The Journal of Open Source Software

eyecite: A tool for parsing legal citations

Jack Cushman[1], **Matthew Dahl**[2], **and Michael Lissner**[3]

1 Harvard University, Library Innovation Lab **2** University of Notre Dame, Department of Political Science **3** Free Law Project

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

License
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

Summary

`eyecite` is a Python package for high-performance extraction of legal citations from text. It can recognize a wide variety of citations commonly appearing in American legal decisions, including:

- full case (e.g., `Bush v. Gore, 531 U.S. 98, 99-100 (2000)`)
- short case (e.g., `531 U.S., at 99`)
- statutory (e.g., `Mass. Gen. Laws ch. 1, § 2`)
- law journal (e.g., `1 Minn. L. Rev. 1`)
- supra (e.g., `Bush, supra, at 100`)
- id (e.g., `Id., at 101`)

It also offers tools for pre-processing citation-laden text, aggregating like citations, and annotating citations with custom markup.

Statement of need

Citations are the bedrock of legal writing and a frequent topic of legal research, but few open-source tools exist for extracting them from legal texts. Because of this, researchers have historically relied on proprietary citation data provided by vendors like LexisNexis and Westlaw (e.g., Black & Spriggs, 2013; Fowler et al., 2007; Spriggs & Hansford, 2000) or have used their own personal scripts to parse such data from texts ad hoc (e.g., Clark & Lauderdale, 2012; Fowler & Jeon, 2008). By providing an open-source, standardized alternative to these approaches, `eyecite` promises to increase scholarly transparency and consistency. It also promises to give researchers the extendability and flexibility to develop new methods of citation analysis that are currently not possible under the prevailing approaches.

For example, one burgeoning research agenda seeks to apply machine learning techniques to citation analysis, either to recommend relevant authorities to legal practitioners (Ho et al., Forthcoming), model the topography of the legal search space (Dadgostari et al., 2021; Leibon et al., 2018), or automatically detect and label the semantic purpose of citations in texts (Sadeghian et al., 2018). One obvious application of `eyecite` would be to use it to generate empirical training data for these kinds of machine learning tasks.

To facilitate those kinds of projects and more, `eyecite` exposes significant entity metadata to the user. For case citations, `eyecite` parses and exposes information regarding a citation's textual position, year, normalized reporter, normalized court, volume, page, pincite page, and accompanying parenthetical text, as well as `eyecite`'s best guess at the names of the plaintiff and defendant of the cited case. For statutory citations, `eyecite` parses and exposes information regarding a citation's textual position, year, normalized reporter, chapter, section, publisher, and accompanying parenthetical text.

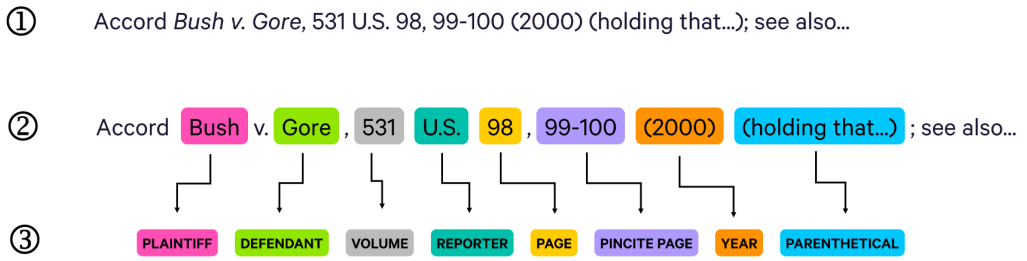

Figure 1: In step (1), `eyecite` consumes raw, cleaned text. In step (2), it parses the text into discrete tokens using Hyperscan and its regular expression database. In step (3), it extracts meaningful metadata from those tokens, returning a unified object for each parsed citation.

Recognizing that researchers often want to parse many documents and citations at once, `eyecite` is designed with an eye toward performance: it makes use of the Hyperscan library (Wang et al., 2019) to tokenize and parse its input text in a highly efficient fashion.[1] It performs this parsing process using a database of thousands of regular expressions that have been built from nearly every citation format found in the collections of the Caselaw Access Project and CourtListener, the Cardiff Index to Legal Abbreviations, and the LexisNexis and Westlaw databases. Additionally, because researchers are often working with imperfect text (perhaps obtained via optical character recognition), `eyecite` provides tools for pre-processing and cleaning its input text. Figure 1 depicts `eyecite`'s extraction process of a full case citation at a high level.

