to go through).

For **annotation**, each article was assigned to two annotators. 20 articles were used for annotator training and therefore assigned to all annotators. Note that the order in which the articles were shown to the annotators was randomized and thus not chronological. Multiple annotators could work simultaneously on the same article, while being monitored and compared by the researchers in real time. These features simplify and improve instructions and supervision, thus being more time-efficient than traditional approaches. Additionally, the fact that actor and claim categories are presented directly to annotators in the tool without laborious switching between applications led to a quicker and more comfortable annotation experience. Particularly helpful for high recall was the **integration of ML-based pseudo-annotators**: the identification of claims is a hard task for human coders, so that even the merging of several independent human annotations does not guarantee full coverage. We found that relatively simple neural sequence classifiers were already good enough to substantially boost the recall during the merging phase (Padó et al., 2019).

After several training rounds, we proceeded to a first **evaluation**. The quality of the human annotations was assessed in comparison to the gold standard. We computed annotation reliability for two parts of the annotation: claim detection and claim classification. For claim detection, we use Cohen's Kappa: For each sentence we compare whether the two annotators classified the sentence as part of a claim or not. We obtained a Kappa value of 0.58. Since claim classification is a multi-label task, Kappa cannot be used. We therefore computed Macro-F1 for all nine categories, obtaining an average F1 score of 63.5%.

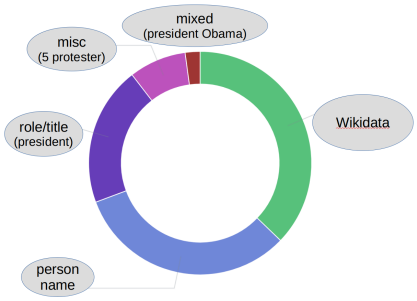

Figure 7: Actor Linkage: Distribution of categories.

Our annotation workflow greatly benefited from **integration with the claim category and actor KBs**. Indeed, the codebook was subject to many changes throughout the annotation process, as expected: Several iterations of reading articles and applying annotations were needed before convergence on a final version. Regarding the actor KB, Figure 7 illustrates the distribution of the actor categories annotated in the TAZ documents. The green portion are those person names that occur directly in Wikidata (roughly one third). Another third is composed of roles and titles (*Kanzlerin* (chancellor)) and mixed mentions (*Kanzlerin Merkel*), which may or may not be covered in Wikidata. The last third (light blue) are person names lacking completely in Wikidata. This underscores the need to easily and seamlessly extend the actor KB.

5 Outlook

We introduced an annotation environment whose features have been shaped by the goal of annotating German political debates to support discourse network analysis. Our tool is, however, highly flexible. First of all, its use is not restricted to German and it can be linked to any NLP pipeline. Moreover, it is not restricted to a specific document type: it can be employed, for example, for annotation targeting fewer layers (e.g., just claims and polarity, like in forum discussions). Finally, the framework can be (and has been, in the CRETA project (Blessing et al., 2017)) adapted to a broader range of text analysis contexts: e.g., it can be employed in literary studies to identify textual spans associated to characters, or having specific stylistic features. Finally, from a NLP perspective, our tool is a straightforward evaluation platform for classification models.

Acknowledgments

We acknowledge funding by Deutsche Forschungsgemeinschaft (DFG) through MARDY (Modeling Argumentation Dynamics) within SPP RATIO and by Bundesministerium für Bildung und Forschung (BMBF) through the Center for Reflected Text Analytics (CRETA).

References

André Blessing, Nora Echelmeyer, Markus John, and Nils Reiter. 2017. An end-to-end environment for research question-driven entity extraction and network analysis. In *Proc. of LaTeCH*, pages 57–67.

André Blessing, Fritz Kliche, Ulrich Heid, Cathleen Kantner, and Jonas Kuhn. 2015. Computerlinguistische Werkzeuge zur Erschließung und Exploration großer Textsammlungen aus der Perspektive fachspezifischer Theorien. In C. Baum and T. Stäcker, editors, *Grenzen und Möglichkeiten der Digital Humanities (= Sonderband der ZfdG 1)*.

Susanne Friese. 2019. *Qualitative data analysis with ATLAS*. SAGE Publications Limited.

Maarten A Hajer. 1993. Discourse Coalitions and the Institutionalization of Practice: The Case of Acid Rain in Britain. In *The Argumentative Turn in Policy Analysis and Planning*, pages 43–76. Duke University Press.

Sebastian Haunss and Jeanette Hofmann. 2015. Entstehung von Politikfeldern – Bedingungen einer Anomalie. *dms – der moderne staat*, 8(1):29–49.

Ruud Koopmans and Paul Statham. 1999. Political Claims Analysis: Integrating Protest Event And Political Discourse Approaches. *Mobilization*, 4(2):203–221.

Jonas Kuhn. to appear. Computational text analysis within the humanities: How to combine working practices from the contributing fields? *Language Resources and Evaluation*.

Philip Leifeld. 2009. Die Untersuchung von Diskursnetzwerken mit dem Discourse Network Analyzer (DNA). In *Politiknetzwerke. Modelle, Anwendungen und Visualisierungen*, pages 391–404. VS Verlag für Sozialwissenschaften, Opladen.

Philip Leifeld. 2016. Discourse Network Analysis: Policy Debates as Dynamic Networks. In *The Oxford Handbook of Political Networks*. Oxford University Press.

Sebastian Padó, André Blessing, Nico Blokker, Erenay Dayanik, Sebastian Haunss, and Jonas Kuhn. 2019. Who sides with whom? towards computational construction of discourse networks for political debates. In *Proceedings of ACL*, Florence, Italy.

Stefan Rädiker and Udo Kuckartz. 2019. *Analyse qualitativer Daten mit MAXQDA: Text, Audio und Video*. VS Verlag.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.