Using Data Packages to Ship Annotated Corpora of Parliamentary Protocols: The *GermaParl* R Package

Andreas Blätte
University of Duisburg-Essen
andreas.blaette@uni-due.de

Abstract

This paper suggests to disseminate linguistically annotated and indexed versions of corpora of parliamentary debates as R data packages. The *GermaParl* Corpus of Parliamentary Protocols serves as an example to illustrate the advantages of this approach. Keeping data in data packages offers established mechanisms to version and document data, and for ensuring the reproducibility of the data. The package may include further annotation layers, and functionality to analytically exploit additional annotations. Finally, sharing packages via a CRAN-like repository is a user-friendly way to make data available.

Keywords: Corpus (Creation, Annotation, etc.), Document Classification, Digital Humanities

1. Introduction

Corpora of parliamentary protocols are available for an increasing number of countries, and are gaining a strong standing as language resources. There are many advantages of corpora of parliamentary protocols for substantial research and methodological innovation. A variety of disciplines may benefit from these data. The largely unproblematic licensing conditions for plenary proceedings provide a particularly important reason, why it is worth to invest time and energy into this family of corpora: Corpora of parliamentary protocols are sustainable because the raw data is in the public domain.

It is a common problem that copyright law and restrictive licenses impede research; consider for instance the difficulty to share corpora of newspaper articles and social media content for reproduction. Corpora of plenary protocols can be prepared, stored, analyzed, and shared without fear.

Corpora of parliamentary protocols will typically be large corpora. Because of their size, these corpora call for computational support for exploiting the analytical potential of the data. They can be used to develop and test all kinds of procedures and algorithms. But size and technical data formats give rise to another restriction that may make the data exclusionary. Even without legal restrictions, many researchers who might use corpora of parliamentary protocols very productively, but who do not have a strong technical background, will have great difficulties to handle corpora of considerable size.

Hosting the data centrally on a server and offering a web interface is a reasonable solution to serve users who work with standardized analytical procedures. CQPweb (Hardie, 2012), SketchEngine (Kilgarriff et al., 2004; Kilgarriff et al., 2014) and NoSketchEngine[1] are powerful, server-based systems that deservedly have attracted communities of users and developers. But these systems are too restrictive for users who intend to explore all kinds of algorithms, and that approach data with an experimental stance. Because of security reasons, administrators will usually want the technically more ambitious users to run their experiments on their own local machines. And of course, a server accessed by many users will often be not the best place to conduct computationally expensive experiments.

The solution suggested here is to share annotated and indexed corpora as data packages. The R data package *GermaParl* serves as the showcase. One line of code will be enough to install a linguistically annotated and indexed version of *GermaParl* on your personal machine (or server). Moreover, the data package includes a detailed documentation how the corpus has been prepared, a vignette how it can be used, and custom functionality to create thematically defined subcorpora for specific research purposes. The suggested approach can serve the aims offering open data, making the preparation of the data reproducible and transparent and minimizing barriers of entry. Therefore, offering corpora of parliamentary protocols as data packages is the suggestion of this paper.

2. Versions of *GermaParl*

The *GermaParl* Corpus – the naming of the corpus is inspired by the *DutchParl* corpus (Marx and Schuth, 2010) – is a corpus of parliamentary protocols of the German Bundestag. The data has been consolidated for the years 1996 to 2016. The corpus covers the period for which txt files are publicly available. The data is explained and documented elsewhere (Blätte, 2018; Blätte and Blessing, 2018). Here, we focus on the data types that can be offered and shared *after* the initial preparation of the data.

- *XML (TEI standardization):* The basic variant of the corpus is an XMLification of the raw data (txt and pdf documents) that follows the standards of the Text Encoding Initiative (TEI)[2]. The XML version of the corpus is available at a GitHub repository.[3] GitHub has many advantages such as an accessible display of data,

[1] `https://nlp.fi.muni.cz/trac/noske`.

[2] See `www.tei-c.org`.

