Challenges and Opportunities for NSLP in Scientific Publishing A Case Study

Thomas Kleinbauer **Michael Didas** **Michael Wagner**
Schloss Dagstuhl – Leibniz Center for Informatics
Wadern, Germany
firstname.lastname@dagstuhl.de

Abstract

Research software is not always meant to reach production-grade quality. The same requirements, for instance, regarding performance, security, or reliability that are imposed on professional software do not necessarily apply in the lab. However, recent years have seen an increased interest to bring cutting-edge research results into production. Specialized natural language processing, such as NSLP, is no exception. In this paper, we discuss three real-life challenges in scientific publishing as they relate to NSLP, and also highlight the potential NSLP has to offer in overcoming these challenges. Specifically, we identify issues related to the elicitation of metadata – particularly with respect to what we term the /metadata externality problem –, privacy and data protection laws, and software reliability. This is not a research paper; rather than introducing novel research results, our intention is to contribute to the academic discourse in the NSLP community by providing the perspective of a potential end-user.

1. Introduction

Over the last decades, funding agencies have increasingly mandated more rigorous dissemination strategies, particularly for projects supported by public funds. In parallel, the science community has seen an increased interest in publishing *reproducible* research since the 2010s, culminating in the strong Open Science movement we have today.

In Computer Science, this – together with the rise of professionalized and easily accessible code and data sharing platforms on the internet – has been a key driver for a change in the scientific publishing culture: more and more research publications do not consist solely of a written article but are accompanied by supplementary material. Examples include research software, datasets, or machine learning models (Liu et al., 2024; Zhou et al., 2024).

This development poses continuing challenges for infrastructure providers, especially scientific publishers. Guided by the FAIR principles (et al., 2016), research output should be *findable, accessible, interoperable, and reusable*. When a scientific publication consists of multiple artifacts, rich metadata that provide informative cross-references between the various parts thus become critically important.

In practice, however, we observe what could be labeled the *metadata externality problem*: researchers are both users and providers of publication metadata; however, while the benefit of rich metadata is obvious, the effort of creating rich metadata often meets some reluctance.

Natural scientific language processing (NSLP) has the potential to bridge that gap, e.g. when metadata can be extracted automatically from the provided artifacts. But this leads to another problem: while cutting-edge research prototypes can deliver impressive results, their implementation is not always production-ready (Trisovic et al., 2022). The reasons for that are manifold, ranging from questions regarding the technical setup, over code quality, to reliability, dependability, and security issues: what is perfectly fine for a research prototype does not always translate directly to applicability in a professional context.

A scientific publisher who operates in a commercial environment might thus shy away from using state-of-the-art NSLP tools, thereby forfeiting the opportunity to help addressing the metadata externality problem.

A complementary dimension to technical issues are legal considerations that also play a more central role for a professional outlet dealing with paying customers. Copyright, privacy regulations, and patent laws are just three examples with the potential of creating liabilities.

In this paper, we do not present novel research results. Rather, we wish to illustrate the situation laid out above by discussing three concrete examples from our real-world experience at *Dagstuhl Publishing* with the hope of providing a useful and novel perspective for the research community in NSLP.

2. Background

Dagstuhl Publishing (DP) is one of the three pillars of *Schloss Dagstuhl – Leibniz Center for Informatics*[1], besides Dagstuhl Seminars and the dblp[2] database. It specializes in high-quality scientific

[1] https://www.dagstuhl.de/
[2] https://dblp.org

Proceedings of the 3rd Int. Workshop on Natural Scientific Language Processing (NSLP 2026) @ LREC 2026, pages 44–48
12 May 2026.

publications in the field of Computer Science, focusing on conference and workshop series as well as scientific journals. DP's mission is to accelerate academic innovation processes and to improve the visibility of research results through Open Access publishing.

Our publishing process is entirely digital. Authors upload their manuscripts to our submission server which also collects additional information from the authors. An important aspect of our process is the elicitation, maintenance, and publication of metadata. This occurs in a guided process for the authors during the upload process, but authors also have the option to refine metadata annotations at a later point.

DP provides style guidelines to assure a high quality in typesetting. Further, we assist authors in conforming to these style guidelines by reviewing the submitted manuscripts and provide careful layout enhancements for approval. Articles are published exclusively in digital form, they are made available for download through a dedicated website[3].

3. Challenges

There are a number of different challenges for publishers of scientific articles. Here, we select three exemplary issues that relate to NSLP. In Section 4, we discuss how NSLP can potentially be of help for each of them. All three issues center around software mentions in research articles.

