Technical Solutions for Reproducible Research

Alexander König
Eurac Research, Italy
Alexander.Koenig@eurac.edu

Egon W. Stemle
Eurac Research, Italy
Egon.Stemle@eurac.edu

Abstract

In recent years, the reproducibility of scientific research has more and more come into focus, both from external stakeholders (e.g. funders) and from within research communities themselves. Corpus linguistics and its methods, which are an integral component of many other disciplines working with language data, play a special role here – language corpora are often living objects: they are constantly being improved and revised, and at the same time, the tools for the automatic processing of human language are also regularly updated, both of which can lead to different results for the same processing steps. This article argues that modern software technologies such as version control and containerization can address both issues, namely make reproducible the process of software packaging, installation, and execution and, more importantly, the tracking of corpora throughout their life cycle, thereby making the changes to the raw data reproducible for many subsequent analyses.

1 Introduction

While reproducibility has always been one of the main pillars of scientific research, within the last ten years this has come even more into focus for the social sciences and humanities (SSH). Prominent cases of scientific fraud, for example the case of Diederik Stapel in the Netherlands (Levelt et al., 2012), have brought problems about the reproducibility of scientific research into focus. In this article, we discuss some possible techniques to handle this problem using standard tools from the realm of software development. We propose to use versioning software to ensure the persistence of data (see section 2) and containerisation to ensure the same for NLP tool-chains (see section 3). We will then briefly discuss a case study where this approach has been partly implemented (see section 4) and highlight some challenges that were encountered along the way (see section 5).

2 Ensuring persistence of data

After initially collecting the data of a (text) corpus it is common that the corpus keeps evolving while at the same time the first analyses are already being carried out. It is also likely that while working on the corpus and analysing the data, mistakes in the transcription or the annotation are discovered, which need to be corrected. And with a rich annotation scheme that is constantly being re-evaluated and refined this is usually all the more true.

While these kind of changes are unproblematic as long as the corpus is still in its ”building phase”, as soon as the first analyses have been made public, any change to the data will endanger the possibility of reproducing these analyses. Therefore, the researchers have to preserve a version of the corpus as it was when a specific analysis was made.

If the corpus in question is a text corpus (probably being stored in some kind of XML format like e.g. TEI[1]), an obvious solution to this problem is to use existing versioning tools like subversion[2] or

[1] https://tei-c.org/

[2] https://subversion.apache.org/

git[3] to keep track of all changes within the corpus. This is also possible for corpora that are not mainly text-based, for example, multimodal corpora. First, they often have a text-component in their annotations (which could have been done, for example in ELAN[4] or EXMARaLDa[5], both of which store their data in XML format) and second, the problem of storing large files in versioning tools is being addressed - and will likely eventually be solved. Using such an existing versioning software solution, all changes throughout the life cycle of a corpus can be tracked and through the use of code hosting platforms like Github[6] or GitLab[7] all changes can be made transparent to the research community as a whole.

But having the corpus available on such a code hosting platform, while having the advantage of being very transparent about all changes made to the data, might not be the ideal way of providing the data to other researchers. Therefore traditional data repositories like CLARIN Centres, META-SHARE[8], and zenodo[9] will still play a role in making the data available to the users and especially in providing findability (through participation in search interfaces like the VLO[10] or OLAC[11]) and issuing persistent identifiers to specific versions.

3 Methods and tools and their impact on reproducibility

In linguistic research – especially in the sub-fields of corpus linguistics and natural language processing – data is often processed with the use of quite intricate software tool-chains. Ranging from more simple tasks like tokenisation or lemmatisation to more complicated ones like fine-grained syntactic parsing. The unification of all the necessary tools for the automatic processing of human language into a unified processing framework is more the exception than the norm. This inevitably leads to a wide variety of individual solutions each with their own installation procedures, development life cycles with maintenance and update schedules, etc. (Wieling et al., 2018). Furthermore, linguistic models that are often at the heart of such tools are also subject to change, and this change need not necessarily be synchronized with the tool itself, spanning an even wider range of possible combinations (see e.g. (Nothman et al., 2018)).