[3] See `https://github.com/PolMine/GermaParlTEI`. Of course, git has been designed to support the development of code, but its logic makes it suited very well for versioning corpora. The strongest argument against keeping corpora at GitHub is the usual size limitation of repositories to 1 GB. Moving to GitLab, or a self-hosted GitLab server are viable alternatives to GitHub.

an option to download the data and a system for managing issues and to manage user feedback. However, the XML/TEI variant of the corpus is not the data format to actually work with. A set of further processing steps is necessary.

- *Linguistically annotated corpus:* In a manner that is common in Natural Language Processing (NLP), the XML/TEI variant of the corpus is passed through a pipeline for linguistic annotation, using standard tools. The NLP tool currently used is *Stanford Core NLP*[4] The output of *Stanford Core NLP* (JSON, in this case) is transformed into a verticalized data format that can be imported into the *Corpus Workbench* (CWB)[5].
- *CWB indexed version:* Indexing and query engines are crucial to make corpora a useful resource for research. The *Corpus Workbench* (CWB) is one of the older systems. But it is still a powerful, mature system that is well-maintained. Due to the flexibility and power of the *Corpus Query Processor* (CQP) and its uncompromising open source orientation, it keeps being a good choice as an indexing and query engine for scientific purposes. The linguistically and structurally annotated data that has been prepared in step two is imported into the CWB, achieving a considerable data compression and a data format that researchers can work with efficiently and productively at the same time.

Technically advanced and experienced users may work efficiently with the XML/TEI version of the corpus. Students from the social sciences, early stage researchers, and newcomers to the *eHumanities* or the *computational social sciences* will find it difficult to make productive use of that kind of XML. Thus, the best way to offer the data, is to grant access to the CWB indexed variant of the corpus. That could be done with a server-based system (such as CQPweb), thus restricting the flexibility of users. Another approach is to grant access to the (zipped or tarred) data at some kind of online storage, so that users can download the corpus and install it themselves locally. The approach suggested here is to wrap the data in an R data package that may include a fully developed documentation and specialized functionality. This can be hosted without a lot of effort at a CRAN-style repository. Conventional R mechanisms make downloading and installing the package minimally demanding, if the package is prepared appropriately. The next section explains how this is implemented in the *GermaParl* data package.

3. Hosting and installing the *GermaParl* R Data Package

The size of the *GermaParl* data package (almost 1 GB) exceeds the size limitations for packages by the *Comprehensive R Archive Network* (CRAN) by far.[6] Of course, from the perspective of users, it would be ideal to be able to install an established text resource from CRAN. But offering an alternative is neither difficult for those offering data, nor difficult to handle from the perspective of users. Because it is easy, it is fairly common to host and administer a 'private" CRAN-style repositories. The package *miniCRAN* supports setting up a private CRAN-like repository in an enterprise setting[7], the package *drat* offers functionality to insert packages into a repository, and is specialized on using (or abusing) GitHub Pages to host a CRAN-like repository[8].

All that is necessary to host a CRAN-like repository is to mimick the directory structure of CRAN, and to register any new package that you put in the repository, so that some metadata is written to a file called 'PACKAGES'. The aforementioned packages *miniCRAN* and *drat* support that, but it can also be done 'manually'. The directory structure of CRAN-like repositories is designed to host the source tarballs of packages as well as binary versions for macOS and Windows. As data packages with corpus data will usually not include code that needs compilation, it is just necessary to put the package into the *src* directory of the CRAN-like repository.

Residing in the folder src/contrib of the PolMine repository at `http://polmine.sowi.uni-due.de/packages`, the *GermaParl* package can be installed in an R session using the *install.packages* function.

```
install.packages(
  "GermaParl",
  repos = "http://polmine.sowi.uni-due.de/packages"
)
```

The package includes a configuration mechanism that will adjust paths in the so-called registry files describing the annotation of a CWB indexed corpus, so that they point correctly to the binary data files in the package. An even simpler installation mechanism is provided by the R package *polmineR*.[9]

```
library(polmineR) # load polmineR package
install.corpus("GermaParl") # install the corpus
```

This is all it takes to have the corpus installed, and to be ready to perform analyses. The following lines of code are examples for initial checks and basic analyses.

```
library(polmineR)
use("GermaParl") # activate GermaParl
corpus() # to see that the corpus is present
size("GERMAPARL") # get the size of GermaParl
kwic("GERMAPARL, query = "Corpus") # concordances
```

[4] See `https://stanfordnlp.github.io/CoreNLP/`. The command-line version of Stanford Core NLP does not work robustly with the structural XML annotation of the corpus. We iterate through the text nodes of the XML documents using an R package 'ctk' (*corpus toolkit*) that offers bindings for Stanford Core NLP, see `https://github.com/PolMine/ctk`.

