2022

On the ethics of working with library technology: the case of the Open Journal Matcher

Mark E. Eaton
CUNY Kingsborough Community College

On the ethics of working with library technology: the case of the Open Journal Matcher

Mark E. Eaton[a]*

[a]*Library, Kingsborough Community College, City University of New York, New York, USA.*

*mark.eaton@kbcc.cuny.edu; ORCiD: 0000-0001-8579-5073

This work was funded by PSC-CUNY under Grant 64672-00 52; and by a Google Cloud Platform Research Credit Grant.

On the ethics of working with library technology: the case of the Open Journal matcher

Through the methodological lens of a case study, this practical communication aims to capture and analyze one aspect of the multi-faceted ethical concerns that arise with building with technology in libraries. I define and deploy the concept of "pervious" technology (the opposite of impervious technology) as a way to think through how librarians work with technologies in their workplaces. I describe how pervious technologies are readily manipulable tools that librarians can reach into (metaphorically, hence "pervious"), giving them a greater say in how they engage with their library work. Specifically, I examine the ethical implications of pervious and impervious technologies in library workplaces by analyzing a specific, librarian-built application: the Open Journal Matcher.

Keywords: journal recommender; ethics; library organizations; technology; web applications; librarians; diversity

Introduction

In March 2019, I began building the Open Journal Matcher (OJM, https://ojm.ocert.at), a recommender tool for academics looking to find a suitable journal for the publication of their scholarly work. The OJM allows a user to paste in their draft abstract, and the application will compare it to abstracts from more than 5,600 English-language journals indexed by the Directory of Open Access Journals (DOAJ, https://doaj.org/). The OJM then returns the top five best-matching open access journals. This can be very useful to an author trying to find an appropriate journal for their work.

Upon its public release in June 2020, the OJM received a favorable reception on social media from the open scholarship community. Since then, it has been linked to from library websites all over the world. This has reaffirmed for me the need for such a tool, and has encouraged me to continue developing it. In some respects, the OJM is not unique; there are other journal matching projects, such as those described by Schuemie and Kors (2008), Kang et al. (2015), Mohtaj and Tavakkoli (2018), and Errami et al. (2007), as well as some commercial projects. But, to my knowledge, the OJM is the only one that is fully interdisciplinary and fully open source. The code for the matcher application, the code for the matching algorithm, and the content of the journals, is all openly licensed. Additionally, the OJM covers the full range of disciplines reflected in the DOAJ.

The OJM provides a practical frame for the analysis that follows. My argument is that we can better support our libraries when we build technologies that are *pervious* (the opposite of *impervious).* I define pervious technologies as those that anyone can

manipulate, adapt, and use to their liking. They are technologies that we can reach into (metaphorically, hence pervious) and tinker with.

When I was building the OJM, I wanted to make a technology that librarians can reach into. I wanted everyone to be able to determine their level of engagement with the technology stack. The OJM allows this. As a result, this project can be experienced in very different ways from one user to another. To put it succinctly, it is a pervious tool. Some example uses (in order of increasing involvement) are:

(1) A user can interact with the web version of the OJM (at https://ojm.ocert.at), and use it to find appropriate matching journals.
(2) A user can submit modifications or patches to the project's GitHub repository.
(3) A programmer who wants to tinker a bit can clone the repository, set-up a local installation, and run a version of the tool themselves.
(4) To go even further, a curious librarian could dig into the underlying matching algorithm to see how the matcher is making its recommendations.

All of these are achievable, realizable ways that a librarian could engage with the OJM. This points to a central feature of pervious technologies: that it is up to the user how much they want to be involved. My argument is that allowing librarians to decide upon their own level of participation is empowering. Pervious tools and pervious methodologies allow librarians to determine their own level of technological engagement.

To better grasp the distinction I am making, it may help to compare some widely-known examples of pervious and impervious code. Openly-licensed code downloaded from GitHub, for example, can usually be modified as much as one likes. In this sense, it is pervious technology. On the other hand, Twitter, at the time of writing, is impervious to users. The vast majority of users have no opportunities to change Twitter's behavior. At best, it has a limited public API that developers can use, as long as they do so in accordance with Twitter's terms of service. But, for the most part, it is completely beyond the user's reach.

Perviousness should not be thought of too narrowly. The concept of perviousness describes not only technologies, but also methodologies. It describes the ways in which technologies are enacted. Because of this, it can be compared to other workflow methodologies, as I do below. The concept of perviousness is a prompt to think about how librarians' work encourages them to reach into their tools (or not). The choices they make when building with technology will have a major impact on how technologies are deployed, both by themselves and by others.

