Building Open-Source Tools for Reproducible Research and Education

Experiences developing and distributing LensKit

Michael Ekstrand, GroupLens Research, University of Minnesota (ekstrand@cs.umn.edu)

Concerned with the difficulty of reproducing and extending recommender systems research results, we have developed LensKit [1], an open-source toolkit for building, researching, and studying recommender systems. In the course of developing, distributing, and maintaining LensKit, we have encountered a number of challenges, as well as good experiences and responses, with maintaining open-source software for the benefit of the research community.

Our interest in this workshop centers around two aspects of open-source software as a component of research: (1) the promise of open source software for promoting reproducible research and making research results easy to reuse and extend, and (2) discussing ways to deal with and mitigate the challenges of producing useful, high-quality open source software in an academic environment.

The Story of LensKit

Our work on LensKit was motivated in part by several concerns with the state of recommender systems research, including:

- The difficulty of understanding existing recommender algorithms in enough detail to implement them. Many algorithms are published in the research literature, but the level of detail they provide varies greatly. This makes it difficult in many cases to understand exactly what the original authors did, particularly in edge cases, in enough detail to re-implement the algorithm and make it useful.
- The difficulty of reproducing and comparing research results. In addition to the variation in their descriptions of algorithms, recommender research papers do not always specify their evaluation protocols in enough detail to reproduce the exact measurement.
- Inconsistent choice of evaluation setups and metrics.
- Because of the difficulty of understanding algorithms, algorithm authors do not always compare against high-quality implementations of prior work. For example, the baseline algorithm may lack state-of-the-art optimizations and data normalization, resulting in an evaluation that overstimates the improvement made by the proposed new approach.

We intend LensKit to help address these difficulties. Open-source access to high-quality implementations of existing algorithms will make it easier to test new algorithms against good baselines. A common evaluation framework will make it easier to write and use well-defined, reproducible experimental setups. And the components and infrastructure provided by LensKit make it easier to try new recommender ideas in the first place. Our hope is to see an increasing number of recommender researchers publish source code for their algorithms and scripts to re-run their evaluations on publicly available data sets, and we hope that LensKit (and open-source recommender software in general) makes this easier.

We are not the only ones to see open-source software distribution as a useful tool for advancing the field. There are a number of other open-source projects to provide recommender platforms; among the most notable are Mahout[1] and MyMediaLite[2]. Mendeley has recently open-sourced a recommender platform with a particular emphasis on reproducible evaluations[3].

[1] http://mahout.apache.org
[2] http://mymedialite.net/
[3] https://github.com/mendeley/mrec

Looking to open-source software to improve the reproducibility and reusability of research results is also not unique to the field of recommender systems. For example, Lougee-Heimer [2] observes a similar set of problems arising from common software practices in the operations research community and proposes open-source software as a tool for remedying them; the benefits seem applicable to many more fields.

Challenges and Difficulties

We believe that open source distribution is a powerful way to both disseminate research results in a usable form, to aid education, and to support reproducible, extensible research. This is not a trivial undertaking, however; as Reghr [3] notes, the structure of the academic environment creates challenges for significant software development efforts. Some projects, such as IRNIA's OCaml and the University of Pensylvania's UNISON, are able to deliver and maintain robust, high-quality software that is widely used. But it is a lot of work, and is not commonplace. Some of the challenges include:

- Bug fixing and software maintenance. Unless the project has developed a diverse, self-sustaining community, is easy for bugs and enhancements that don't directly impact the research work of the primary developers to languish untouched.
- Documentation. The combination of the curse of knowledge and limited time to divide between research and development means that the work of writing extensive, accessible documentation can easily be neglected. The documentation, once written, also needs to be maintained.
- Advertising and community management. We have tried to make LensKit visible to the research community, but building up a community around a software project is difficult in general, and needing to divide time between research effort and software project management makes it harder.
- Compatibility and change management. If the software is being actively used in and developed for research, it can change very quickly. In addition to making documentation out of date, it can make the software a rapidly moving target that is unattractive to develop against.

This challenges are compounded by mismatches between common requirements and evaluation structures in academic environments and the effort necessary to build and maintain high-quality software (bug fixing isn't writing research papers or grant applications). While high-quality software may improve the impact of research, it can be difficult to allocate time and resources to maintaining and supporting the software for a broad audience.

Bio and Acknowledgements

Michael Ekstrand is a Ph.D candidate in computer science at the University of Minnesota, working in the GroupLens research group. He research focuses on how recommender algorithms differ in ways that are meaningful to users' information needs. He is the founding and lead developer of LensKit. More information on his research can be found at http://elehack.net/research.

This work has been supported by NSF grants IIS 05-34939, 08-08692, 08-12148, and 10-17697.

References

1. Michael D. Ekstrand, Michael Ludwig, Joseph A. Konstan, and John T. Riedl. 2011. Rethinking the Recommender Research Ecosystem: Reproducibility, Openness, and LensKit. In *RecSys '11*.
2. Robin Lougee-Heimer. 2003. The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community. *IBM Journal of Research and Development* **47**(1), pp. 57–66.
3. John Regehr. 2013. Producing Good Software From Academia. http://blog.regehr.org/archives/1058 (accessed Nov. 15, 2013)