# Towards Transparency and Knowledge Exchange in AI-assisted Data Analysis Code Generation

Robert Haase[1,2,✉]

[1]Data Science Center, Leipzig University, Humboldtstraße 25, 04105 Leipzig, Germany
[2]Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden / Leipzig

**The integration of Large Language Models (LLMs) in scientific research presents both opportunities and challenges for life scientists. Key challenges include ensuring transparency in AI-generated content and facilitating efficient knowledge exchange among researchers. These issues arise from the in-transparent nature of AI-driven code generation and the informal sharing of AI insights, which may hinder reproducibility and collaboration. This paper introduces git-bob, an innovative AI-assistant designed to address these challenges by fostering an interactive and transparent collaboration platform within GitHub. By enabling seamless dialogue between humans and AI, git-bob ensures that AI contributions are transparent and verifiable. Moreover, it supports collaborative knowledge exchange, enhancing the interdisciplinary dialogue necessary for cutting-edge life sciences research. The open-source nature of git-bob further promotes accessibility and customization, positioning it as a vital tool in employing LLMs responsibly and effectively within scientific communities.**

## Introduction

Generative artificial intelligence (AI) and Large Language Models (LLMs) in particular are changing the way we do data science. Most prominently, scientists use the technology for interacting with scientific data (1), answer data analysis questions (2, 3), generate data analysis code (4–6), and [re-]writing scientific manuscripts (7). Unfortunately, the prompts sent to LLMs are commonly not conserved, hindering knowledge exchange between scientists on how to use the technology efficiently and responsibly. At the time of publication of scientific code and manuscripts, it might be hard to identify the parts of the project which were implemented by a human and the parts that were created by AI. A professional peer-review system, for documenting how LLM-written code was prompted for, and which human reviewed it, is not established in contemporary scientific culture. However, such systems do exist for collaborative code editing involving multiple humans. E.g. the github.com online platform is well-established in the open-source software community for discussing issues and potential solutions, building code together, and for peer-reviewing code and documentation, before they are published to a wider community. As it was shown before that LLMs can solve real-world GitHub issues (8), developing an AI-assistant that interacts with humans directly on the Github platform is the obvious next step. In this manuscript, I am presenting git-bob, a functional proof-of-concept implementation of an LLM-based AI-assistant that can respond to GitHub issues, discuss potential solutions with humans iteratively, write code for them, and submit it as pull-request to be reviewed by humans. It is technically similar to various online services for data analysis such as the OpenAI ChatGPT Data Analyst or Github Copilot Workflows, with three major differences: 1) Multiple humans can interact with git-bob in one communication thread. This allows bringing together domain specialists, e.g. a life scientist, data-analyst and the AI-assistant in one discussion, stimulating knowledge exchange on how to interact with the AI-assistant. 2) Discussions with git-bob and resulting code-modifications are conserved in an online-platform that others can read and follow, making the interaction with the AI-assistant fully transparent, and 3) git-bob is open-source. Other developers can read its built-in system prompts and modify them to their needs. Git-bobs source code is available online: https://github.com/haesleinhuepf/git-bob.

## Features and limitations

A common workflow involving git-bob is demonstrated in Figure 1: a user opens an issue on a repository on github.com, where git-bob is installed. Github issues are discussion threads commonly used by open source software users and developers for reporting bugs, asking questions and discussing further development. In such a thread, a repository member can trigger git-bob to answer by writing a command such as "git-bob comment on this". If externals try so, an automatic response will inform them that only repository members are allowed to trigger git-bob because running git-bob may cause costs for them. Once triggered, git-bob will respond to the question, potentially including a code snippet and resulting plots or images. Users and the AI-assistant can then discuss back and forth until some potential solution is reached. Optionally, git-bob can then be asked to implement the solution and send a GitHub pull-request, another kind of discussion thread, but accompanied by file modifications to the repository, e.g. including a Jupyter Notebook containing the breviously discussed code solution to a given issue. A human would need to review this pull-request and merge it into the code base of the repository. Git-bob also has the capability to review pull-requests, e.g. originating from humans,