[5] See `http://cwb.sourceforge.net/`

[6] According to the *CRAN Repository Policy*, packages should usually not exceed 5 MB, see `https://cran.r-project.org/web/packages/policies.html`.

[7] See https://CRAN.R-project.org/package=miniCRAN

[8] See `https://cran.r-project.org/package=drat`.

[9] The *polmineR* package can be installed from CRAN, see `https://CRAN.R-project.org/package=polmineR`. The most recent version of the package is available at GitHub (see `https://www.github.com/PolMine/polmineR`). See the README at CRAN for installation instructions.

A small example may demonstrate that users can indeed proceed quickly to substantial research once *polmineR* – a specialized package to work with CWB indexed corpora using R – and *GermaParl* are installed.[10] Let us assume that you are interested in the adjectives preceding mentions of the European Union ("Europäische Union" in the German corpus). Figure 1 displays a screenshot of an RStudio session (RStudio is the IDE we would recommend for working with R) to do this little exercise. After loading the *polmineR* package, the *GermaParl* corpus is activated by calling *use*. Then, the query (Q) is defined to find matches for the combination of an adjective and the EU. The result of calling the 'count'-method, a data.table is stored as variable Y, and the columns interesting for us (match, count and share) are viewed.

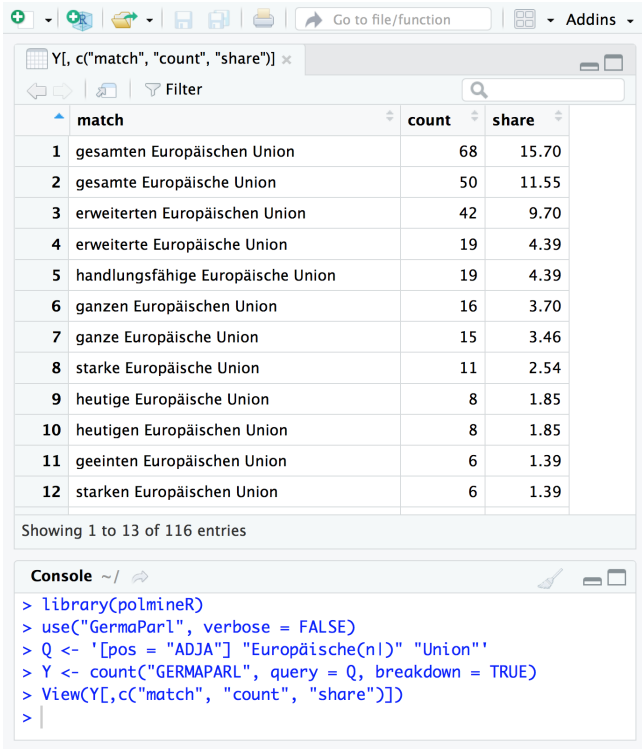

Figure 1: Adjectives before 'European Union' in *GermaParl* (RStudio session).

In a second, related example, we assume we are interested in references to an *enlarged* European Union, this is one of the occurring combinations we saw in the previous exercise. The CQP query we use is '[lemma = "erweitert"] ("EU" — "Europäische(n—)" "Union")'. So we use the linguistic annotation of the corpus and start with the lemma "erweitert" (enlarged). We then allow for the alternatives "European Union" ("Europäische Union"), and its abbreviation as "EU". Five commands lead to the barplot in figure 2: We load *polmineR*, activate *GermaParl*, define the query (variable Q), retrieve the hits using the hits-method, turn the object into a data.table, and produce the barplot. As you may learn from the progress bar in figure 2, it did not even take a second to find the matches for query Q in the 100 million token corpus. Speed is one the advantages of sharing an indexed and compressed corpus.