Literature review

This paper will suggest that pervious tools are a constructive way to develop the technical skills of library workers. Open and pervious tools helpfully allow librarians to participate in their technology as much as they want. Being actively involved with technologies leads to an exciting opening of possibilities. For example, Papert (1993), di Sessa (2001), Kafai and Burke (2014), and Vee (2017) all point to the expansiveness that

learning technologies can bring. Each of these authors describes a technological awakening that is common among people learning to program. Librarians who are able to engage with their technologies as much as they choose may likewise experience an expansion of their professional and technical horizons. The possibilities of working with code begin to appear when somebody rolls up their sleeves and digs in. In a practical sense, librarians' involvement with the pervious technologies in libraries is only limited by available time, interest, and other responsibilities, among other things. Yet contemporary technologies are also, for practical purposes, nearly endlessly complex, so engagement with pervious tools can also be as broad and as deep as can be imagined.

Pervious workflows have strong resonances with some other contemporary workflow theories. For example, Agile methods, which grew out of the software industry, provide an adjacent (and extensive) literature on how to improve productivity and teamwork. One of the focal points of Agile is on building collaboratively, rather than in silos (Moreira, 2013; Medinilla, 2012). It is an approach that dovetails nicely with the proposal to build accessible and pervious technologies. The principal difference between perviousness and Agile is in their respective objectives. While Agile is focused on increasing productivity, meeting business objectives (Cobb, 2011), and maximizing profit (Medinilla, 2012), pervious methodologies aim instead at diversity and inclusion (as will be shown below). Nonetheless, there are useful insights that can be gleaned from Agile approaches, for example: the benefits of being relentlessly responsive to user's needs; and the positive consequences of focusing on iteration. Such lessons can illuminate library workflows that can help improve librarians' work.

There is also an analogy between perviousness and user-centred design. User-centred design is often oriented toward meeting the needs and desires of the end user (Kraft, 2012). On the other hand, pervious approaches are aimed at meeting the intermediate needs of the librarians, programmers and other tinkerers who are building tools for their communities. In a sense, librarians building pervious tools are doing user-centred design for an audience that is internal to their professional community. In this light, pervious technologies can be seen as a powerful intermediate step in the building of tools for user communities.

The last workflow methodology I will touch upon here is open source. The elaboration of an open source workflow is arguably one of the more important methodological developments in how software has been built. The open source movement has shown that voluntary, decentralized labor can effectively produce valuable and important code.

There is a long history of open source in library scholarship. Some noteworthy examples include Muir (2005), who documents the early days of the arrival of open source software in libraries; Balnaves (2008), writing a few years later, who synthesizes knowledge gleaned from comparing open source library tools; while more recently, Njoku (2017) advocates for the broader, global spread of open source library technologies. The technical work described in this paper draws upon and adds to this literature.

Using open source licenses is an obvious step that can be taken to build library technologies that are more pervious. But it is not sufficient. Technologies can also be built in a way that is amenable to modification and re-purposing. To a certain extent, this involves building reusable code, a practice which is already well established in

open source and in the literature on coding practice (Hillard, 2020; Hunt & Thomas, 1999). Tools can also be made pervious by providing users with opportunities to use the tools as they see fit, by creating affordances for tinkering and building in kind.

It would be a mistake to limit this concept of pervious technologies to those that are openly licensed. I intend "pervious" to refer more broadly to tools that can be tinkered with. Some closed source tools can (and sometimes do) have ample room for tinkering, and can therefore be considered pervious. The reality is that software today is much too complex for "pervious" and "open" to be synonymous.

Google Cloud Platform, which is used extensively by the OJM, is an excellent example of a technology that is highly pervious without being openly licensed. Its affordances are such that it can be tinkered with as much as most anyone would want, yet it is not openly licensed. For the purpose of this case study, it is significant that Google Cloud Platform clearly demonstrates that there is no exact match of perviousness to openness. Nonetheless, generally speaking, it is probably accurate to say that pervious tools are often openly licensed.

To bring together the various literatures touched upon here, it can be seen that the concept of perviousness is in good company. Given that there are obvious parallels to other workflow concepts, perhaps perviousness does not carve out radically new theoretical ground. Nonetheless, irrespective of its novelty, I argue that the concept of perviousness is very useful because, as will be seen in the following analysis, it provides a particular emphasis that makes it possible to think though ethical issues in the library workplace.