This short overview already shows the difficulty for other researchers to exactly recreate a certain tool-chain to verify research results. It can only be ensured if the original researchers document their setup carefully, noting down exactly which version of a certain tool was used and how exactly the various tools were combined. An additional problem is that some software manufacturers do not make older versions of their products easily available, so even if the version is known it is not certain that it can be obtained when necessary. For this reason it has been discussed whether scientific software should be archived in research repositories alongside the data, but so far, while some CLARIN repositories do also host linguistic tools, little progress has been made in this regard.

The recent trend in software deployment and administration towards containerisation of services seems to us to be a promising solution to the aforementioned problems regarding the reproducibility of data processing in linguistic research.

Containerisation means that certain programs are installed not on a real computer or even a full-blown virtualised environment like a virtual machine, but instead in a very basic environment that leaves out everything that is not vital for the program in question to work. The idea is to minimize both the amount of memory and processor time needed for such a containerised service and also the possibility for unwanted side effects. With Docker[12] this way of packaging programs and services has been widely adopted within the last years and the additional possibility to orchestrate the deployment of such minimal

[3] https://git-scm.com/
[4] https://tla.mpi.nl/tools/tla-tools/elan/
[5] https://exmaralda.org/en/
[6] https://github.com/
[7] https://www.gitlab.com
[8] http://www.meta-share.org
[9] https://zenodo.org
[10] https://vlo.clarin.eu/
[11] http://search.language-archives.org
[12] https://www.docker.com/

containers using a platform like Kubernetes[13] makes using existing containers "off the shelf" quite easy, especially because a lot of the big infrastructure providers (e.g. Google[14], Microsoft[15] or Amazon[16]) offer ways to deploy containers on their infrastructure for a moderate price and there seems to be a trend to make this kind of deployment as easy as possible[17].

As a researcher, building the tool-chain for a new project can be done directly in Docker. There are already a variety of places where the resulting docker images (from which various container instances can be created) can be stored for re-use by others. For example, GitLab offers such a Docker image registry for free. GitLab is also a place where the data can be stored (see section 2), and both the data and the tool-chain used could thusly be stored in one place, making it much easier for researchers planning to recreate an experiment to get both in exactly the same versions that had been used originally. The wide availability of container hosting (see above) also means that it will be quite easy to simply take such a container with the whole tool-chain setup and use it to verify the results or look for something else in another set of data while ensuring that the same methodology is used as in the original research.

4 Case Study: The MERLIN Corpus

The Institute for Applied Linguistics (IAL) at Eurac Research is currently investigating how it can move towards such a setup for more reproducibility in research as outlined in the previous sections. One of the first corpora that was transformed into such a strictly versioned environment is the MERLIN corpus (Boyd et al., 2014). The corpus is completely available on a publicly reachable on-premise GitLab installation[18]. The repository is divided into multiple parts for the various formats in which the data is available and is accompanied by extensive documentation. The different versions of the corpus are realised as tags in GitLab, while these tagged versions are also uploaded into the Eurac Research CLARIN Centre (ERCC), the CLARIN DSpace repository hosted by the IAL, so they can be easily downloaded by less tech-savvy users[19]. Another advantage is, of course, that this integration of the data into a CLARIN Centre will make the metadata available to various search engines (e.g. the VLO or the OLAC search) and it can therefore be discovered easily. All the data for a tagged version is available both at the ERCC and on GitLab with each of these hosting platforms referencing the other. At both places, all versions are accompanied by a changelog that explains the changes between versions. On GitLab, the interested user can also make use of the integrated version diff to get more fine-grained information on the changes between versions.

5 Challenges and Pitfalls

While trying to implement the paradigm as described above, we already encountered a number of surprisingly challenging cases than the ideal one of a corpus that can be provided completely as open access. As linguistic corpora always consist of personal data produced by individuals there are both privacy and IPR concerns that need to be considered. And if not all of the data can be made publicly available, there has to be additional access protection both on the side of the DSpace repository and on the side of GitLab. While it is easy to have some data require a login with an academic account (using the CLARIN federated login) in DSpace, the GitLab repository should ideally not be made completely password protected, but have at least an openly available landing page that describes the corpus. We have tried to implement this for the DiDi corpus (Frey et al., 2015) using git submodules where the main repository with the documentation and the overview of the various data formats is publicly accessible and the actual data is in sub repositories that require a login[20]. It is likely that more complex access scenarios will prove even more difficult to map to a code hosting platform.

