CodeAnno: Extending WebAnno with Hierarchical Document Level Annotation and Automation

Florian Schneider[†], **Seid Muhie Yimam**[†], **Fynn Petersen-Frey**[†]
Gerret von Nordheim[‡], **Katharina Kleinen-von Königslöw**[‡], **Chris Biemann**[†]
[†] Language Technology Group, Department of Informatics, Universität Hamburg, Germany
[‡] Institute of Journalism and Communication Studies, Universität Hamburg, Germany
{florian.schneider-1, seid.muhie.yimam, fynn.petersen-frey,
gerret.vonnordheim, katharina.kleinen, chris.biemann}@uni-hamburg.de

Abstract

WebAnno is one of the most popular annotation tools that supports generic annotation types and distributive annotation with multiple user roles. However, WebAnno focuses on annotating span-level mentions and relations among them, making document-level annotation complicated. When it comes to the annotation and analysis of social science materials, it usually involves the creation of codes to categorize a given document. The codes, which are known as **codebooks**, are typically hierarchical, which enables to code the document either with a general category or more fine-grained subcategories. **CodeAnno** is forked from WebAnno and designed to solve the coding problems faced by many social science researchers with the following main functionalities. 1) Creation of hierarchical codebooks, with functionality to move and sort categories in the hierarchy 2) an interactive UI for codebook annotation 3) import and export of annotations in CSV format, hence being compatible with existing annotations conducted using spreadsheet applications 4) integration of an external automation component to facilitate coding using machine learning 5) project templating that allows duplicating a project structure without copying the actual documents. We present different use-cases to demonstrate the capability of CodeAnno.

1 Introduction

When WebAnno was initiated, the main purpose was to address the missing functionalities from the annotation tools, particularly adding the distributive, web-based, generic, and customizable annotation layers behaviors (Yimam et al., 2013). Since then, WebAnno has been improved a lot, including the support for semantic annotation (Eckart de Castilho et al., 2016), support for automation functionality (Yimam et al., 2014), and adapted to the integrating knowledge-supported search 'INCEpTION' platform (Boullosa et al., 2018). The **Span** annotation type, which allows annotating tokens, sub-tokens, and phrases as well as the **Relation** type that connects two-span annotation types with relation value in WebAnno covers most of the linguistic and machine learning annotation tasks. The **Document** level annotation type is one of the most sought annotation types needed in WebAnno, which is also missed functionality from several annotation tools (Neves and Ševa, 2019). One of the workarounds in WebAnno was the **Zero-width spans**, which is not attached to any span but only to a specific sentence in a document. However, this still could not fully support document-level annotation, where the task is to label or classify a document based on a predefined category. The document-level annotation is particularly important for annotations in social science and digital humanities research. A typical example is document coding, where a researcher in communication science prepares a list of codes or tags known as **codebooks**. Journalists and media analysts prepare the codes for a given codebook but the annotation of content involves a lot of distraction, such as 1) preparing the code in a spreadsheet, 2) coding or annotating an article or media which is not linked with the codebook in the spreadsheet, 3) analyzing and interpreting the results from the coding.

CodeAnno is a fork and an extension to WebAnno, which enables coding based on hierarchical and fully customizable document-level annotations within WebAnno. CodeAnno facilitates the creation of codebooks and coding using the existing functionality from WebAnno such as curation, coding agreement, distributed annotation (annotating a single document by multiple coders or users), web-based annotation, and so on. As CodeAnno supports all existing functionalities of WebAnno, for example, span and relation annotation, one can easily connect the coding with the built-in WebAnno annotation support that can help to draw some conclusions.

The following are the main contributions of this

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics
System Demonstrations, pages 11–17

work: 1) Support codebook creation and annotation (hierarchical and document-level annotation), 2) Importing and exporting annotations in comma-separated format, hence compatible with existing annotations conducted using spreadsheet applications, 3) Integration of an external automation component to facilitate media coding using machine learning models, 4) Project templating that allows duplicating a project structure without copying the actual documents.

