Some Boring Hacks for OMR

Organization

- **Jorge Calvo Zaragoza**, University of Alicante, Spain
- **Jan Hajič Jr.**, Charles University, Czech Republic
- **Alexander Pacha**, TU Wien, Austria

Proposed duration: full day
Logistical requirements: tables, projector, power plugs
Technical requirements: (Good) Internet connection

Workshop proposal

Optical music recognition (OMR) has undisputedly made significant progress in recent years, but it is still difficult to leverage this progress in the various application scenarios. Among the obstacles are a dearth of interoperability between individual OMR contributions and systems: it is fine that system A, for instance, detects notation objects better than system B, but system A cannot be swapped into another OMR pipeline because it either uses a slightly different set of objects, or at least represents the output differently. While this situation has been recently improved with the introduction of extensive datasets[1], so that individual researchers can just follow the specifications of these, again the dataset builders have resolved the issues involved in designing a data model for OMR differently. Furthermore, there is a prominent lack of evaluation standards --- not just the metrics themselves, but also the underlying formal specifications and implementations. Authors of both systems A and B might be reporting different metrics, or if they are reporting the same metric, they may be using different implementations for the same metric, with potentially different solutions to some underspecified corner cases.

At the same time, there are multiple ongoing initiatives to improve the way music notation is represented. The SMuFL specification of a naming system for engraving music, the W3C Music Notation Community Group, and of course the Music Encoding Initiative (MEI) are the major drivers of change in this domain, and interoperability is -- or at least should be -- a keyword for each of these initiatives. However, none of them takes the needs of OMR into account, such as representing syntactically incorrect scores that nevertheless may often be a result of the recognition process. At least for the purposes of OMR research, such eventualities need to be taken into account. Since OMR is understandably not a major concern for these communities, it is the OMR community who must put in the technical work to make sure its outputs can be reasonably integrated into users' application scenarios.

[1] https://apacha.github.io/OMR-Datasets/

We intend this workshop --- emphasis on “work” --- to be structured like a hackathon with multiple groups focusing on individual sub-tasks, such as evaluation, format specification or tooling. The output of the workshop should be a suite of OMR software tools, format specifications and evaluation protocols that help conducting future experiments in a replicable and comparable manner, similar to the mir_eval[2] package or TREC evaluation suite[3]. Frankly speaking, this is work that no one will ever do individually, but at the same time it is work that is necessary in order to (a) push the field of OMR further towards replicability, (b) make it possible to integrate OMR improvements into existing pipelines, especially those that rely on MEI.

We see the Music Encoding Conference (MEC) as an ideal opportunity to get this kind of work done since a significant portion of both the OMR community and the MEI community will be present. On the one hand, there is sustained interest in OMR from the music library community, as evidenced e.g. by the presence of two large publicly funded institutions at the very informal, very first Workshop on Reading Music Systems (WoRMS)[4] in Paris (Bavarian State Library, National Library of Finland) who would make quite natural users of MEI. On the other hand, OMR evaluation protocols should not be developed by (only) people on the technical side, but stakeholders have a fundamental relevance in this issue. MEC, therefore, represents a suitable venue to hold such hackfest.

There will be a preparatory phase in which we collect ideas on what to hack in a Github repository or equivalent. Some ideas were already discussed at the aforementioned WoRMS workshop.

[2] https://github.com/craffel/mir_eval
[3] https://trec.nist.gov/trec_eval/
[4] https://sites.google.com/view/worms2018