Case Study: Software Mentions. In the field of Computer Science particularly, articles regularly reference software which is either used as part of the reported research or in some other way associated with the work. Not always, however, are these mentions explicitly marked, but appear simply as part of the article's body.

From the perspective or a rich metadata description of the research work, this is unfortunate, because the ability to extract software mentions automatically from structured data would relieve the authors from manually specifying them again as part of the metadata elicitation.

Note that often times, the reverse direction also occurs, e.g., software repositories that mention academic papers (Wattanakriengkrai et al., 2022). Here, however, we focus on papers referencing software.

3.1. Elicitation of metadata

There is no shortage of metadata standards today that lend themselves to describing scientific articles, such as e.g. Dublin Core (DCMI, 2012), Schema.org[4], JATS (NISO, 2019) and others. For other types of artifacts, in particular software and data, CodeMeta[5] or DataCite[6] are examples of topically specialized metadata schemes.

Such schemes offer a varying degree of complexity and granularity. For instance, Dublin Core consists of just 15 core elements[7] (such as, e.g. *title*, *author*, *year*, etc.), while the Article Authoring subset of JATS v1.4 contains 300 elements. Considering their practical application, the question arises: who is going to provide such elaborate metadata? Clearly, authors of research papers cannot be expected to have the necessary expertise nor the time to fill in long metadata forms correctly. But for a small publisher like DP with around $2,000$ articles published each year, it is equally unrealistic that this task could be handled in-house.

Even in the case of reduced metadata sets or subsets, DP has to carefully balance merit and user experience. As a customer-facing service, we are interested in making the submission process as seamless and smooth for authors and editors as possible. It is a substantial usability problem to elicit an acceptable set of metadata from uploaders without causing frustration or an otherwise negative impression of the process.

3.2. Privacy and data protection

As an organization operating in Germany, DP is subject to General Data Protection Regulation (GDPR, 2016) and other applicable legislation.

This has direct consequences on our handling of customer-provided contents. In particular, this potentially affects DP's ability to use external LLM-based services, which are the basis of many natural language processing tools today. While it is possible to obtain consent from authors with respect to the contents they created themselves, the situation can become significantly less clear when the publications also includes further artifacts.

Software supplements, for instance, are rarely written entirely from scratch, using third-party libraries is common practice. Depending on the restrictions imposed by the accompanying license, great care has to be taken if the source code of a supplement is to be analyzed by an external service. It is quite complex to assess whether a software upload provided by a user may or may not be relayed to a third-party service, and when the authors of the publications are not the authors of all the contained software components, they are in no position to

[3] https://drops.dagstuhl.de/

[4] https://schema.org

[5] https://codemeta.github.io/

[6] https://datacite.org/

[7] Note that a number of refinements and qualifiers have been added to the original scheme in subsequent versions.

grant the publisher any rights in that matter.

A similar, but potentially even more complex situation arises in the case of dataset supplements. If the dataset contains personal data of any participating subjects, it may not be possible to employ external processing, e.g. by an LLM-base service, at all without the explicit consent of each individual related to the dataset.

Of course, consent needs to be obtained from each affected individual if the dataset or the software supplement are intended to be published in the first place. However, verifying whether this is the case for author-provided supplements is challenging if not impossible in practice.

While these considerations directly affect the use of some NSLP techniques, automated language processing also has the potential to mitigate some of these issues, see Section 4.

3.3. Reliance on external software and services

Using software in a commercial context comes with a number of requirements that may be considered less important for research prototypes used only in the lab. When offering a service to customers, aspects such as *reliability*, *performance*, and *security* become highly important. They are key decision points when choosing between a cutting-edge vs. a tried and trusted solution.

When faced with the choice between high accuracy or high reliability, users in a commercial context will likely tend to prefer solutions that work consistently even at the price of lower quality. False positives are worse than false negatives in many contexts (Lafreniere et al., 2021). It is telling that at DP, we currently detect software mentions solely using regular expressions – this is unsatisfactory because of its poor performance but we can be sure that this approach is fast and without external dependencies.

We can make a distinction between three kinds of external software dependencies: (a) open source libraries/components/etc.; (b) closed source variants thereof; and (c) external web services, e.g. RESTful APIs (Fielding et al., 2017).

Unsurprisingly, open-source software run locally allows users to carefully examine their inner workings, adjust it to specialized needs or even fix bugs or inconsistencies. This is usually not possible for closed source software. However, usually for reasons of interoperability, it has become very common to implement software tools as micro- or web services (Pahl and Jamshidi, 2016). As long as they are run locally, they fall into the same two categories as above. However, when a service is only available through the web because they are run by an external service provider, they are outside of the control of the user of the tool.