In the remainder of this paper, we will first discuss the related works of document-level annotation in Section 2. In Section 3 and 4, the main components of CodeAnno and the automation functionalities are presented. Some use-cases showing the codebook annotation and automation components are presented in Section 5 while Section 6 highlighted the main contribution of this work.

2 Related Work

Most of the annotation tools support linguistic annotations such as POS tagging, dependency relation, named entity recognition and so on (Yimam et al., 2013; Neves and Ševa, 2019). Brat (Stenetorp et al., 2012), WebAnno (Yimam et al., 2013), Docanno (Nakayama et al., 2018), and LightTag (Perry, 2021) are some popular annotation tools. The survey by Neves and Ševa (2019) indicated that only 5 annotation tools out 78 have some kind of document level annotation supports. ActiveAnno (Wiechmann et al., 2021) is one of the latest annotation tools that support annotation and automation for document-level annotation tasks. ActiveAnno focuses on five central design goals, namely, efficiently creating annotations of high quality, supporting a broad range of use cases, responsive web application, open-source and APIs integration. While ActiveAnno has great support for document-level annotation, it does not natively support hierarchical annotation using codebooks. It also lacks the extensive generic annotation support from WebAnno, which can be used to link the document level annotation to span level annotations.

INCEpTION is another annotation platform, which integrates all the functionalities of WebAnno and focuses on corpus creation, annotation, and knowledge management. While INCEpTION extends the functionality of WebAnno to more advanced functionality, including the external recommender system and knowledge supported search, CodeAnno particularly focuses on the creation of document-level annotation with the hierarchical codebooks. Furthermore, CodeAnno uniquely supports importing/exporting documents in a spreadsheet format, which is the de facto annotation format in social science document coding. Project templating, external automation service, and document-level codebook agreements are additional specific functionalities in CodeAnno.

In social science, coding refers to the process of assigning descriptive or inferential labels to documents or parts of the document, that could help in developing a new concept or theory (Chen et al., 2018). Researchers usually take a sample of the data and apply manual coding. As the size of data increases, manually coding the entire dataset in detail is not feasible for social science researchers. Building a representative machine learning model is also becoming challenging (Chen et al., 2018). To address this issue, we have integrated an external automation component that could provide an initial suggestion of codes. The process is iterative and adaptive, where small portions of the document are labeled to train the external predictive model.

3 CodeAnno Features

CodeAnno extends WebAnno by introducing new functionalities in WebAnno. In WebAnno, "layers" are used for span, chain, or relation annotations on token, sentence, or paragraph level. CodeAnno introduces document-level annotations, referred to as Codebooks. The new functionality is seamlessly integrated into WebAnno, i.e., the backend is also based on Apache UIMA[1] while the frontend is based on Apache Wicket[2]. Besides the support for Codebooks, another main contribution of CodeAnno is the integration of an external machine-learning-based service to automate Codebook annotations.

Codebooks are hierarchical document-level annotations consisting of a name, optional parent and child Codebooks, and a list of Codebook Tags. The resulting Codebook structure is therefore best represented as a tree. Codebook Tags are the value of the respective Codebook annotation and can be predefined by the project organization or, if allowed, created ad-hoc by the annotator. Further, Codebooks and Codebook Tags can have a textual description so that the annotators know what they stand for and when or how to use them to ensure high coding quality.