`eyecite` offers other tools as well. It can heuristically resolve short case, supra, and id citations to their appropriate full case antecedents, and it integrates well with custom resolution logic. For practical applications, it can also "annotate" found citations with custom markup (like HTML links) and re-insert that markup into the appropriate place in the original text. This works even if the original text was pre-processed, as `eyecite` uses the diff-match-patch library (Google, 2006) to intelligently reconcile differences between the original text and the cleaned text.

State of the field

To the best of our knowledge, no open-source software offering the same functionality as `eyecite` exists. Other similar packages are either no longer maintained or lack the robust parsing, resolution, or annotation features of `eyecite` (e.g., LexPredict, 2021; Sherred, 2021; Tauberer, 2017). `eyecite` also benefits from being used in production by two public data projects, the Caselaw Access Project and CourtListener, to process and analyze millions of documents in their collections. From these applications, `eyecite` has honed a test suite of real-world citation strings. To further minimize unexpected errors, its codebase enjoys static type checking for all of its functions. At least one study has already used an earlier version of the data generated by `eyecite`'s underlying code (Carmichael et al., 2017).

References

Black, R. C., & Spriggs, J. F., II. (2013). The Citation and Depreciation of U.S. Supreme Court Precedent. *Journal of Empirical Legal Studies*, *10*(2), 325–358.

Carmichael, I., Wudel, J., Kim, M., & Jushchuk, J. (2017). Examining the Evolution of Legal Precedent Through Citation Network Analysis. *North Carolina Law Review*, *96*(1),

[1] We estimate that `eyecite` can parse typical legal text on the order of approximately 10MB/second, though this depends on the density of citations within the text.

227–269.

Clark, T. S., & Lauderdale, B. E. (2012). The Genealogy of Law. *Political Analysis*, *20*(3), 329–350.

Dadgostari, F., Guim, M., Beling, P. A., Livermore, M. A., & Rockmore, D. N. (2021). Modeling law search as prediction. *Artificial Intelligence and Law*, *29*, 3–34.

Fowler, J. H., & Jeon, S. (2008). The authority of Supreme Court precedent. *Social Networks*, *30*(1), 16–30.

Fowler, J. H., Johnson, T. R., Spriggs, J. F., II, Jeon, S., & Wahlbeck, P. J. (2007). Network Analysis and the Law: Measuring the Legal Importance of Precedents at the U.S. Supreme Court. *Political Analysis*, *15*(3), 324–346.

Google. (2006). Diff Match Patch. In *GitHub repository*. GitHub. https://github.com/google/diff-match-patch

Ho, D. E., Huang, Z., Low, C., Teng, M., Zhang, H., Krass, M., & Grabmair, M. (Forthcoming). Context-Aware Legal Citation Recommendation Using Deep Learning. *Proceeedings of the 18th International Conference on Artificial Intelligence and Law*.

Leibon, G., Livermore, M., Harder, R., Riddell, A., & Rockmore, D. (2018). Bending the law: Geometric tools for quantifying influence in the multinetwork of legal opinions. *Artificial Intelligence and Law*, *26*, 145–167.

LexPredict. (2021). LexNLP: Information retrieval and extraction for real, unstructured legal text. In *GitHub repository*. GitHub. https://github.com/LexPredict/lexpredict-lexnlp

Sadeghian, A., Sundaram, L., Zhe Wang, D., Hamilton, W. F., Branting, K., & Pfeifer, C. (2018). Automatic semantic edge labeling over legal citation graphs. *Artificial Intelligence and Law*, *26*, 127–144.

Sherred, S. R. (2021). CiteURL. In *GitHub repository*. GitHub. https://github.com/raindrum/citeurl

Spriggs, J. F., II, & Hansford, T. G. (2000). Measuring Legal Change: The Reliability and Validity of Shepard's Citations. *Political Research Quarterly*, *53*(2), 327–341.

Tauberer, J. (2017). Citation. In *GitHub repository*. GitHub. https://github.com/unitedstates/citation

Wang, X., Hong, Y., Chang, H., Park, K., Langdale, G., Hu, J., & Zhu, H. (2019). Hyperscan: A Fast Multi-pattern Regex Matcher for Modern CPUs. *Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation*.