This dependency without control is not without problems. For one, if DP were to offer a service that needs to communicate with a third-party service over the web, this potentially impacts DP's reliability (e.g.when the third-party server is down) as well as its performance (owing to the performance of the external service as well as to network connectivity issues). In addition, it is very hard if not impossible to detect if the functionality of the external server has changed in any way.

Finally, all three options of reliance on external software require trust. Integrating any new code into a customer-facing system potentially opens the door to new security vulnerabilities. Again, this is a point that research prototypes often do not have to worry about.

We now turn to discussing how NSLP is affected by these three issues, either as part of the challenge or as part of a possible solution.

4. The role of NSLP

The three issue above were chosen because of their capacity to demonstrate that (a) NSLP has the potential to remedy some of the outlined shortcomings but (b) at the same time may itself be subject to the same concerns. But its role is different in each of the three points.

4.1. Reliance on external software and services

The problem outlined in the previous section regarding the reliance on external software is not specific to NSLP, but it affects research software in general. It can be stated with some confidence that both the providers of research software as well as the potential consumers have a heightened interest in bringing state-of-the-art approaches into production. But from the perspective of an end-user such as DP, depending on externally developed tools, is not without risks.

What can both sides contribute in order to remedy this issue?

If researchers have an interest in impacting real-world applications with their software developments, they need to first be aware of issues like the ones laid out in the paper. Second, and consequently, any software developed as part of a research project should consider adopting professional software development standards.

In practice, this is often easier said than done. Even with the best of intentions, software is usually not the primary concern of research projects, and the additional time, effort, and cost to push them toward industry standards need to be justified. In

addition, professional software development may neither be the main interest nor core expertise of researchers.

Similarly, end-users like DP need to spend additional efforts if they want to make use of research software. In order to mitigate the risks introduced by any external software components, they regularly employ certain strategies. For instance, to address the problem that external REST services may be down at unpredictable times, appropriate fall-back solutions should be implemented. To encapsulate possible security problems, sandboxing solutions might be an appropriate countermeasure (Maass et al., 2016).

Note, however, that these and other measurements possibly induce an additional cost for development and maintenance which, depending on their magnitude, may pose an insurmountable hurdle in the adoption of research results.

4.2. Privacy considerations

State-of-the-art natural language processing systems often rely on large language models (Qin et al., 2026). However, the frontier LLM are for the most part only accessibly through commercial web-based services. On-premise hosting offers a potential remedy, but many state-of-the-art models are not available for this approach. Besides, the technical and financial implications of self-hosting LLMs can be challenging for a relatively small publisher like DP. Therefore, NSLP in particular faces challenges when its processing relies on such services for processing supplementary material such as datasets or software, as questions of privacy and data protection arise.

Further, in case of sensitive data, tight consent management might be a requirement that adds substantial efforts and costs.

At the same time, NSLP could potentially be part of solutions to specific sub-problems. For instance, automatic analyses of provided materials could provide invaluable assistance – for instance in determining potential privacy violations before they occur, i.e. before publication.

Likewise, automatic anonymization techniques, e.g. based on *k-anonymity* (Sweeney, 2002) or *differential privacy* (Dwork and Roth, 2014) could provide valuable support in meeting privacy concerns.

4.3. The metadata externality problem

The biggest potential for NSLP in all of the three challenges discussed in this paper might lie in addressing the metadata externality problem, i.e. the discrepancy of wanting rich metadata as a user vs. the burden of providing them as an author.

Automatically or semi-automatically detecting and providing metadata from and for both articles and supplementary material (cf. e.g. Yang et al. (2025)) would be a great help for both sides of the externality problem. Let us consider this issue in more detail using the example from before.

In Section 3.1, we sketched the problem of software mentions in research articles when they appear not as part of structured data but simply in the text of the document. NSLP offers a possibility to support the creation of rich metadata by extracting software mentions from unstructured natural language texts automatically.

The 3rd International Workshop on Natural Scientific Language Processing (NSLP 2026) features a Shared Task on *software mention detection and coreference resolution*[8].

In the context of the automated acquisition of metadata, this task is of immediate relevance. At least two situations come to mind in which it could find direct application:

- When authors upload an article that contains software as a supplementary material, automatically detected references could support the authors when asked to provide metadata about their own work.