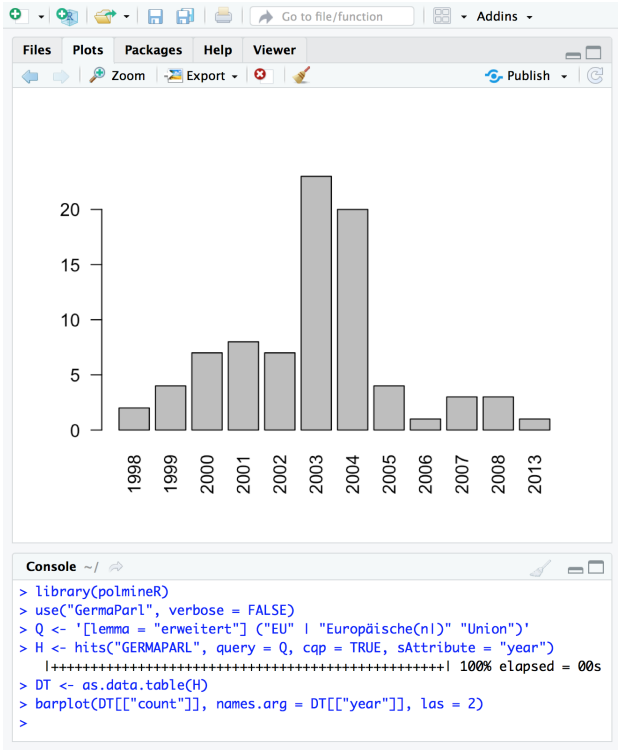

Figure 2: Enlarged 'European Union' (by year).

4. Accessing Data Documentation

One of the big advantages of wrapping corpora into a data package is that it can be accompanied by additional information and extensive documentation. Apart from the standard info file provided for by the CWB, a short document that provides some basic information on a corpus, the *GermaParl* package at the present stage includes two documents called 'vignettes' in the R jargon. The first vignette (with the title 'GermaParl') offers a general introduction to the corpus. It explains corpus preparation, and the structural and linguistic annotation that the corpus has. The second vignette ('MakingOfGermaParl') documents the workflow from XML/TEI documents to the CWB indexed version. These documents are easily accessible from R and can be inspected as follows.

```
Corpus$new("GERMAPARL")$showInfo() # show info file
browseVignettes(package = "GermaParl")
vignette("GermaParl", package = "GermaParl")
vignette("MakingOfGermaParl", package = "GermaParl")
```

From the point of view of reproducible research, the vignette 'MakingOfGermaParl' deserves particular attention. It is a complete documentation of the steps that take the XML corpus from the XML/TEI variant through the NLP pipe to the import into the CWB. It is generated from an *Rmarkdown* document. Following an advice of Hadley Wickham, it is part of the *GermaParl* git repository, i.e. it is stored in the folder data-raw (Wickham, 2015). Upon

[10] I should like to thank the anonymous reviewers for suggesting to include this kind of example.

executing the code in the document and generating the html document from the original Rmarkdown, the CWB indexed corpus is being prepared. Thus, the corpus data included in the package is perfectly reproducible.

Different versions of the 'Making Of'-document result in the different versions of the package. The raw data package is under version control, i.e. it is kept in a git repository. A technical difficulty is that the binary files of the indexed and compressed CWB corpus are large. Using Git LFS (for Large File Storage) is the appropriate solution for this scenario. Because of the size of this git repository, it is hosted at a private GitLab server of the PolMine Project. In a manner known from GitHub, GitLab offers an issue tracker that is very useful to manage issues, user feedback and feature requests. The changes that the corpus has seen are documented in an accessible manner with the file NEWS.md that is included in the package.[11].

5. How to Put Data in a Data Package

Hadley Wickham has written an excellent book on developing R packages (Wickham, 2015).[12] Developing an R data package is usually much easier than writing a package with complex code. To wrap a CWB indexed corpus into a package, we have chosen to put the binary files of the corpus into a package subdirectory inst/extdata/cwb/NAME-OF-THE-CORPUS, and the registry file describing the corpus into a directory inst/extdata/cwb/registry.