but it is not allowed to merge them. This reflects established practices in science, where eventually a human is responsible for data analysis code that becomes part of the project. Additional tasks git-bob is capable of are: 1) The assistant can support users of open source libraries by providing advice and code examples, as shown in Supplementary Figure S1. In case the assistant is not sure about the answer, it is capable of forwarding the question to a human (Supplementary Figure S2). 2) It can be used to document code (Supplementary Figure S4). Such a task can be time-consuming when performed without AI-assistance, which can generate documentation for multiple Python functions in seconds to minutes. 3) It can not just write but also execute data analysis code, e.g. to summarize and plot CSV files which are stored in a repository (Supplementary Figure S3). 4) If manuscript files are stored in a github repository, e.g. in latex format, git-bob can assist in writing. For example, the abstract for this manuscript was written by the AI-assistant and this is documented transparently as shown in Supplementary Figure S5.

When writing code for data analysis or manuscript text, the assistant is intrinsically limited by the capabilities of the used LLM. For example, it has been shown before that common commercial state-of-the-art LLMs can solve bio-image analysis questions by generating functionally correct code just above $50\%$ of tested cases ([5]). This fundamental limitation may disappear when improved LLMs are published. For now, it can be evaded in multi-turn interactions between humans and AI. Yet, we humans need to guide the AI towards a workable solution. Further technical limitations are file size of files to be read and written, as prompt input and output lengths are limited with common LLMs. Also when processing data, technical limitations of the GitHub IT infrastructure have to be considered. git-bob executed in public repositories is per default executed on virtual machines with 4 CPU cores, 16 GB of RAM and 14 GB of SSD storage. In private repositories, only 2 CPU cores and 7 GB RAM are available. More capable systems are available on a paid basis ([9]).

A highlight of git-bob is that a local installation or an institutional internet server are not required. Git-bob is implemented as GitHub workflow, and hence runs within the IT infrastructure of github.com. It is compatible with GPT4-omni, other OpenAI LLMs, Anthropic's Claude, Google Gemini, models hosted on Github Models Marketplace and other models which use the OpenAI application programming interface (API). Git-bob reports which model was used in all of its messages, as good scientific practice suggests. The first three of the mentioned vendors require payment for using their services through the API, the others may offer usage of their API for free. Obviously, the communication with the selected LLM is transmitted to the service provider, including source code files from the repository and images provided with the github issue. Hence, users are recommended to not submit any personal or sensitive information.

Using git-bob and also LLMs in general does not eliminate the need for expertise in the domain we are working in. For example, when tasked to setup a bio-image analysis workflow, one of the humans interacting with git-bob should have the expertise to judge if a proposed solution is reasonable at least. When LLMs improve, especially their reasoning capabilities, this limitation may feel less and less relevant. However, when we analyse data in research projects, we commonly do not know what's the correct result. AI can assist in writing code for this and also for semi-automated quality assurance. Eventually, a human may still be required who has the right domain expertise to judge if a workflow is producing reasonable results and quality assurance works as expected. Git-bob can be used in private repositories giving scientists the necessary privacy to work on projects before they eventually publish their work. Above mentioned abstract writing for this manuscript was performed while the repository was private.

## Conclusion

LLMs are being integrated in contemporary scientific workflows unavoidably, but documentation of how in detail they are employed is commonly not done, also because of lack of tools allowing this conveniently. If the scientific community documented usage of LLM prompts like they document usage of open source data analysis software, we could learn from each other how to prompt efficiently and responsibly. To overcome current limitations, I propose git-bob, a functional, LLM-based proof-of-concept AI-assistant. It works integrated on github.com enabling scientists to interact with an LLM via Github Issues and Pull-Requests offering new ways for implementing good scientific practice for the documentation of discussions between humans and AI when they are working on projects together.

## References

1. Loïc A. Royer. The future of bioimage analysis: a dialog between mind and machine. *Nature Methods*, 20(7):951–952, 2023. ISSN 1548-7105. doi: 10.1038/s41592-023-01930-y.
2. Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Scott Wen tau Yih, Daniel Fried, Sida Wang, and Tao Yu. Ds-1000: A natural and reliable benchmark for data science code generation, 2022.
3. Wei Lei, Cristina Fuster-Barceló, Georg Reder, et al. Bioimage.io chatbot: a community-driven ai assistant for integrative computational bioimaging. *Nature Methods*, 21:1368–1370, 2024. doi: 10.1038/s41592-024-02370-y.
4. Loic A. Royer. Omega – harnessing the power of large language models for bioimage analysis. https://doi.org/10.5281/zenodo.8240289, 2023.
5. Robert Haase, Christian Tischer, Jean-Karim Hériché, and Nico Scherf. Benchmarking large language models for bio-image analysis code generation. *bioRxiv*, 2024. doi: 10.1101/2024.04.19.590278.
6. Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, et al. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021.
7. Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.
8. Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues?, 2024.
9. GitHub. About github-hosted runners, 2024. Accessed: 2024-10-14.