- When authors reference third party software, i.e., software that will not be included in the publication as an artifact in its own right, it could still constitute relevant metadata that should be encoded as such.

In the simplest case, a positive detection result by a (semi-)automatic approach could serve as a reminder to the authors to provide the respective metadata at all. Ideally, however, the NSLP tool would go a step further and also pre-classify which type of mention it found: own software or third-party packages.

Further, it would be ideal to not just detect the words that constitute a software mention but to extract relevant information in a structured way, such as, e.g.: the *name* of the referenced software, its *version number*, the *software license*, the underlying *platform/programming language*, etc.

Naturally, not all of these and other points of information will always be mentioned in an article. But recognizing and extracting the available information in such a structured way would be a tremendous help for metadata providers or, possibly, even pave the way for fully automated metadata extraction.

[8]SOMD 2026, https://nfdi4ds.github.io/nslp2026/docs/somd_shared_task.html

5. Conclusion

This paper examines the potential of NSLP in scientific publishing in three concrete cases and analyzes some of the challenges involved. These challenges sometimes overlap with those encountered by researchers, but in other cases they are complementary or distinct.

Our goal here is to raise awareness of some of the difficulties involved in bringing state-of-the-art research results into production. The discussion here is intended as an encouragement to provide relevant tools for natural scientific language processing but also wishes to give a perspective that may not always be prominently featured within the research community.

6. Bibliographical References

DCMI. 2012. Dublin core metadata element set, version 1.1. ISO 15836. ISO standard for resource description.

Cynthia Dwork and Aaron Roth. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407.

Mark D. Wilkinson et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Sci Data*, 3(160018).

Roy T. Fielding, Richard N. Taylor, Justin R. Erenkrantz, Michael Martin Gorlick, Jim Whitehead, Rohit Khare, and Peyman Oreizy. 2017. Reflections on the REST architectural style and "principled design of the modern web architecture" (impact paper award). In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017, Paderborn, Germany, September 4-8, 2017*, pages 4–14. ACM.

GDPR. 2016. Regulation 2016/679 of the european parliament and of the council (general data protection regulation). https://eur-lex.europa.eu/eli/reg/2016/679/oj. OJ L 119.

Ben Lafreniere, Tanya R. Jonker, Stephanie Santosa, Mark Parent, Michael Glueck, Tovi Grossman, Hrvoje Benko, and Daniel Wigdor. 2021. False positives vs. false negatives: The effects of recovery time and cognitive costs on input error preference. In *UIST '21: The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 54–68. Https://doi.org/10.1145/3472749.3474735.

Mugeng Liu, Xiaolong Huang, Wei He, Yibing Xie, Jie M. Zhang, Xiang Jing, Zhenpeng Chen, and Yun Ma. 2024. Research artifacts in software engineering publications: Status and trends. *Journal of Systems and Software*, 213:112032.

Michael Maass, Adam Sales, Benjamin Chung, and Joshua Sunshine. 2016. A systematic analysis of the science of sandboxing. *PeerJ Computer Science*, 2(e43).

NISO. 2019. Ansi/niso z39.96-2019: Journal article tag suite (jats). ANSI/NISO Standard.

Claus Pahl and Pooyan Jamshidi. 2016. Microservices: A systematic mapping study. In *6th International Conference on Cloud Computing and Services Science*, pages 137–146.

Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S. Yu. 2026. Large language models meet nlp: a survey. *Frontiers of Computer Science*, 20(2011361).

Latanya Sweeney. 2002. k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570.

Ana Trisovic, Matthew K. Lau, Thomas Pasquier, and Mercè Crosas. 2022. A large-scale study on research code quality and execution. *Scientific Data*, 9(60).

Supatsara Wattanakriengkrai, Bodin Chinthanet, Hideaki Hata, Raula Gaikovina Kula, Christoph Treude, Jin Guo, and Kenichi Matsumoto. 2022. Github repositories with links to academic papers: Public access, traceability, and evolution. *Journal of Systems and Software*, 183:111117.

Wenli Yang, Rui Fu, Muhammad Bilal Amin, and Byeong Kang. 2025. The impact of modern ai in metadata management. *Human-Centric Intelligent Systems*, 5:323–350.

Siqi Zhou, Lukas Brunke, Allen Tao, Adam W. Hall, Federico Pizarro Bejarano, Jacopo Panerati, and Angela P. Schoellig. 2024. What is the impact of releasing code with publications?: Statistics from the machine learning, robotics, and control communities. *IEEE Control Systems*, 44(4):38–46.