[13] https://kubernetes.io/
[14] https://cloud.google.com/kubernetes-engine/
[15] https://azure.microsoft.com/en-us/services/kubernetes-service/
[16] https://aws.amazon.com/containers/
[17] https://cloud.google.com/blog/products/serverless/introducing-cloud-run-button-click-to-deploy
[18] https://gitlab.inf.unibz.it/commul/merlin-platform
[19] https://hdl.handle.net/20.500.12124/6
[20] https://gitlab.inf.unibz.it/commul/didi/data-bundle

There is also, as always when using external services for sensitive data, the consideration whether one should store their data with a commercial provider, possibly one based in another country and jurisdiction. One way to avoid this is the GitLab "community edition"[21] that can be installed on local infrastructure, meaning that the researcher/the institute will be able to keep full control of the data and container hosting.

Another possible pitfall is foreseen in the use of Dockerfiles[22] to create a persistent setup of a tool-chain. Unfortunately when writing the Dockerfile there is currently no enforcement of explicit versioning of the used software. This means that two containers built using the same Dockerfile at two different points in time can contain two slightly different versions of the software which may result in different behaviour. This has to be kept in mind when creating the Dockerfile and can be averted by always requiring explicit versions of a tool, for example by using the functionality that is provided for this by the various Linux package managers.

6 Conclusion and Outlook

Reproducibility of corpus-linguistic research is a central problem within the linguistic community (Wieling et al., 2018) which is currently not well addressed in a large number of projects. In this paper, we have presented a promising approach to tackle this problem using existing tools from the software development world. We have started using this approach for existing corpora, but already encountered a number of potential problems of which we highlighted some. Nevertheless, this way of ensuring that results in corpus-based linguistic research can easily be reproduced by fellow researchers seems like an idea that is worth pursuing in the future. It makes sense for the CLARIN community with its focus on providing infrastructure for improving the process and the outcomes of research to follow up on this and see how it can help researchers to make their research easier to reproduce for others. Maybe CLARIN can offer a central infrastructure for the hosting of code/data on the one hand and docker images on the other, for example by installing a gitlab instance on one of the CLARIN-ERIC servers. Another possibility would be to develop guidelines or best practices for this kind of setup that can then be followed by the CLARIN community and which would result in a distributed infrastructure for version managed data and processing tool-chains.

References

Adriane Boyd, Jirka Hana, Lionel Nicolas, Detmar Meurers, Katrin Wisniewski, Andrea Abel, Karin Schöne, Barbora Stindlová, and Chiara Vettori. 2014. The MERLIN corpus: Learner language and the CEFR. In *LREC*, pages 1281–1288.

Jennifer-Carmen Frey, Aivars Glaznieks, and Egon W Stemle. 2015. The DiDi corpus of South Tyrolean CMC data. In *Proceedings of the 2nd Workshop on Natural Language Processing for Computer-Mediated Communication/Social Media at GSCL2015 (NLP4CMC2015), Essen, Germany*, pages 1–6.

Willem JM Levelt, PJD Drenth, and E Noort. 2012. Flawed science: The fraudulent research practices of social psychologist diederik stapel.

Joel Nothman, Hanmin Qin, and Roman Yurchak. 2018. Stop Word Lists in Free Open-source Software Packages. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 7–12, Melbourne, AU, July. Association for Computational Linguistics.

Martijn Wieling, Josine Rawee, and Gertjan van Noord. 2018. Reproducibility in Computational Linguistics: Are We Willing to Share? *Computational Linguistics*, 44(4):641–649, December.

[21] `https://about.gitlab.com/install/ce-or-ee/`

[22] the recipe for constructing a container, see `https://docs.docker.com/engine/reference/builder/`