**Fig. 1.** Use-case example for the interaction with git-bob: After creating a GitHub issue, optionally including upload of an example image, the AI-assistant and the human exchange about how to process the data (A). Once they concluded to implement the solution, a GitHub pull-request is sent (B) containing the programmed solution as files in the repository. Also here, human and AI-assistant can continue discussing and modify the solution until the example is satisfactory (C). The discussion also contains error messages observed while the AI-assistant was executing code or creating files, allowing to modify prompts or improve git-bob itself. The entire discussion and corresponding code can be read online: https://github.com/haesleinhuepf/git-bob-playground/issues/13 and https://github.com/haesleinhuepf/git-bob-playground/pull/14
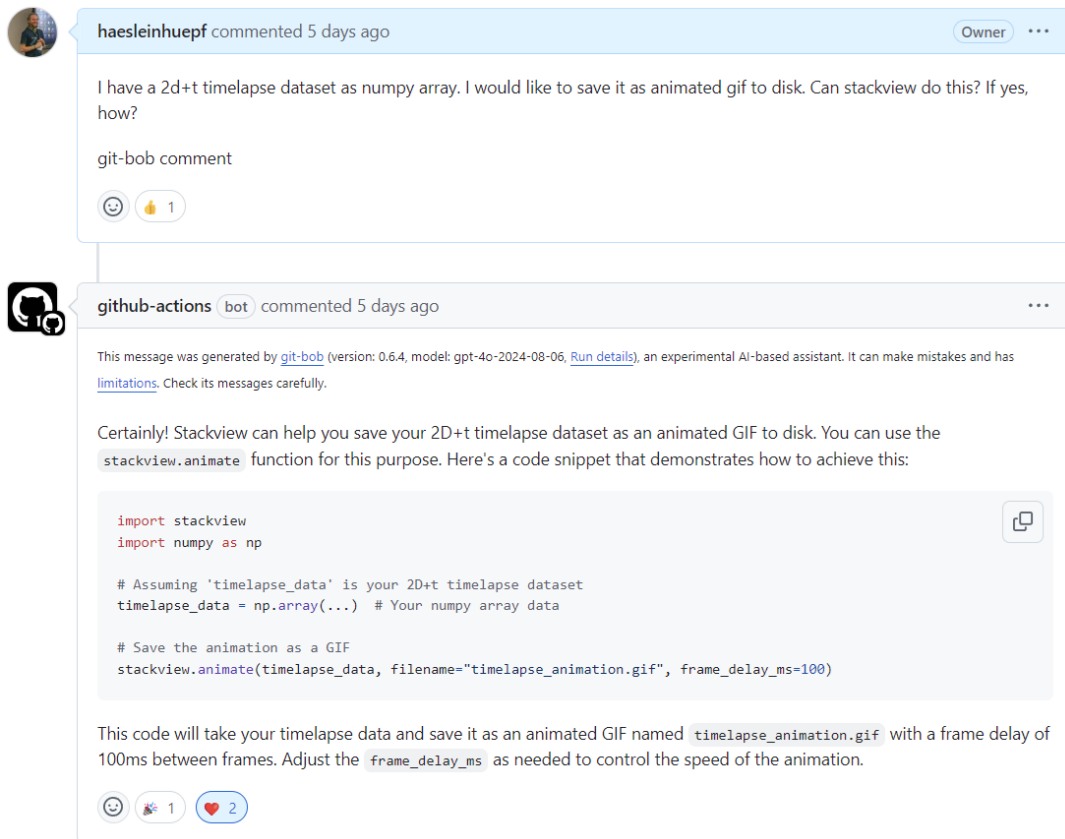
# Supplementary material