Technical Review

[PLACE FIGURE 1 HERE]

Figure 1. Screenshot of the Open Journal Matcher

It will be helpful to take a brief, deep dive into the technical details of the OJM, to get a sense as to how the project implements pervious technologies and methodologies. The OJM is built with the intention of assembling what I call a pervious technical stack. The tools listed below were chosen because they allow librarians and programmers the opportunity to reach in and tinker with the technologies. This approach is critical to the ethical argument that follows.

At the core of the project is a Flask application (https://flask.palletsprojects.com/en/2.0.x/) that serves up the web application and coordinates its data processing work. Flask is a widely used tool, openly licensed under the BSD 3-Clause license (https://opensource.org/licenses/BSD-3-Clause). My use of Flask is somewhat similar to the Django-based approach suggested by Mohtaj and Tavakkoli (2018), insofar as they also rely on similarly popular Python web frameworks.

In this application, Flask serves up web pages, but it also calls a Google Cloud Function that I created to do the necessary intensive data processing. This data processing is done asynchronously with Python. Asynchronous techniques in Python are more fully described by Solomon (2020) and Hattingh (2020). My Google Cloud Function is called ~5,600 times asynchronously, once for each journal.

Each invocation of the Google Cloud Function loads pre-trained word vectors from a natural language processing library called spaCy (https://spacy.io/). The Cloud Function uses spaCy to compare the user's draft abstract with the abstracts of a specific

open access journal. It produces a similarity score, which the Cloud Function returns to Flask. The Flask application assembles these results and sorts them by their similarity scores. It determines the top five matches, retrieves these titles from the DOAJ API, and then displays these results to the user. This workflow can be seen in Figure 2.

[PLACE FIGURE 2 HERE]

Figure 2. A schematic view of the Open Journal Matcher

The critical computational work is done with spaCy. In this application, spaCy uses word vectors to do its similarity comparisons. Word vectors are representations of words in a multi-dimensional space, where each dimension represents one characteristic of the word. For a more in-depth discussion of word vectors, see Parrish (2017) and Kearns and Roth (2020). The spaCy library comes with pre-trained vectorized models. Similarity between two texts can be calculated by comparing the similarity of their word vectors. The OJM draws upon this functionality.

Discussion

While the technical details may be interesting, I would like to go further and consider the ethical implications of this pervious infrastructure. For example, critical attention should be applied to spaCy's pre-trained models. It is necessary to be aware that bias can be trained into these models, both easily and unintentionally. This is not meant to be an accusation, but a reminder to keep critical faculties engaged. What interests me here are not the details of spaCy, but rather how, from a pragmatic perspective, salient ethical questions can be asked when using pervious technologies.

In what follows, I hope to lay the groundwork for further detailed inquiry into algorithmic tools. My argument is that librarians should be actively thinking about how the tools they use may facilitate bias, and how they can address that problem. Kearns and Roth (2020), and Broussard (2019) provide helpful examples of how such an analysis might proceed. What is important here is that this type of analysis is only possible when the tool is pervious or openly licensed. It is possible to meaningfully address issues of bias only when the tools are pervious enough to peek inside.

This is because using non-open, impervious systems can make some very difficult bias problems even more intractable. For example, Broussard (2019) and Cowan (2019) cite pernicious problems of bias in proprietary criminal sentencing algorithms and recidivism predictors. The sentencing software they examine was shown to systematically favor harsher sentencing for people of color. While that is fundamentally unacceptable, the ethical problems created by these tools is compounded by the fact that the software is closed and inaccessible. Unfortunately, this type of problem is not unique to the judiciary: Noble (2018) describes similar opaque problems of bias in search engines, where black women are routinely sexualized and objectified in search results; while Reidsma (2019) shows significant algorithmic problems in library discovery algorithms, where unintended and troubling results are surfaced in response to user queries. These documented cases of algorithmic bias strongly suggest that, for ethical reasons, librarians need to have the ability to reach into algorithmic processes.

One reason that I chose to use spaCy is that it is released under the MIT license (https://opensource.org/licenses/MIT). This license is widely regarded as being very open. This licensing was important to make this a highly pervious project. SpaCy's

openness and perviousness are what give access to the algorithmic logic that undergirds spaCy and the OJM. While it would be highly speculative at this point to try to guess what a deep dive into spaCy's algorithmic logic might find, it is clear that the possibility of undertaking on such an algorithmic analysis is important to us here. Pervious tools make it entirely up to the programmer as to how deeply they might want to dive in.