The only tricky part is to infuse a configuration mechanism into the package that will set correctly the paths pointing to the data directory with the binary files and the info file. Our best practice is to use an R template script called 'setpaths.R' in a subdirectory 'tools' that is called from the package configure script (for Linux and macOS), or configure.win script (on Windows) respectively.[13]

6. Features and Extra Functionality

Corpora of parliamentary protocols cover all kinds of issues across time. They are multi-purpose corpora. The multiple audiences these corpora target justify why it makes sense to invest resources in developing and maintaining these corpora. But large multi-purpose corpora engender the wish to create thematically defined subcorpora. Having the corpus wrapped into an R data package offers a convenient way to supplement the data with specialized functions to address issues such as this one.

A classification of speeches or agenda items based on a theoretically justified typology of issues and respective training data would be ideal. At the present stage, an additional annotation layer derived from optimized topic models is added to the *GermaParl* corpus. A standard topic modelling approach (Latent Dirichlet Allocation, LDA) has been used, taking as documents parliamentary speeches (not agenda items, for instance).[14] Following what is emerging as good practice, a set of topic models with varying numbers of topics has been trained. A set of parameters has been used to estimate the quality of the models, using the R package *ldatuning*.[15] According to rules of thumb to optimize the number of topics, around 250 topics is a good choice for a topic model for *GermaParl* (see figure 1).

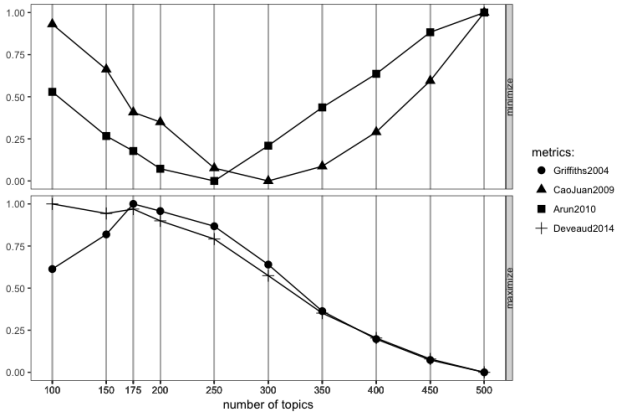

Figure 3: Visualisation of topicmodel optimization exercise.

The integer values of the five topics most prevalent in a speech according to the optimized topic model (k = 250) have been added as a structural attribute to the corpus. Once a user has has identified the topics relevant for his research based on the lists of tokens associated with topics, it is easy to formulate a query on the structural attributes to create a thematic subcorpus.[16]

Distributing a corpus in an R data package does not only offer a coherent way to distribute additional annotations, and the documentation of it. Functionality for additional analytical techniques specific to the data that is disseminated can be included in the package.

7. License and Attribution

The license chosen for the data package is a CLARIN PUB+BY+NC+SA license. The CLARIN licenses[17] are modeled on the Creative Commons licenses. The

[11] A nice side effect of the data package is that it is easy to generate a website from the different documentation files included in a package. The package *pkgdown* offers a handy mechanism to do this, see `https://github.com/r-lib/pkgdown`. Generating a website to promote and document the data is possible with minimal cost, for *GermaParl*, see `http://polmine.sowi.uni-due.de/docs/GermaParl/`

[12] See also `http://r-pkgs.had.co.nz/`.

[13] The script is included in the *ctk* package, see `https://github.com/PolMine/ctk/tree/master/inst/R`.

[14] Identifying speeches is not a trivial question, as parliamentary speeches are interrupted by interjections frequently. The *polmineR* package includes a function with a heuristic to identify speeches.

[15] See `https://CRAN.R-project.org/package=ldatuning`.

[16] This feature is currently only available in the development version of the GermaParl package, but it will be available in an upcoming release. A full documentation of the topic modelling exercise will be provided in an additional vignette. A future version of the *GermaParl* package will also include the functionality to generate classifications based on manually created training data. The data has already been prepared by trained coders in a CLARIN-funded project 'Plenarprotokolle als öffentliche Sprachressource der Demokratie' in 2015/16.