**Fig. S1.** Use-case example for supporting users: The assistant can be configured to act as expert on a specific Python library and answer user questions. The entire discussion and corresponding code can be read online: https://github.com/haesleinhuepf/stackview/issues/79
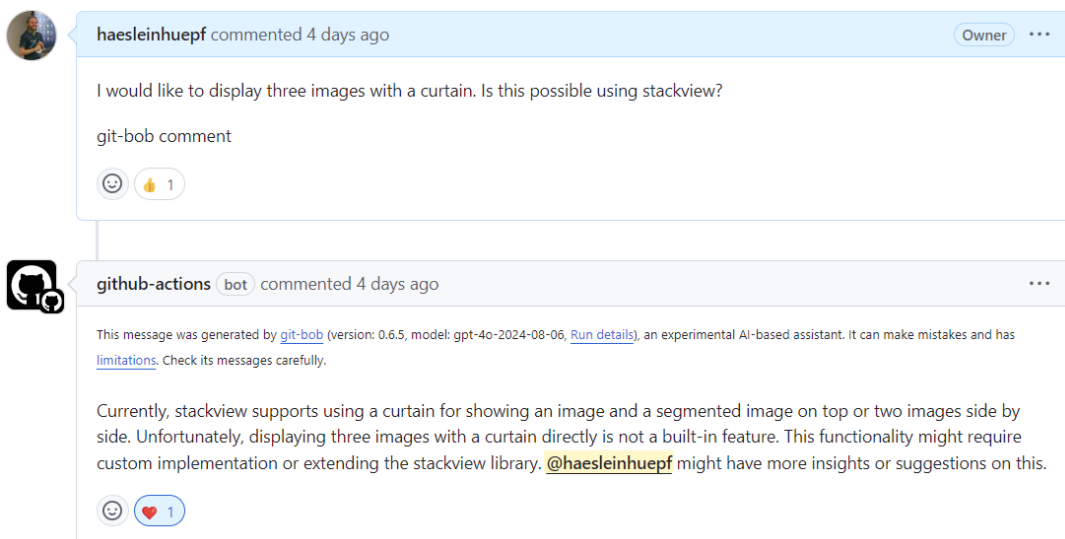


**Fig. S2.** Use-case example for asking an expert: The answer to the question shown here is "No", but this is nowhere written in the documentation or the configuration of the assistant. In this case the assistant is not sure, and it can be configured to forward a question to a maintainer of the library where the question arrived. The entire discussion and corresponding code can be read online: https://github.com/haesleinhuepf/stackview/issues/80

**haesleinhuepf** commented 3 minutes ago      Member   • • •

In the folder "download_statistics", there are multiple files named for example "20240628.csv". The filename without ending is obviously a date, the creation date of the data. Write a jupyter notebook that opens all these files using pandas and plots over time how many records are in these files. Save the notebook in the notebooks directory and make sure the relative link the download_statistics folder is correct.

git-bob try again

😊   👍 1

---

**github-actions** (bot) commented 2 minutes ago      Contributor   • • •

This message was generated by git-bob (version: 0.8.2, model: github_models:gpt-4o, log), an experimental AI-based assistant. It can make mistakes and has limitations. Check its messages carefully.