Implications

This practical communication intends to serve as a starting point for considering the ethics of pervious and impervious technologies. I maintain that when we build pervious tools, it helps librarians become more technologically engaged. I also argue that pervious technologies encourage us to build more equitable libraries. To be clear, it is not my intention to suggest that the OJM is an ethically exemplary project. Rather, it is an ordinary project. In this context, however, this ordinariness is also one of its virtues. It provides us with a useful sounding board to work through common, yet ethically fraught problems.

In my opinion, the best outcomes for libraries are realized when librarians use and build technologies that are pervious and manipulable. These results are achieved when librarians can *reach into* their technologies. By this, I mean that it is useful to be able to determine one's own level of engagement with the technical stack. It is possible to reach in as far as one wants, make the changes that are desired, and perhaps ignore (for now) other parts. From this perspective, technology is not a black box. It is not something that is just for initiates. Deployed constructively, it can be something that anyone can work with to their own satisfaction.

While it is interesting and useful to consider the technical affordances of our library technologies, as I have done above, it is also important to situate these discussions in the context of library workplaces (including our virtual workplaces). Moreover, as I said above, perviousness is a social concept, as well as a technical one, so it is worth examining how perviousness affects our librarianship. I will now briefly turn to organizational questions.

To put it succinctly, in many libraries, there is often a problematic tight coupling of technical knowledge and ethical influence over library projects. Specialist library technologists usually work on, and determine how, technologies are used in their library. This is a very common, yet impervious approach. I would argue that when technical and ethical decisions are taken by a small number of technologists in a library, the outcomes are frequently problematic. This is largely a consequence of impervious workflows. Monopolization of technical decisions by a few technically-minded librarians or programmers can easily lead to less-than-optimal outcomes. Thankfully, it is an approach that can, and should be, be contested.

Giving technologists too much say over library technologies sets up an unpalatable ethical hierarchy in the library. Providing the technologist with a privileged position to make decisions about library tools does very little to distribute technical agency, knowledge about the tools, or ethical influence over the library's work. Moreover, as we are all fallible humans, librarians need to be prepared for the fact that a technologist will sometimes fail at fully recognizing or addressing all of the ethical issues raised by a project. Ethical lacunae can sometimes go unaddressed as a result of the monopolization of

technical knowledge. When this happens, discrepancies in influence within the organization, usually along predictable racial and gendered lines, are likely to be reinforced.

How do we deal with the biases that our workflows produce? Technical solutions have been suggested. Cowan (2019) points to software tools that can help visualize and identify bias, while Kearns and Roth (2020) also suggest technical fixes. However, I suggest that a more readily realizable solution is having more eyes on the problem – and specifically, a diversity of perspectives. Having pervious technologies helps with this.

Pervious technologies are more visible, in Tkacz's (2007) sense, than their closed counterparts. To continue with the visibility metaphor, Raymond (1999) famously said that "with enough eyes, all bugs are shallow". While this aphorism is well known among programmers, it is not without its own agenda: this quote should be read within Raymond's libertarian push to make free software more commercially oriented (Kelty, 2008; Coleman, 2013). But perhaps his aphorism should be radically reworked, to say instead: "with pervious technologies, and enough diverse perspectives, all ethical problems are apparent". Not quite as catchy, but hopefully a more constructive way to look at library technology.

Crucially, I suggest that pervious systems, when combined with diverse perspectives, yield ethically improved systems. Pervious methodologies provide an alternative to the technologist's problematic monopolization of technical authority. Rather than depending upon technologists for technical and ethical decisions, pervious technologies encourage everyone to be as involved as much as they want. Having more librarians participate in technical work is not at all redundant, rather, it increases the expertise of

the entire library. Taking away the privileged role of the technologist makes all librarians into simultaneous teachers and learners of technical skills. It encourages mutual support and learning together.

While pervious technology is certainly not a panacea for the politics of technology in the library workplace, it does have benefits. It obviously does not overcome the systemic inequities in technical and ethical privilege that have, in my opinion, long been the baseline for library organizations. But pervious tools can improve the distribution of technical knowledge in the library, by fostering participation and supportive co-learning for librarians.

Importantly, librarians should discourage monopolization of ethical responsibility and technical knowledge within their organizations, and instead distribute technical participation as widely as possible. This will allow for more participation in technical decision making. San Diego argues that librarians need to take concrete actions to make our communities more diverse and open: “show, don’t just tell” (qtd. in Shivers-McNair, 2021, p. 97). Building pervious tools is a step along that road. When librarians build inclusively, together, they “show”.

