Advancing OMR as a Community: Best Practices for Reproducible Research

Alexander Pacha
Institute of Visual Computing and Human-Centered Technology
TU Wien
Vienna, Austria
alexander.pacha@tuwien.ac.at

***Abstract*—Optical Music Recognition has been under investigation for over 60 years but remains an unsolved problem, because research happens distributedly, often without reusability in mind. As scientists, it should be one of the goals to share knowledge in a way, that it becomes accessible and useful for someone else to build on top. Without that, ones effort is often doomed to rot in a drawer. To oppose this development, not only the paper, but also the source code, datasets, and executables should be made publicly available for the community to finally advance beyond the state, where the wheel is re-invented every time a new researcher joins the field.**

***Index Terms*—optical music recognition, reproducible research**

I. Introduction

Optical Music Recognition (OMR) has seen decades of research with many questions remaining unsolved [1]. Solutions, algorithms, and systems were frequently developed independently, trying to solve particular (sub-)problems [2]–[7]. However, without the perspective of what happens to a project after the scientific publication has been accepted, the advances we have achieved remain scattered across the world. It would be great to share datasets, algorithms, results, and knowledge alike. This is slowly becoming the scientific norm, but to leverage the work of someone else, the author has to do more than just providing a link to the source-code: properly documenting the code base, providing automatic tests that can be executed on a continuous integration platform and ideally one-click solutions for running the application are necessary. These things are part of the standard repertoire used in industry projects and should be equally relevant in scientific projects from the field of computer science.

To finally solve OMR, the entire community has to work together. Therefore, this paper contains a collection of many best-practices, showcases existing works that already follow these guidelines and give hints on how to make research truly reproducible and reusable for other researchers to build on.

II. Towards Reproducible Research

There are three kinds of artifacts that researchers commonly share with the community: knowledge, primarily in the form of scientific articles, datasets, and source code. Papers usually undergo a scientific peer-review to ensure a certain quality level and are published in journals, or presented at conferences. Many of these publications now come with links to the source code [8]–[12] that is hosted on Github or similar platforms. Sharing access to the repository allows not only to see the final result, but also the steps that led to that result. To enable free reuse, an appropriate license should be attached to the repository, e.g., the MIT license. https://tldrlegal.com/ contains a concise summary of many popular licenses. Note, that if no license is specified, default copyright applies that restricts the allowed usage heavily.

A. Publishing source code

When publishing source code, there should be at least a minimum amount of documentation in the form of a README, which summarizes the purpose of this repository and how to build or run the application. Keep in mind, that the actual source code should contain only very few comments and instead use good names for methods, variables, and packages because comments in source code are considered a code smell, that indicate potential issues [13]. On the other side, sensible documentation that is generated directly from the source code[1] can help other users to quickly get started, ideally with a few examples on how to use your project. http://muscima.readthedocs.io/en/latest/Tutorial.html contains an example of a good tutorial.

B. Publishing executables

For the sake of preventing unexpected regressions, unit- and integration-tests come in handy and can assist the developers. If being used in combination with a continuous integration platform, such as Travis[2], that automatically builds and tests the project on each commit. Many of these services are freely available for open-source projects. They can be used to not only check if the code still behaves as expected, but also as a distribution platform for binaries, which can directly be executed by potential users. If an application has a complex environment and requires specific dependencies, consider using containerization, e.g., with Docker[3]. Such containers allow specifying all necessary dependencies with a version that is known to work to avoid conflicts with other versions of those libraries that might be installed on the target system. The

[1] https://readthedocs.org/ (accessed 04.07.2018)
[2] https://travis-ci.org/ (accessed 04.07.2018)
[3] https://www.docker.com/ (accessed 04.07.2018)

additional benefit of setting up an application as a ready-to-use container is a possible integration with environments like the DIVAServices [14], that provide interactive demos as part of their website[4].

C. Publishing Datasets

A delicate matter concerns the creation of datasets. This is a sore point, that is still a major issue within the OMR community. Although there are a few large datasets that have recently been published or that will be published in the near future, these datasets are not compatible with each other due to different notation formats, encodings, and granularity of the underlying information. WoRMS offers a great possibility to address this issue as a community. From a practical point of view, datasets will probably always be encoded somewhat differently, but if the authors claim, that their dataset can (in theory) be exported into a particular format, it is their responsibility to provide a respective converter. As a general guideline, a simple, plain format (e.g., comma-separated-values) that can be converted easily into other data formats is always preferable. It is also beneficial, if additional visualization functions are provided, that can help to inspect intermediate and display final results. Finally, to gain visibility, a dataset can be referenced from the OMR datasets project website (see Figure 1).

III. Showcases

Two projects demonstrate the implementation of above-mentioned guidelines: The OMR datasets project[5] and the Music Object Detector[6] [15]. Their source code is freely available under an MIT license, and detailed README files explain how to use the source code, how to reproduce the results and how to test the final system with pre-trained models and demo-scripts. The Music Object Detector is also available via the DIVAServices Spotlight, where arbitrary music scores can be uploaded to test the object detection capabilities easily.