[17] See `https://www.clarin.eu/content/license-categories`.

CLARIN license is derived from the CC Attribution-NonCommercial-ShareAlike 3.0 Unported License.[18] Thus, the elements of the license mean:

- *PUB*: The language resource can be distributed publicly.
- *BY*: Attribution – You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- *NC*: NonCommercial – You may not use the material for commercial purposes.
- *SA*: ShareAlike – If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

Excluding a commercial usage of the corpus is a tactical issue. The restrictive licenses of most commercial publishing houses create considerable burdens for academics to use newspapers in their research. If corpora of newspaper articles are prepared as a corpus, they usually cannot be shared for reproduction and further research. So for the sake of (unfulfilled) reciprocity, it seems well justified if academics who invest energy in preparing corpora of parliamentary protocols preclude commercial users from using their data freely – until they can use commercial data freely, too.

Asking users to attribute the data they are using should be a standard in academic practice, but it is not necessarily how the digital world behaves. However, the R community has always had very strong ties to academia, and packages are meant to be quoted. Packages include statements of authorship and can include binding suggestions how they should be cited.[19] Using an R data package to disseminate corpus data implies a solution how authorship can be attributed. Offering data in a way that is quotable is an incentive to share data.

8. Perspectives

The *GermaParl* corpus of parliamentary protocols is made available as XML (TEI standardization) at a GitHub repository. Yet many users cannot be expected to be sufficiently acquainted with the NLP techniques necessary to turn the XML into a linguistically annotated corpus without hassle. This paper suggests that offering a linguistically annotated and indexed version of the corpus wrapped in an R data package may be a neat way to disseminate the data. It lowers barriers of entry for academic users that are not full-fledged computational linguists. What is more, the R data package *GermaParl* is intended to suggest a way how corpus preparation can be documented and made transparent. Making progress towards reproducible research is the ultimate aim.

To be sure, the suggestion is accompanied by the idea that *GermaParl* might become part of a larger family of corpora of plenary protocols that will be available as R data packages via CRAN-like repositories. For future research, it might indeed be very productive to have shared ideas how we maintain and share our corpora. The basic corpus preparation – attaining XML – is time-consuming and may absorb considerable attention. But there is a set of relevant questions and best practices beyond the XML stage of data preparation. How do we share the results of supervised, or unsupervised learning, for instance? There will be many further questions that the emerging availability of corpora of parliamentary protocols will engender. It will be good to have some common ideas how we maintain, document and share our data – for the sake of being somewhat more cumulative in our research endeavours.

9. Bibliographical References

Blätte, A. and Blessing, A. (2018). The GermaParl Corpus of Parliamentary Protocols. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2018, 7-12 May 2018, Miyazaki, Japan*.

Blätte, A. (2018). Zum Verwechseln ähnlich? Eine Klassifikationsanalyse parlamentarischen Diskursverhaltens auf Basis des PolMine-Plenarprotokollkorpus. In Joachim Behnke, et al., editors, *Computational Social Science. Die Analyse von Big Data*. Nomos, Baden-Baden.

Hardie, A. (2012). CQPweb combining power, flexibility and usability in a corpus analysis tool. *International Journal of Corpus Linguistics*, 17(3):380 – 409.

Kilgarriff, A., Rychl, P., Smr, P., and Tugwell, D. (2004). The Sketch Engine. *Information Technology*.

Kilgarriff, A., Baisa, V., Buta, J., Jakubek, M., Kovv, V., Michelfeit, J., Rychl, P., and Suchomel, V. (2014). The Sketch Engine: Ten years on. *Lexicography*, pages 7–36.

Marx, M. and Schuth, A. (2010). DutchParl. The parliamentary documents in dutch. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2010, 17-23 May 2010, Valletta, Malta*.

Wickham, H. (2015). *R packages. Organize, test, document and share your code*. O'Reilly, Sebastopol, CA.

[18] See `https://creativecommons.org/licenses/by-nc-sa/3.0/` for further explanations.

[19] On CITATION files see `https://cran.r-project.org/doc/manuals/r-release/R-exts.html`.