The Journal of Open Source Software

PyPaperRetriever: A Python Tool for Finding and Downloading Scientific Literature

Joseph I Turner [1,2] **and Kaydance D Turner**[3]

1 NYU Grossman School of Medicine **2** Center for Brain Circuit Therapeutics, Harvard Medical School **3** Department of Computer Science, Brigham Young University

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

PyPaperRetriever is a Python tool that automates the discovery and retrieval of scientific literature, starting from either a DOI or PubMed ID. It queries multiple sources—Unpaywall, NIH Entrez/PMC, and Crossref—to locate and download lawfully available PDFs, prioritizing open access. The tool supports programmatic PubMed searches with custom queries, enabling at-scale retrieval where lawfully available, metadata resolution, and optional figure extraction in a single, reproducible workflow. Usable via command line or as a Python module, PyPaperRetriever provides a scalable, PubMed-first solution for literature reviews, dataset creation, and other projects requiring comprehensive literature aggregation.

Legal/Ethical Notice: This article documents software that prioritizes lawfully available sources (e.g., PubMed Central, Unpaywall-indexed OA, and publisher-permitted copies). It does not instruct readers to access or download materials from unauthorized sources. Any references to third-party websites that may host unauthorized copies are provided only for contextual completeness and are not endorsed. Readers are solely responsible for compliance with applicable laws and institutional policies.

Core Features

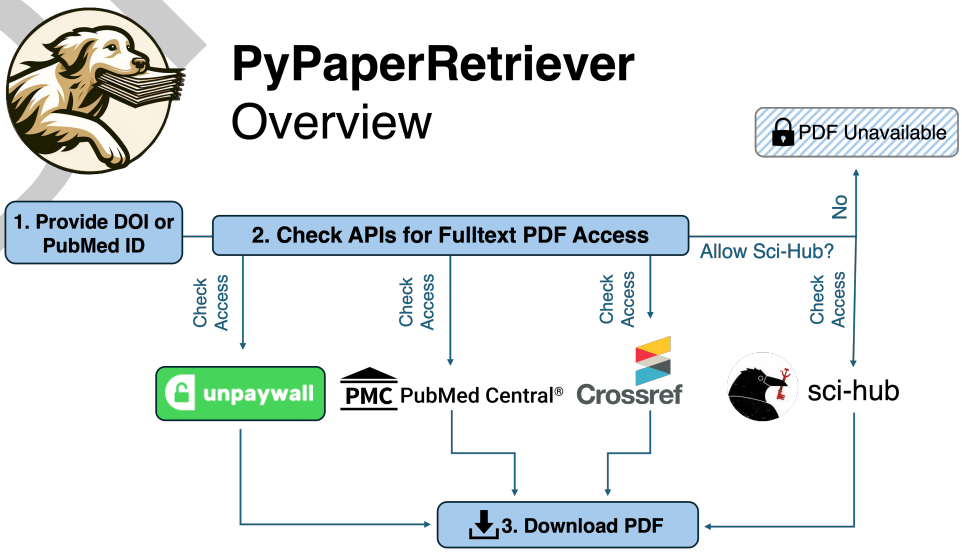

Figure 1: PyPaperRetriever paper retrieval workflow.

The Journal of Open Source Software

1. Find and Download PDFs for a Given DOI or PubMed ID: PyPaperRetriever can find and download PDFs for a given DOI or PubMed ID by querying multiple APIs, including Unpaywall, NIH's Entrez, and Crossref. If a PDF is lawfully available (e.g., via PubMed Central or publisher-permitted OA), PyPaperRetriever will download it to the user's system.