[1] https://uima.apache.org/
[2] https://wicket.apache.org/

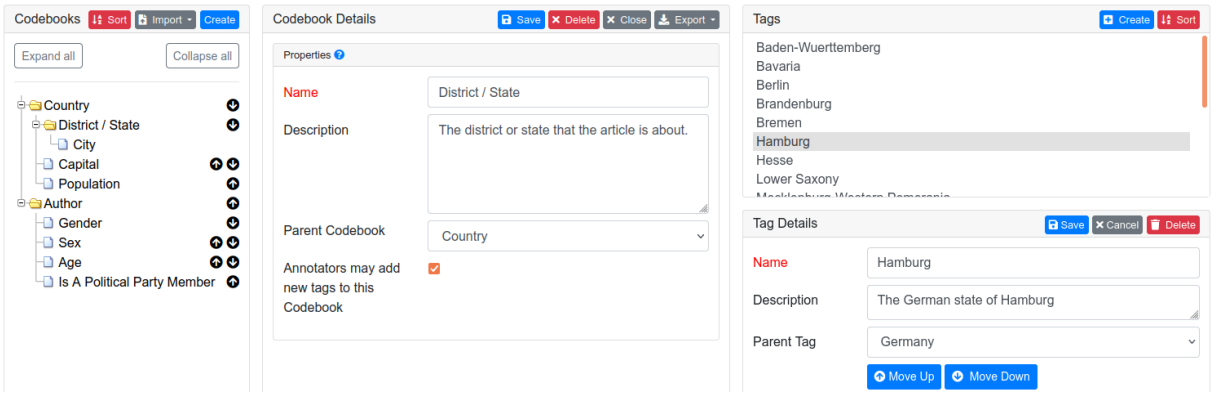

Figure 1: The Codebook settings UI in project settings to create and manage Codebooks and Codebook Tags.

Before the actual coding process starts, the project manager has to set up the Codebook structure that the coders will use to annotate the documents. This is done in the project settings with an intuitive and easy-to-use Codebook Editor shown in Figure 1. In the left part of the figure, a Codebook structure is displayed as a directory tree. The "Expand all" and "Collapse all" buttons are used to show or hide nodes in the Codebook tree. Codebooks can be moved up or down on the same level with the arrow buttons on the right. During creating a new Codebook, the project manager enters the Codebook properties in the middle part of Figure 1. To create a Codebook hierarchy, the parent Codebook can be selected from the drop-down list. If the Codebook is a root, none is selected. It is also possible to change the parent Codebook afterwards to update the tree structure. Further, if annotators should be allowed to enter custom Codebook Tags, the respective checkbox must be selected. Otherwise, only predefined Codebook Tags shown on the right side of Figure 1 can be chosen during the coding process.

In the "Tag Details" panel, that appears when a new Codebook Tag is created, its name, its description, and its parent tag will be recorded. The sort, move-up, and move-down buttons can be used to sort the list of Codebook Tags alphabetically or manually. When deleting a Codebook that is not a leaf node in the structure, all Codebooks in the subtree with their Codebook Tags also get deleted. Moreover, it is possible to import or export Codebook structures in human-readable JSON format. This is especially helpful when multiple coding projects with CodeAnno are planned.

3.1 Codebook Annotation

The WebAnno annotation interface is extended by a Codebook Editor panel that can be opened from the left sidebar to annotate documents with Codebooks. As shown in Figure 2, the classical WebAnno Layer Annotation editor can also be shown or hidden to enable span-level annotation at the same time. An example that shows the Codebook Editor with the configured Codebook structure from Figure 1 is shown in the left side of Figure 2. Each node in the tree represents a Codebook with its name in bold. The input for a Codebook combines a text input field and a dropdown selection. If enabled, annotators can enter any value to create a custom Codebook Tag or select a predefined tags from the dropdown list.

3.2 Codebook Curation and Agreement

Often, to ensure high-quality annotations, it is necessary that multiple users annotate the same document. However, since it can be subjective to decide which Codebook Tag fits best for the document, it can easily result in different Codebook annotations. This gives rise to the Codebook Curation feature, where project managers or curators can see how each user annotated the selected document in a separate UI. Based on this information, the curator chooses the "final" or correct Codebook Tag for each Codebook in the structure. An example of this UI is shown in Figure 3 – for space reasons, we cropped the image so that the document viewer, which would be on the right, is not shown. The Codebook structure is shown in a tree view similar to the annotation UI. The Codebook Tags of each user are shown in a list per Codebook. If all users agree on the same tag, the name of the respective Codebook is highlighted in green. Otherwise, it is highlighted in red. Below the list, the curation user can choose the correct Codebook Tag from a dropdown selection, where every tag is contained. If all users agree, the respective tag is preselected