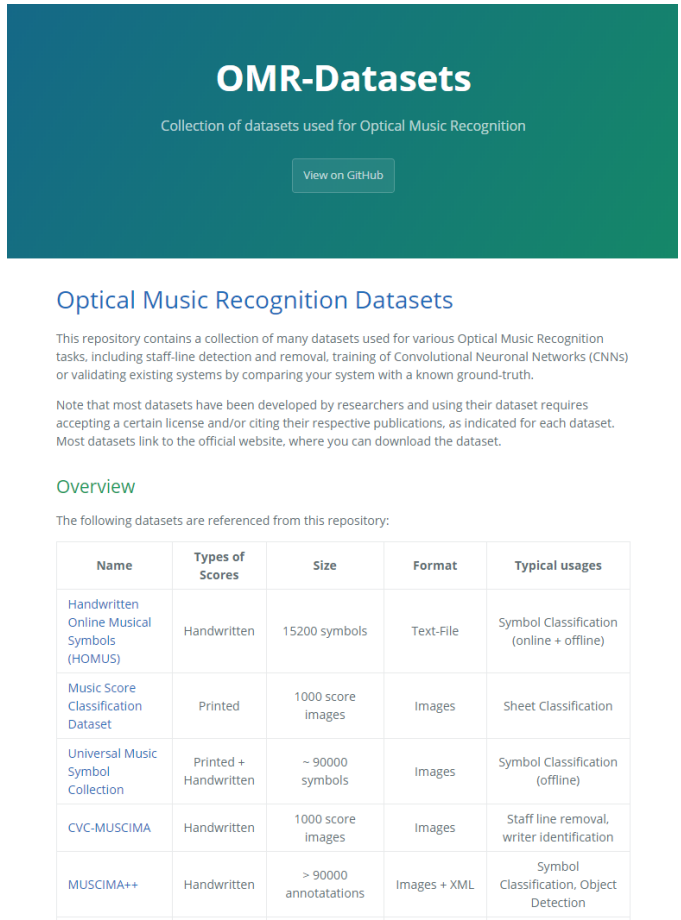

Fig. 1. Screenshot of the public website of the OMR dataset project.

References

[1] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. Marcal, C. Guedes, and J. S. Cardoso, "Optical music recognition: state-of-the-art and open issues," *International Journal of Multimedia Information Retrieval*, vol. 1, no. 3, pp. 173–190, 2012.

[2] J. Ashley, B. Laurent, P. Greg, and E. I. Fujinaga, "A comparative survey of image binarisation algorithms for optical recognition on degraded musical sources," *ACM SIGSOFT Software Engineering Notes*, vol. 24, no. 1985, pp. 1994–1997, 2008.

[3] C. Dalitz, M. Droettboom, B. Pranzas, and I. Fujinaga, "A Comparative Study of Staff Removal Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 753–766, 2008.

[4] A. Fornés, "Primitive Segmentation in Old Handwritten Music Scores," in *International Workshop on Graphics Recognition 2005*, 2005, pp. 279–290.

[5] P. Bellini, I. Bruno, and P. Nesi, "An off-line optical music sheet recognition," in *Visual Perception of Music Notation: On-Line and Off Line Recognition*. IGI Global, 2004, pp. 40–77.

[6] D. Blostein and H. S. Baird, *A Critical Survey of Music Image Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 405–434.

[4]http://divaservices.unifr.ch/spotlight (accessed 04.07.2018)

[5]http://apacha.github.io/OMR-Datasets/ (accessed 04.07.2018)

[6]http://github.com/apacha/MusicObjectDetector-TF (accessed 04.07.2018)

[7] J. Calvo-Zaragoza, A. H. Toselli, and E. Vidal, "Early handwritten music recognition with hidden Markov models," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 319–324.

[8] A. Pacha and H. Eidenberger, "Towards a universal music symbol classifier," in *Proceedings of the 12th IAPR International Workshop on Graphics Recognition*, New York, USA, 2017, pp. 35–36.

[9] J. Calvo-Zaragoza and D. Rizo, "End-to-end neural optical music recognition of monophonic scores," *Applied Sciences*, no. 4, 2018.

[10] L. Tuggener, I. Elezi, J. Schmidhuber, and T. Stadelmann, "Deep watershed detector for music object recognition," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*, 2018.

[11] J. Hajič jr. and M. Dorfer, "Prototyping full-pipeline optical music recognition with muscimarker," in *Extended abstracts for the Late-Breaking Demo Session of the 18th International Society for Music Information Retrieval Conference*, 2017.

[12] K. MacMillan, M. Droettboom, and I. Fujinaga, "Gamera: Optical music recognition in a new shell," *Proceedings of the International Computer Music Conference*, pp. 1–4, 2002.

[13] A. Shvets, G. Frey, and P. Marina, "Source making." [Online]. Available: http://sourcemaking.com/refactoring/smells/comments

[14] M. Würsch, M. Liwicki, and R. Ingold, "Web Services in Document Image Analysis - Recent Developments and the Importance of Building an Ecosystem," in *13th IAPR Workshop on Document Analysis Systems*, Vienna, Austria, 2018, pp. 334–339.

[15] A. Pacha, K.-Y. Choi, B. Coüasnon, Y. Ricquebourg, R. Zanibbi, and H. Eidenberger, "Handwritten music object detection: Open issues and baseline results," in *2018 13th IAPR Workshop on Document Analysis Systems (DAS)*, 2018, pp. 163–168.