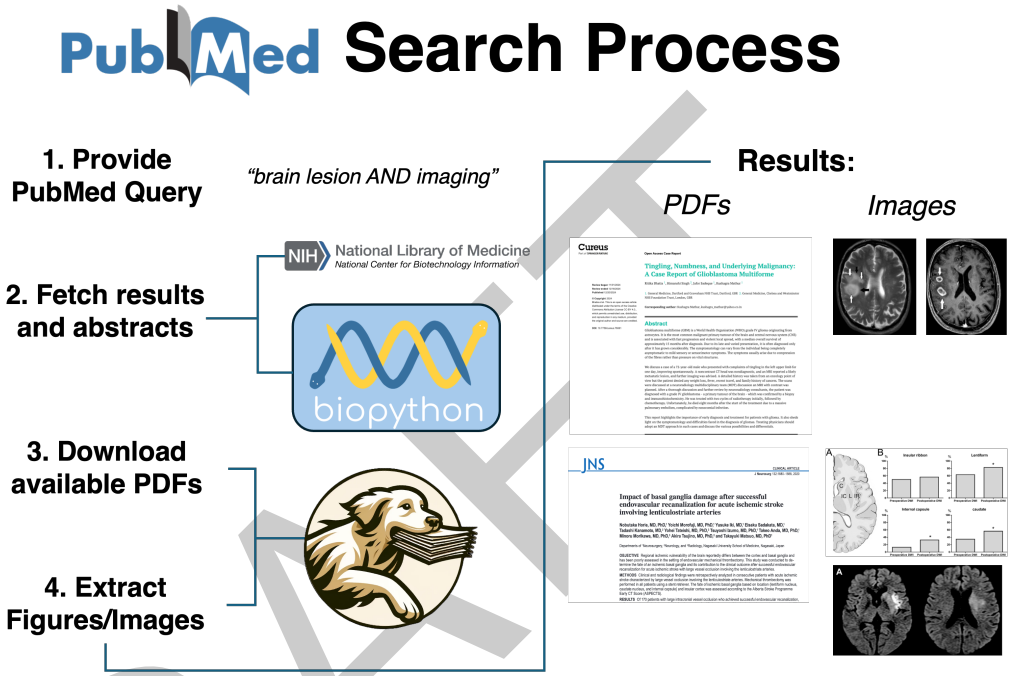

Figure 2: PubMedSearcher workflow for searching PubMed programmatically.

2. Search PubMed programmatically: PyPaperRetriever can search PubMed using a query string and download PDFs of the search results where lawfully available (e.g., PubMed Central or publisher-provided OA). This feature is useful for researchers who want to download multiple papers based on a specific topic or keyword.

3. Extract figures/images from PDFs: PyPaperRetriever can extract figures and images from PDFs and save them as PNG files. This feature is useful for researchers who want to extract and analyze figures or images from scientific papers, and is robust to a variety of PDF formats. PyPaperRetriever has been used to build an extensive catalog of brain lesion images for training computer vision models.

4. Finding references, and building citation networks: PyPaperRetriever allows users to track a paper's citation network using a DOI or PubMedID. It finds papers that reference the given paper, as well as the papers it references. This process can be repeated recursively to any desired depth, enabling users to explore how the paper has influenced others and the foundational work it is built upon across multiple generations of citations.

Statement of Need

Efficient, reproducible access to full texts at scale remains a bottleneck for systematic reviews, dataset construction, and downstream AI/LLM workflows. Manually locating and downloading PDFs is tedious and time-consuming(Singh et al., 2011; Zhang et al., 2023), even with recent advances in open access (OA) and indexes like Unpaywall. Biomedical researchers often begin with PubMed queries rather than DOI lists, and converting those results into full-text PDFs

involves several steps: resolving metadata from search results, handling newly posted green OA, validating links, and downloading large batches reliably. Without a dedicated tool, most labs resort to ad-hoc glue code and custom scripts that are hard to maintain.

PyPaperRetriever automates this process. It provides a PubMed-first, end-to-end pipeline that: (i) executes complex PubMed searches; (ii) resolves and downloads full text from multiple sources (prioritizing OA sources such as PubMed Central and publisher-permitted copies); (iii) supports downstream analyses such as figure extraction and citation graphs; and (iv) integrates cleanly into Python workflows.

The tool has been extensively used in research, including aggregation of over 7,000 brain lesion case reports for LesionBank.org and several thousand additional downloads for related neuroscience projects. By streamlining literature retrieval and integrating advanced search capabilities, PyPaperRetriever saves substantial time in projects requiring comprehensive literature aggregation, and it has been successfully embedded into PRISMA pipelines for systematic review workflows.

Looking ahead, the ability to leverage Large Language Models (LLMs) for screening full text, extracting detailed insights, and mining not only abstracts but also full-text content, figures, and images represents a critical step for the next generation of AI-driven literature tools(Oami et al., 2024; Scherbakov et al., 2024). PyPaperRetriever provides the foundational infrastructure for these advancements, ensuring researchers can access and process the literature at scale.

Methods

PyPaperRetriever is a Python-based tool designed to search, retrieve, and analyze scientific papers using a structured, object-oriented approach. The primary class, PaperRetriever, serves as the central interface and can be used both via the command line and as an importable module for integration into custom Python scripts or Jupyter notebooks. Supporting classes—PubMedSearcher, ImageExtractor, PaperTracker, and ReferenceRetriever—extend its capabilities, allowing for enhanced paper searching, citation tracking, and figure extraction.

Object-Oriented Structure

The software is structured around the following classes:

- **`PaperRetriever`:** The core class responsible for retrieving scientific papers. It supports searching for papers using DOIs or PubMed IDs and attempts to download them using multiple external sources. This class is importable from the PyPaperRetriever module and can also be executed via the command line.
- **`PubMedSearcher`:** Facilitates keyword-based searching of PubMed, retrieving metadata, and assembling structured datasets of search results.
- **`ImageExtractor`:** Extracts images and figures from downloaded PDFs, handling both native and image-based PDFs.
- **`ReferenceRetriever`**: Gathers references and citations for a given paper using multiple external APIs.
- **`PaperTracker`:** Builds citation networks by tracing references upstream (papers cited by the target paper) and downstream (papers that cite the target paper), storing results in structured DataFrames.

Command-Line vs. Programmatic Usage

- **Command-Line Interface (CLI)**: The PaperRetriever class can be executed directly from the command line using the main() function, allowing users to quickly retrieve papers without writing additional code.