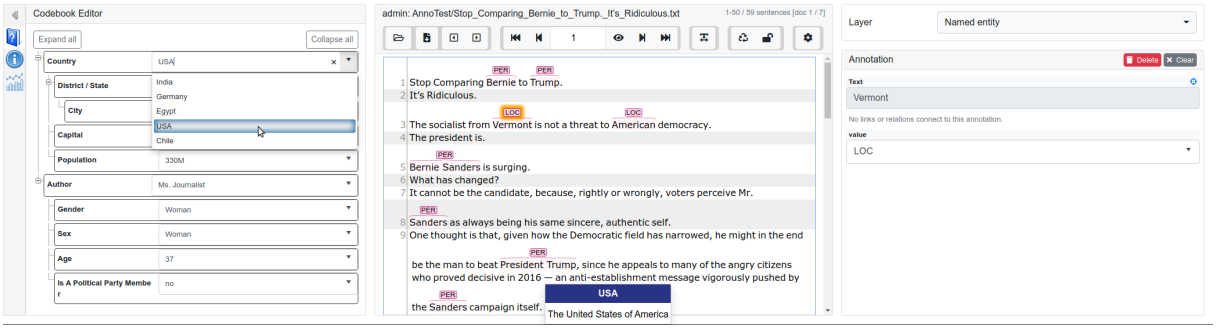

Figure 2: The CodaAnno annotation UI with the Codebook Editor for document-level annotations on the left, the document viewer in the middle, and the Layer Editor for span annotations on the right.

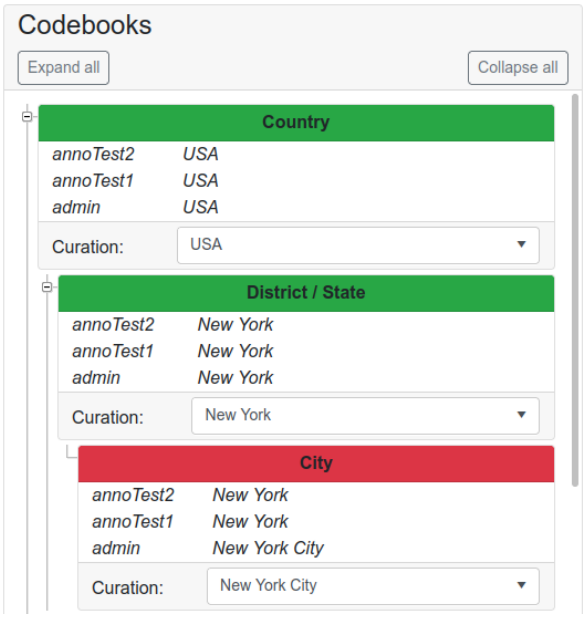

Figure 3: The CodaAnno Curation UI showing the Codebook annotations of different users. The curation user can select the "correct" Codebook from the dropdown menu at the bottom of each Codebook node.

to speed up the curation process.

With the Codebook Agreement feature, it is possible to calculate common inter-rater and intra-rater agreement measures per Codebook and project. For this purpose, the project manager first selects the respective project and Codebook. With a dropdown to select the measure, she can choose between Cohen's Kappa, Fleiss' Kappa, Krippendrof's Alpha, and Krippendorfs's Kappa. When the calculation has finished, the results are presented in a table like with layer annotation agreement in WebAnno.