In this update, a Jupyter notebook was created to read CSV files and plot the number of records over time, resulting in the creation of notebooks/plot_zenodo_links_over_time.ipynb. Additionally, an image file was generated:
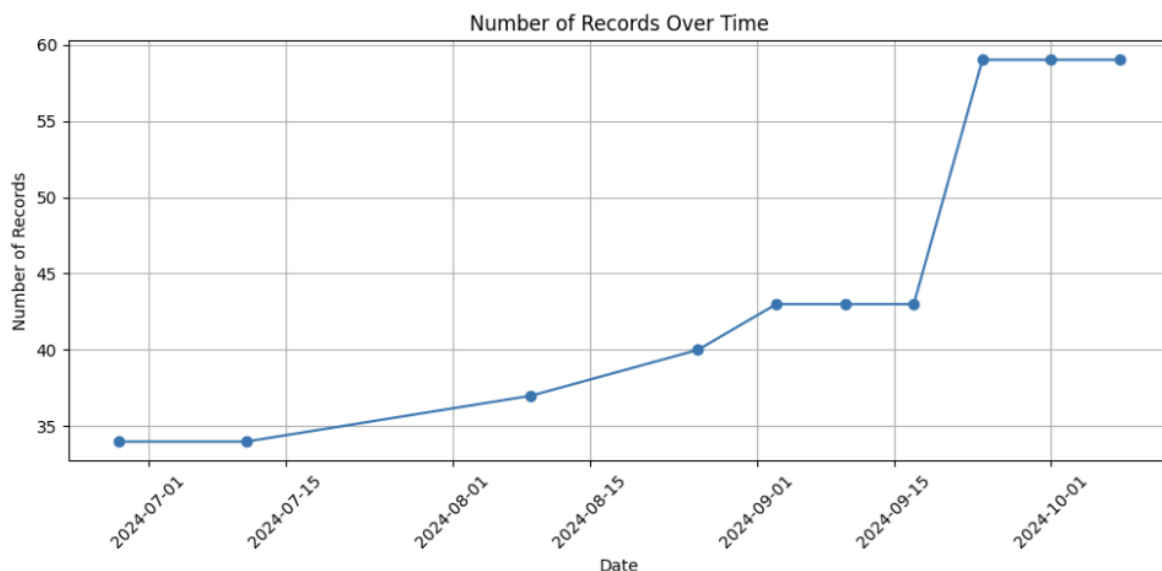


😊

**Fig. S3.** Use-case example for plotting data: after explaining the assistant the folder structure of the project, it generates code for parsing a folder of CSV files and plotting results. The entire discussion and corresponding code can be read online: https://github.com/NFDI4BIOIMAGE/training/issues/250

## A)

# Add numpy-style docstrings to _github_utilities.py #29

**Merged**  **haesleinhuepf** merged 1 commit into `main` from `mod-ajh4CjX8wR` ⧉ on Aug 2

| Conversation 0 | Commits 1 | Checks 0 | Files changed 1 |

**github-actions** `bot` commented on Aug 2                                    `Contributor` ···

This comment was generated by git-bob, an AI-based assistant.

Added numpy-style docstrings for all functions in `src/git_bob/_github_utilities.py` . Closes #22.

☺

○─  Update file content                                                    d123ac1

⑂  haesleinhuepf merged commit **dafe0ec** into `main` on Aug 2              [ Revert ]

## B)

| Conversation 0 | Commits 1 | Checks 0 | **Files changed 1** |

Changes from **all commits** ▾    File filter ▾    Conversations ▾    Jump to ▾   ⚙ ▾          🤖 Ask Copilot    **Review** ▾

∨  ⊕  78 ■■■■□  src/git_bob/_github_utilities.py ⧉

                                                                    ☐ Viewed  💬  ···

```
       @@ -6,6 +6,18 @@
 6   6    from github import Github
 7   7
 8   8    def add_comment_to_issue(repository, issue, comment):
     9  +      """
    10  +      Add a comment to a specific GitHub issue.
    11  +
    12  +      Parameters
    13  +      ----------
    14  +      repository : str
    15  +          The full name of the GitHub repository (e.g., "username/repo-name").
    16  +      issue : int
    17  +          The issue number to add a comment to.
    18  +      comment : str
    19  +          The comment text to add to the issue.
    20  +      """
 9  21        print(f"-> add_comment_to_issue({repository}, {issue}, ...)")
10  22
11  23        access_token = os.getenv('GITHUB_API_KEY')
```

**Fig. S4.** Use-case example for adding and revising documentation in code: git-bob was used to partially write the code documentation of its own code. When asked to add documentation in a specific format, it sent a pull-request (A) and the human could inspect the code modifications (B, excerpt) before mergin the code into the project's code base. The entire discussion and corresponding code can be read online: https://github.com/haesleinhuepf/git-bob/pull/29

A)

## Add abstract #8

⊙ Open · haesleinhuepf opened this issue 5 minutes ago · 2 comments

**haesleinhuepf** commented 5 minutes ago · · ·

We need an abstract for the manuscript in the git-bob-manuscript.tex file. The manuscript will be submitted to a journal where life scientists are the target audience. I'd like to have a abstract focusing on the scientific challenges (transparency and knowledge exchange) when using LLMs. git-bob should be mentioned in the second half of the abstract.

git-bob think about this and answer in plain text with no markdown or tex styling.