- **Python Module Import**: While PaperRetriever can be used standalone, the supporting classes (PubMedSearcher, ImageExtractor, PaperTracker, and ReferenceRetriever) are intended for use in Python scripts and Jupyter notebooks, providing more flexibility for data analysis and automation.

Similar Tools

Several tools exist for finding and downloading scientific literature, but PyPaperRetriever stands out due to its versatility and robust integration with multiple APIs. Here, we compare PyPaperRetriever with similar tools to highlight its advantages:

1. Unpaywall

Unpaywall is a widely used open-access index for DOIs, offering a REST API that identifies locations of freely available PDFs on the web. It's integrated into many reference managers and academic tools, and even has an R client for programmatic access. While Unpaywall is excellent for finding OA versions of papers, it is DOI-centric and does not support PubMed ID–based searches; nor does it provide a complete end-to-end solution for downloading PDFs at scale.

PyPaperRetriever builds on Unpaywall by handling the entire workflow, from running PubMed searches, to resolving mixed PMID/DOI metadata, to downloading PDFs, with additional queries to NIH's Entrez (PubMed/PMC) and Crossref to improve coverage. In practice, we've found that Unpaywall's API can occasionally miss newly deposited green OA copies or point to PDFs that are hosted in ways that block automated retrieval. By cross-checking multiple sources and validating links, PyPaperRetriever improves recall and reliability, helping ensure that open-access copies outside Unpaywall's current index are found and retrieved.

2. PyPaperBot

PyPaperBot (Ferrulli, 2020), while functional, has significant limitations that prompted the development of PyPaperRetriever. PyPaperBot relies primarily on Sci-Hub, which is ethically controversial, may be unlawful to use in many jurisdictions, and is often blocked by academic institutions and in certain countries. Additionally, it lacks support for PubMed ID-based searches, a critical feature for researchers in biomedical sciences.

PyPaperRetriever addresses these shortcomings through several key improvements. It integrates with three different APIs (Unpaywall, NIH's Entrez, and Crossref) to expand access to a wide range of sources, and prioritizes open-access sources. The tool also supports PubMed ID searches and programmatic PubMed queries, while enabling module-level imports for integration into Python workflows, unlike PyPaperBot's command-line-only functionality.

3. Proprietary Software

There are several proprietary software tools for managing scientific literature, including DistillerSR and Convidence. These often come with high costs and limited flexibility. PyPaperRetriever offers a free, open-source alternative with comparable functionality.

Ethical and legal note on Sci-Hub

Use of third-party sites that may host unauthorized copies (e.g., Sci-Hub) can be unlawful in many jurisdictions, and institutions commonly block such access. This article does not provide instructions for accessing or downloading materials from unauthorized sources. PyPaperRetriever prioritizes lawfully available sources (e.g., PubMed Central, Unpaywall-indexed OA, and publisher-permitted copies). The authors do not endorse or encourage any unlawful use; users are responsible for ensuring compliance with applicable laws, licenses, and institutional policies.

Availability

All code and documentation for PyPaperRetriever are available on GitHub. The tool is distributed under the MIT License, allowing for free use, modification, and redistribution. Instructions for installation and usage are provided in the README file. We welcome contributions and feedback from the community to improve the tool and expand its capabilities. Opening an issue on the GitHub repository is the best way to report bugs or request features; pull requests are also welcome for contributions and will be reviewed promptly.

Acknowledgements

JIT conceived the idea for PyPaperRetriever, developed the codebase, and wrote documentation. JIT is the primary author of this paper and takes full responsibility for the content. KDT also provided feedback on the tool's design and functionality, contributed to the documentation, and assisted in testing and debugging.

References

Ferrulli, V. (2020). PyPaperBot: A tool to automatically download scientific papers. In *GitHub repository*. GitHub. https://github.com/ferru97/PyPaperBot

Oami, T., Okada, Y., & Nakada, T. A. (2024). Performance of a Large Language Model in Screening Citations. *JAMA Network Open*, *7*(7), e2420496. https://doi.org/10.1001/jamanetworkopen.2024.20496

Scherbakov, D., Hubig, N., Jansari, V., Bakumenko, A., & Lenert, L. A. (2024). *The emergence of Large Language Models (LLM) as a tool in literature reviews: an LLM automated systematic review*. https://doi.org/10.48550/arXiv.2409.04600

Singh, A., Singh, M., Singh, A. K., Singh, D., Singh, P., & Sharma, A. (2011). "Free full text articles": where to search for them? *International Journal of Trichology*, *3*(2), 75–79. https://doi.org/10.4103/0974-7753.90803

Zhang, M., Doi, L., Awua, J., Asare, H., & Stenhouse, R. (2023). Challenges and possible solutions for accessing scholarly literature among medical and nursing professionals and students in low-and-middle income countries: A systematic review. *Nurse Education Today*, *123*, 105737. https://doi.org/10.1016/j.nedt.2023.105737