3.3 Codebook Import and Export

CodeAnno features multiple new import and export functionalities. They are separated into import/export of the Codebook structure and the coded data belonging to each document. This separation allows reusing the Codebook structure by easily transferring it to a blank project without copying any document data. Since the Codebook is structured hierarchically, CodeBook uses the JSON format for import/export due to its convenient nesting of objects. As such, in the Codebook settings, it allows to import/export the whole CodeBook or only subtrees of it, thereby providing great flexibility to reuse only parts of CodeBook. The separated import/export in JSON of the Codebook (subtree) structure further has the advantage of being both human-readable and easy to manipulate in any text editor or even programmatically. Thus, changing many entries at once during construction of the Codebook, e.g. mass renaming, becomes very efficient. The import/export of Codebook data belonging to a document is possible in multiple ways. A document can be opened in CodeAnno and its Codebook annotations can be exported. In addition, the whole project can be exported, containing all documents and the Codebook structure. Importing documents previously exported is also possible through the Codebook settings. Thus, documents with Codebook annotations can be exported and later imported to move the documents to another project or make a backup.

4 CodeBook Automation (CBA)

Another major component of CodeAnno is the Codebook Automation (CBA) extension. As the name suggests, this functionality enables automatic annotation of Codebooks leveraging state-of-the-art machine learning technology to train and evaluate generic, user-specific classification models. The component consists of three parts: The CBA backend, the CBA WebApp, and the CBA integration in CodeAnno. An overview of the different parts is shown in Figure 4. The backend, which contains all logic and manages data and models, is accessible via a REST API. The CBA WebApp is a web-based user interface that consumes this API so

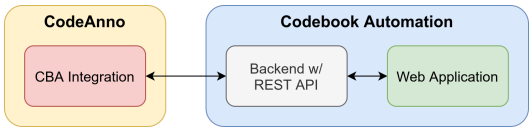

Figure 4: Overview of the Codebook Automation components. The communication between the components indicated by the arrows happens via HTTP.

that users can conveniently manage datasets, and create, train and test user-specific Codebook classifiers. Both parts are separated from CodeAnno and run independently. The third part is the CBA integration in CodeAnno to apply trained classifiers to annotate, i.e., predict documents with Codebooks automatically.

4.1 Backend Architecture of CBA

The CBA backend holds all logic to manage datasets and classifiers and exposes its functionality via a REST API. It is implemented using modern Python libraries and frameworks like FastAPI[3], pandas[4], Redis[5], and TensorFlow[6].

The process to train a classifier is schematically depicted in Figure 5. The dataset, uploaded by a

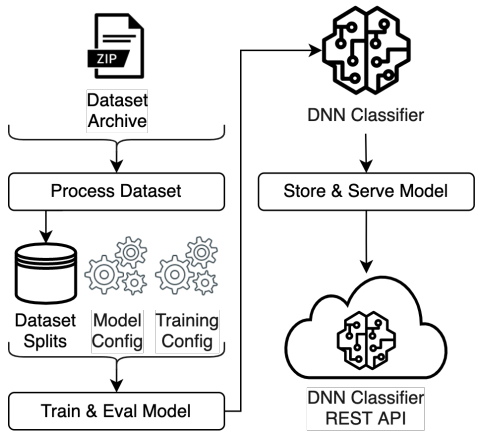

Figure 5: Schematic overview of the process to train a classifier in the Codebook Automation backend.

user, is a ZIP archive containing two CSV files for the training set and the evaluation set. Both files have to consist of two columns: The "text" column contains arbitrarily long text representing the document, and the "label" column contains the class or label for the sample. Further, the user must provide configurations defining the model architecture and training process. A model's architecture configuration contains the number of hidden layers and their respective hidden units, the activation function, the drop-out percentage, and the optimizer used for training. Further, the specification contains an URL pointing to a text-embedding model available on TensorFlow Hub[7]. A typical choice for this is, e.g., a Universal Sentence Encoder (Cer et al., 2018). The training process configuration specifies the batch size, the number of maximum training and evaluation steps, the optimizer, and if early stopping is activated or not. Once the model training is completed, it is persisted in the TensorFlow SavedModel format, supporting efficient model serving.

4.2 CBA WebApp and CodeAnno Integration

The CBA WebApp is the user interface for uploading or managing datasets and training or managing classifiers. The application consumes the REST API of the CBA backend and is implemented using NuxtJS[8].