Conclusion

The OJM has served an expedient purpose in this case study. It has provided a practical frame to allow reflection on some of the ethical issues facing people building and deploying technologies in libraries. The motivation behind this practical communication has been to think through some of the problems with building with technology in the library workplace. I hope that the concept that I have proposed - perviousness - proves

useful when thinking about library tools. Ultimately, I suggest that moving toward more previous technologies and methodologies can help librarians help each other do more ethically-minded technical library work.

References

Balnaves, E. (2008). Open source library management systems: A multidimensional evaluation. *Australian Academic and Research Libraries*, *39*(1), 1–13.

Broussard, M. (2019). *Artificial unintelligence: How computers misunderstand the world* (First paperback edition). MIT Press.

Cobb, C. G. (2011). *Making sense of agile project management: Balancing control and agility*. Wiley.

Coleman, E. G. (2012). *Coding freedom: The ethics and aesthetics of hacking*. Princeton University Press.

Cowan, E. (2019, October 4). *Ethics & bias detection in the real world*. PyGotham, New York. https://pyvideo.org/pygotham-2019/ethics-bias-detection-in-the-real-world.html

DiSessa, A. (2001). *Changing minds: Computers, learning, and literacy*. MIT Press.

Errami, M., Wren, J. D., Hicks, J. M., & Garner, H. R. (2007). ETBLAST: A web server to identify expert reviewers, appropriate journals and similar publications. *Nucleic Acids Research*, *35*, W12-15.

Hattingh, C. (2020). *Using Asyncio in Python: Understanding Python's asynchronous programming features*. O'Reilly.

Hillard, D. (2020). *Practices of the Python Pro*. Manning.

Hunt, A., & Thomas, D. (1999). *The pragmatic programmer: From journeyman to master*. Addison-Wesley Professional.

Kafai, Y. B., & Burke, Q. (2014). *Connected code: Why children need to learn programming*. MIT Press.

Kang, N., Doornenbal, M. A., & Schijvenaars, R. J. A. (2015). Elsevier Journal Finder: Recommending journals for your paper. *Proceedings of the 9th ACM Conference on Recommender Systems*, 261–264.

Kearns, M., & Roth, A. (2020). *The ethical algorithm: The science of socially aware algorithm design*. Oxford University Press.

Kelty, C. M. (2008). *Two bits: The cultural significance of free software*. Duke UP.

Kraft, C. (2012). *User experience innovation: User centered design that works*. Apress.

Medinilla, A. (2012). *Agile management: Leadership in an Agile environment*. Springer.

Mohtaj, S., & Tavakkoli, F. (2018). Maglet: A Persian journal recommender system. *Proceedings of the 9th International Symposium on Telecommunications*, 348–352.

Moreira, M. E. (2013). *Being Agile: Your roadmap to successful adoption of Agile*. Apress.

Muir, S. P. (2005). An introduction to the open source software issue. *Library Hi Tech*, *23*(4), 465–468.

Njoku, I. S. (2017). Use of open source technology for effective academic libraries services in Nigeria. *Library Philosophy and Practice*, 1686.

Noble, S. U. (2018). *Algorithms of oppression: How search engines reinforce racism*. New York University Press.

Papert, S. (1993). *Mindstorms: Children, computers and powerful ideas* (2nd ed.). Basic Books.

Parrish, A. (2017). *Understanding word vectors: A tutorial for "Reading and Writing Electronic Text."* GitHub Gist. https://gist.github.com/aparrish/2f562e3737544cf29aaf1af30362f469

Raymond, E. S. (2001). *The cathedral and the bazaar: Musing on Linux and open source by an accidental revolutionary* (Revised edition). O'Reilly.

Reidsma, M. (2019). *Masked by trust: Bias in library discovery*. Litwin Books.

Schuemie, M. J., & Kors, J. A. (2008). Jane: Suggesting journals, finding experts. *Bioinformatics*, *24*(5), 727–728. https://doi.org/10.1093/bioinformatics/btn006

Shivers-McNair, A. (2021). *Beyond the makerspace: Making and relational rhetorics*. University of Michigan Press. https://doi.org/10.3998/mpub.11724511

Solomon, B. (2020). Async IO in Python: A complete walkthrough. *Real Python*. https://realpython.com/async-io-python/

Tkacz, N. (2007). Power, visibility, Wikipedia. *Southern Review*, *40*(2), 5–19.

Vee, A. (2017). *Coding literacy: How computer programming is changing writing*. MIT Press.