😊 👍 1

**github-actions** bot commented 4 minutes ago · · ·

This message was generated by git-bob (version: 0.9.0, model: gpt-4o-2024-08-06, log), an experimental AI-based assistant. It can make mistakes and has limitations. Check its messages carefully.

An abstract focusing on the scientific challenges of using LLMs in life sciences is provided below:

The integration of Large Language Models (LLMs) in scientific research presents both opportunities and challenges for life scientists. Key challenges include ensuring transparency in AI-generated content and facilitating efficient knowledge exchange among researchers. These issues arise from the opaque nature of AI-driven code generation and the informal sharing of AI insights, which may hinder reproducibility and collaboration. This paper introduces git-bob, an innovative AI-assistant designed to address these challenges by fostering an interactive and transparent collaboration platform within GitHub. By enabling seamless dialogue between humans and AI, git-bob ensures that AI contributions are transparent and verifiable. Moreover, it supports collaborative knowledge exchange, enhancing the interdisciplinary dialogue necessary for cutting-edge life sciences research. The open-source nature of git-bob further promotes accessibility and customization, positioning it as a vital tool in employing LLMs responsibly and effectively within scientific communities.

😊

**haesleinhuepf** commented 2 minutes ago   Author · · ·

Awesome! Add this abstract to the manuscript!

git-bob implement this.

😊 👍 1

🔗 ⊙ github-actions bot mentioned this issue 1 minute ago

Add abstract to manuscript highlighting key challenges and git-bob solution. #9

⊙ Open

B)

## Add abstract to manuscript highlighting key challenges and git-bob solution. #9

⊙ Open

github-actions wants to merge 1 commit into `main` from `git-bob-mod-FWWdOZPtHW` ⧉

Conversation 0 | Commits 1 | Checks 0 | Files changed 1

**github-actions** bot commented 2 minutes ago · · ·

This message was generated by git-bob (version: 0.9.0, model: gpt-4o-2024-08-06, log), an experimental AI-based assistant. It can make mistakes and has limitations. Check its messages carefully.

Added an abstract focusing on the scientific challenges of using LLMs in life sciences to the git-bob-manuscript.tex file, addressing issue #8 by highlighting transparency, knowledge exchange, and introducing git-bob as a solution.

closes #8

😊

⊙ 🐙 I added an abstract to the manuscript.   8b4a46a

C)

Conversation 0 | Commits 1 | Checks 0 | Files changed 1

Changes from all commits ▾   File filter ▾   Conversations ▾   🤖 Ask Copilot   Review ▾
Jump to ▾   ⚙ ▾

∨ ⭥ 2 ■■◻◻◻ git-bob-manuscript.tex ⧉

☐ Viewed   💬 · · ·

```
              @@ -102,7 +102,7 @@
102   102
103   103     \begin{abstract}
104   104
105         - Abstract will be added later
      105   + The integration of Large Language Models (LLMs) in scientific
              research presents both opportunities and challenges for life
              scientists. Key challenges include ensuring transparency in AI-
              generated content and facilitating efficient knowledge exchange
              among researchers. These issues arise from the opaque nature of
              AI-driven code generation and the informal sharing of AI
              insights, which may hinder reproducibility and collaboration.
              This paper introduces git-bob, an innovative AI-assistant
              designed to address these challenges by fostering an interactive
              and transparent collaboration platform within GitHub. By
              enabling seamless dialogue between humans and AI, git-bob
              ensures that AI contributions are transparent and verifiable.
              Moreover, it supports collaborative knowledge exchange,
              enhancing the interdisciplinary dialogue necessary for cutting-
              edge life sciences research. The open-source nature of git-bob
              further promotes accessibility and customization, positioning it
              as a vital tool in employing LLMs responsibly and effectively
              within scientific communities.
106   106
107   107     \end{abstract}
108   108
```

**Fig. S5.**   Use-case example for working on scientific manuscripts: after a first draft of the manuscript was written, git-bob was asked to formulate an abstract (A). The abstract was then submitted as pull-request with a short description (B). The human can also review and potentially modify the proposed text in this online interface (C). The entire discussion can be read online: https://github.com/haesleinhuepf/git-bob-manuscript/issues/8 and https://github.com/haesleinhuepf/git-bob-manuscript/pull/9.