Codebook Automation is also neatly integrated into CodeAnno so that human coders can leverage CBA classifiers to accelerate the Codebook annotation process. To use this functionality, a project administrator can start a bulk prediction process, in which all project documents are sent to the CBA backend and classified by the specified model. Another way is that annotators can send their currently processing document to the backend and let the default model set by an administrator classify the document.

The predictions are available in the Codebook Annotation UI when the process has been completed. Users can then easily accept or reject the suggestions from the CBA classifiers.

Details about the CBA WebApp and the CBA CodeAnno integration can be found on our GitHub[9][10] pages.

5 Case Studies

5.1 The NEPOCS Codebook

NEPOCS stands for Network of European Political Communication Scholars[11] established by a group of social science researchers who are working on political communication research. Each member is an expert on a particular European country, and the mission is to further internationally comparative political communication research. The work

[3] https://fastapi.tiangolo.com/
[4] https://pandas.pydata.org/
[5] https://redis.io
[6] https://www.tensorflow.org
[7] https://tfhub.dev
[8] https://nuxtjs.org/
[9] https://github.com/uhh-lt/codebook_automation
[10] https://github.com/uhh-lt/codeanno
[11] https://nepocs.eu/about-nepocs/

by Hopmann et al. (2017) discussed the methods and approaches they have used in coding around 7,500 news items gathered from 16 different European countries. The news items were collected considering different criteria such as consisting of comparable countries (established Western democracies) and media and political system level. As a source of the news item, televised news, newspapers, and online news outlets were considered. As part of the goal to the "2012 Journalism special issue", Hopmann et al. (2017) build a codebook as a contribution to *increasing standardization of how key concepts are conceptualized*. In collaboration with political communication science experts at Universität Hamburg, we have built standard codebook of NEPOCS, as shown in Figure 6 that can be released as part of CodeAnno for further use.

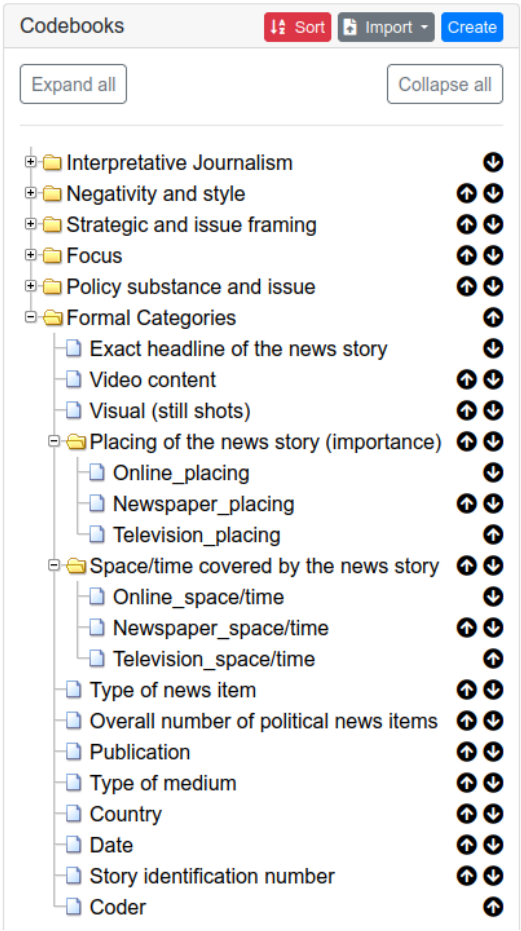

Figure 6: Parts of NEPOCS codebook designed with social science researchers.

5.2 CLICCS project

The Cluster of Excellence "Climate, Climatic Change, and Society" (CLICCS[12]) is following the overarching question: "Which climate futures are possible and which are plausible?" In the B1 sub-project researchers from journalism, media, and communication studies work on the question: "how do journalists frame climate futures?" For a quantitative study of climate future frames across countries, they have manually annotated hundreds of news articles in CodeAnno. Texts have been sampled from four countries (Germany, the United States, South Afrika and India) in German for Germany and English for the other three countries.

The researcher developed a sophisticated, hierarchical Codebook for the task based on a selected sample of documents. Coded are formal aspects like the type of text, author type and topic with a fixed number of categories. For the classification of future scenarios, they use hierarchically structured Codebooks for e.g. type of scenario, timeframe, geographic scope, plausibility, actors etc. In another sub-tree of the Codebook, the causes including attribution of blame and measures are structured into seven nested Codebooks.

In total, the whole Codebook comprises 26 aspects, each either coded as a Boolean or selection from multiple classes. To perform the annotation on the large sample, three coders have been trained on the developed Codebook. The CSV export of all coded documents has been used frequently to perform custom analysis on the annotations.

6 Conclusion

In this paper, we discussed main functionalities of CodeAnno, an extension of WebAnno that supports coding of social science documents using hierarchical Codebooks. Social science researchers usually employ traditional annotation tools, for example spreadsheet applications to code and analyze a text. This annotation process is cumbersome and hinders the development of predictive machine learning models. CodeAnno supports the creation of Codebooks, coding or annotating documents, and integration of custom machine learning models. Further, it eases the analysis of annotated documents and enables to import and export documents in different formats. Since CodeAnno intrinsically supports the annotation of entities and relations from WebAnno, we plan to integrate a functionality that provides automatic coding recommendations using the span-level entity and relation annotations. This facilitates the explainability of Codebook Annotations by linking back to the rationale in the document.

[12] https://www.cliccs.uni-hamburg.de/

Acknowledgement

This work is partly supported by the Cluster of Excellence CLICCS (EXC 2037), Universität Hamburg, funded by DFG, and the D-WISE project, funded by BMBF (grant ID 01UG2124).

References

Beto Boullosa, Richard Eckart de Castilho, Naveen Kumar, Jan-Christoph Klie, and Iryna Gurevych. 2018. Integrating Knowledge-Supported Search into the INCEpTION Annotation Platform. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 127–132, Brussels, Belgium.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium.

Nan-Chen Chen, Margaret Drouhard, Rafal Kocielnik, Jina Suh, and Cecilia R. Aragon. 2018. Using Machine Learning to Support Qualitative Coding in Social Science: Shifting the Focus to Ambiguity. *ACM Trans. Interact. Intell. Syst.*, 8(2).

Richard Eckart de Castilho, Éva Mújdricza-Maydt, Seid Muhie Yimam, Silvana Hartmann, Iryna Gurevych, Anette Frank, and Chris Biemann. 2016. A Web-based Tool for the Integrated Annotation of Semantic and Syntactic Structures. In *Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH)*, pages 76–84, Osaka, Japan.

David Nicolas Hopmann, Frank Esser, Claes de Vreese, Toril Aalberg, Peter Aelst, Rosa Berganza, Nicolas Hubé, Guido Legnante, Jörg Matthes, Stylianos Papathanassopoulos, Carsten Reinemann, Susana Salgado, Tamir Sheafer, James Stanyer, and Jesper Strömbäck. 2017. How We Did It: Approach and Methods. In *Comparing Political Journalism*, pages 10–21.

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text Annotation Tool for Human. Software available from https://github.com/doccano/doccano.

Mariana Neves and Jurica Ševa. 2019. An Extensive Review of Tools for Manual Annotation of Documents. *Briefings in Bioinformatics*, 22(1):146–163.

Tal Perry. 2021. LightTag: Text Annotation Platform. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 20–27, Online and Punta Cana, Dominican Republic.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a Web-based Tool for NLP-Assisted Text Annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France.

Max Wiechmann, Seid Muhie Yimam, and Chris Biemann. 2021. ActiveAnno: General-Purpose Document-Level Annotation Tool with Active Learning Integration. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 99–105, Online.

Seid Muhie Yimam, Chris Biemann, Richard Eckart de Castilho, and Iryna Gurevych. 2014. Automatic annotation suggestions and custom annotation layers in WebAnno. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 91–96, Baltimore, Maryland.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. Webanno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria.