Systematic literature review: References extraction helper and automatic analysis

Benjamin Vignau, Patrice Clemente, Pascal Berthomé

Systematic Literature Review : references extraction helper and automatic analysis

Benjamin Vignau, Patrice Clemente, Pascal Berthomé, INSA Centre Val de Loire, 88 Boulevard Lahitolle, 18 000, Bourges, benjamin.vignau@insa-cvl.fr)

Abstract
This article describes a software suite designed for systematic and automatic literature reviews. Actually, we used these tools to help us build our systematic literature review on our research topic: biometric authentication using PPG (photoplethysmography). However, our tools can be easily applied (and have been applied) to other fields, whether scientific (state of the art) or technological (technology watch). Our software suite is made of two softwares. The first one helps to extract, merge and filter all the references in the PDF file versions of the papers. The second one helps to make statistical analysis on the dataset created for the SLR.

Keywords
Systematic literature review, biometrics, data science, python, software suite, state of the art.

Code metadata

Nr.	Code metadata description	Please fill in this column
C1	Current code version	*1.0*
C2	Permanent link to code/repository used for this code version	https://github.com/bvignau/SI_PPG_SLR ;
C3	Permanent link to Reproducible Capsule	https://codeocean.com/capsule/6876009/tree/v1
C4	Legal Code License	*MIT*
C5	Code versioning system used	*GIT*
C6	Software code languages, tools, and services used	*python, pandas, ruby, jupyter*
C7	Compilation requirements, operating environments & dependencies	bibtexparser, PyMuPDF, jellyfish, pandas, numpy
C8	If available Link to developer documentation/manual	N/A
C9	Support email for questions	benjamin.vignau@insa-cvl.fr

1. Introduction

To help us in our literature reviews following backward snowball methods [5], we have developed two lightweight tools. Our first tool organizes the SLR and automates certain tedious parts. In particular, it automates the phases of extracting references from a set of papers in PDF format. Our second tool is a collection of Jupyter Notebooks for automatically processing a dataset generated from SLR analysis on quantitative and qualitative criteria chosen according to the field studied.

2. SLRIA

Our first tool, a SLR helper called SLRIA (https://github.com/bvignau/SLRIA/tree/main), is a command line interface (CLI). Following Kitchenham's guidelines [5] for the creation of corpus of studies. It was developed to help us with the tedious tasks imposed by the methodology of systematic literature reviews (SLR). These tasks are: classifying and merging duplicate papers in search query results ; extracting references from all PDF papers ; merging duplicates sources from extracted references ; producing a csv with the title, authors, year, journal and number of appearances of each paper in queries for round 0 (cf. 2-Next rounds), and citations for

subsequent rounds ; organizing folders to keep a trace of the papers ; organizing the reading of papers according to the number of keywords contained in the title and the number of hits ; and creating the final bib tex file with all references. The workflow of our software is given in Figure 1

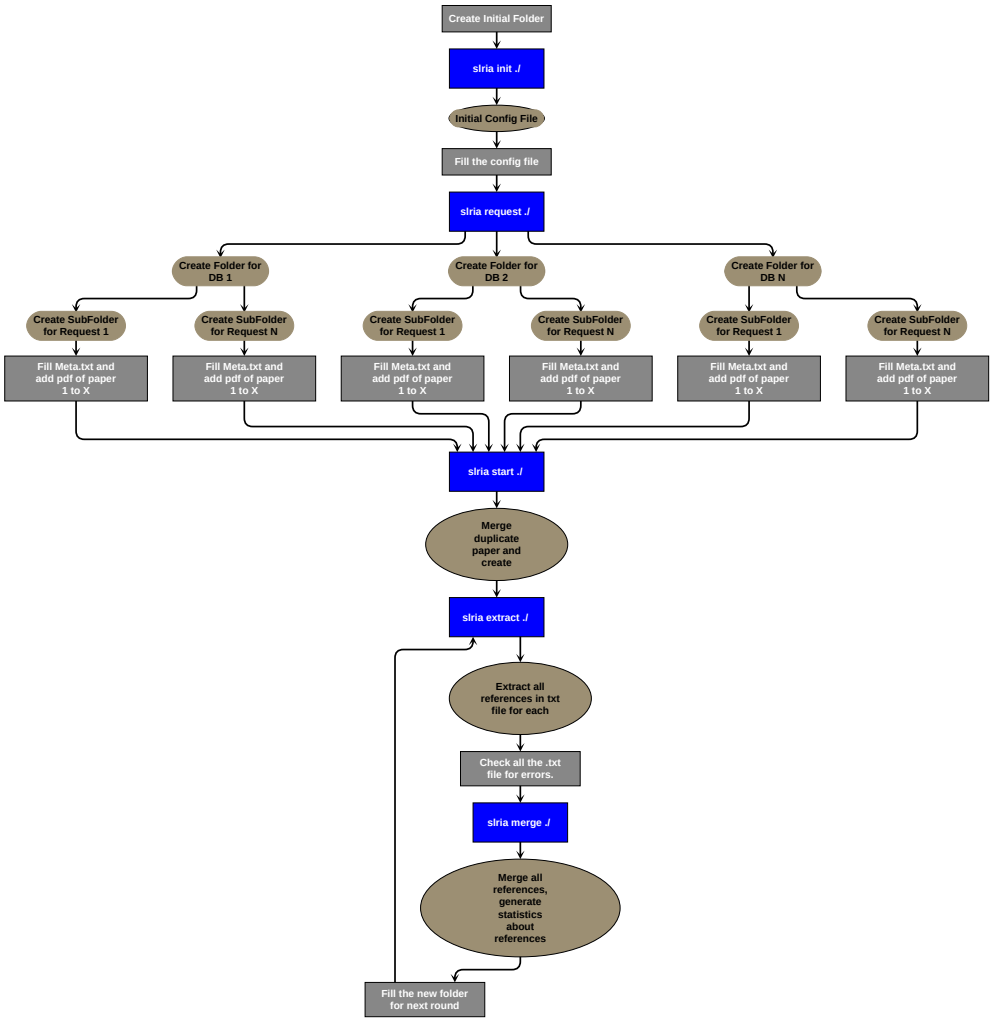

Figure 1: Flow chart of the SLRIA software. Blue squares are the command of the SLRIA software, grey square are manual task that the user have to do and grey circle are the output produced by SLRIA commands.

Initialization. The first command, allows you to initialize a new review by creating a dedicated folder with a configuration file containing the fields to fill out.

You now need to fill in the configuration file with the set of queries you wish to make, the databases you have chosen, and the number of papers you wish to analyze per query.

Organizing folders. The command "`slria request ./`" initializes the organization of queries in a folder. It creates a folder per database, then a subfolder per query. For each request, X meta.txt files are created, where X is the number of papers that we choose to keep for each request. Each meta.txt file contains the title, authors and year for a given paper. You must then carry out searches in the databases, and fill all the meta.txt files with the data from the papers.

Once all the meta.txt files are completed, you can use the "`slria start ./`" command to generate statistics on all the papers, as well as a folder per paper. You must then download and store each PDF in the appropriate folder. If a paper does not correspond to the study, you can delete its folder.

Raw extracting of references. The next step uses the "`slria extract ./`" command to extract all references from all PDFs. This step will produce two txt files per paper. The first contains the entire text of the paper, and the second its references. You have to check for the reference extraction, sometimes errors occurs

due to format of the references or because the PDF is a scan of paper and does not contains raw text.

Merging references. Finally, the last step consists of using the "`slria merge ./`" command to merge all the extracted references, keeping only the most relevant ones. A file is created for each new paper to study, relatively to `kwords` specified in the config file. Papers that don't have the minimum number (`nkwords` parameter) of keywords in their title will be eliminated. To merge the references, we use a comparison score with a threshold. For each references, we extracted a maximum of fields such as title, authors, years, journal etc. But due to differences in references format, sometimes fields are missing. To match the titles between two references, we use the *damerau-levenshtein* distance [10] with a score of less than 5. We add this flexibility to be resilient to the invisible characters in PDF files and errors due to reference extraction.

Final selection of references. Once the first snowball extraction has been completed (called "round 0"), you must select the new papers to be added to the corpus.

Next rounds. As said previously, our system was designed following Kitchenham's guidelines [5] in order to create an initial corpus of studies, and to perform backward snowballing if needed. A backward snowballing phase corresponds to the extraction and analysis of all the references present in the papers of a corpus, in order to add the most relevant ones and thus extend the corpus. In our system, each phase in which references are added is called "round N", with $N > 0$. Our keyword filtering system takes into account only the title of the references, in order to filter out the least relevant. So, we recommend you to read the abstracts of the studies, and possibly the introduction and conclusion to choose the final papers. You must then perform a new round, processing only new studies added to the corpus. If new references are relevant, you add them to the corpus and run a new round. Otherwise, corpus creation stops and you can proceed with the detailed analysis of all studies.

3. Automated analysis

Our second software (https://github.com/bvignau/PPG_SLR_dataset) is a collection of Jupyter notebooks, used to analyze the dataset we have collected with our systematic literature review.

The main objective of our software is then to be able to automatically analyze the collected dataset, compute statistics and generate a report that will give a better understanding of the techniques, data and results of the experiments carried out by the community.

3.1 Generals statistics on single variable

For all statistical studies on a single qualitative variable (validation technique employed, use of different noise filtering and database selection, etc.) we always calculate usage as a function of years and experiments (number of experiments using each technique), in absolute terms and as a percentage. All graphs and calculation codes are available.

For sections with a lot of different categorical information, we offer several chart versions. For example, in our exploration of noise reduction methods for PPG-based biometric systems, we have a figure with all the filters and their parameters, and a condensed figure with only the filter families. The aim here is to make the figures easier to read.

Where numerical values are available, such as the number of features extracted, or performance measurements (EER, accuracy), we provide trends by year and box plots.

3.2 Consumed data and correlation matrix

As described in our main paper [9], we have developed a metric to evaluate the contribution of each dataset to the community. We thus determined the amount of useful data consumed per year, by the community and per dataset. The amount of useful data consumed per year for a dataset corresponds to the following formula $D_{contribution} = N_{subjects} \times T_{aquisition} \times N_{experiences} \times F_{aquisition}$ that defines "data consumption" or "Dataset contribution", as the product of the number of subjects, of the total acquired time, the number of experiences that used the dataset, and the acquisition frequency (see 5.5 in the main paper for more details). We leave the code for computing these values from the dataset and the resulting figures. An example of such a figure is given in Figure 2.

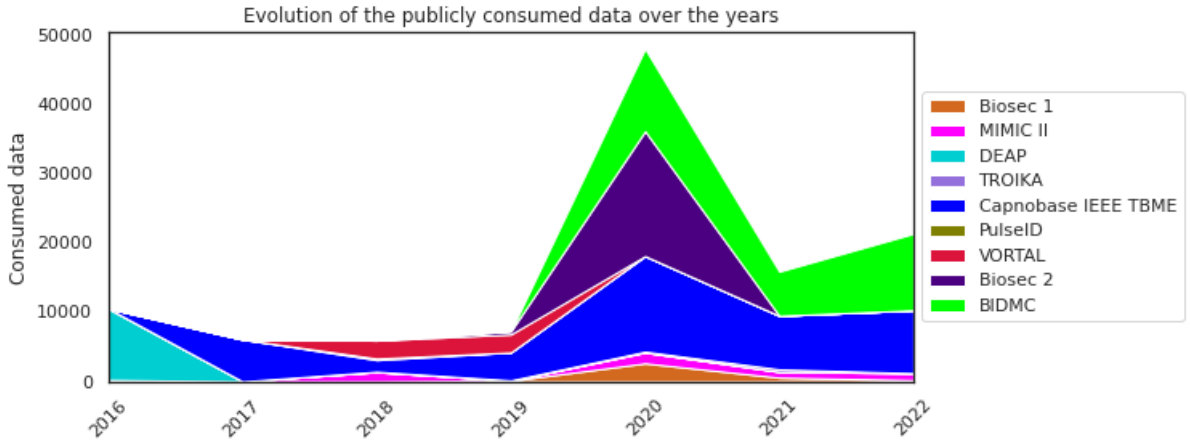

Figure 2: An example of consumed data over the years for the biometric authentication using PPG.

Finally, we wanted to assess the impact of numerical data such as feature size or number of users in a database on system performance. We therefore calculated the pair plots and the correlation matrix. However, for these datasets, the results did not show any trend. The computation methods will be useful in our next version, which will automatically compute all the statistics we need to interpret in our future benchmarks.

4. Impact overview

4.1 Similar tools

To help researchers with the complex task of reviewing the literature, a number of tools exist, providing a variety of functions. These include Google Scholar and Mendeley Library [2], which can be used to create reading lists, organize references, export references in multiples format (bibtex, RIS etc.) and deal with duplicates in the first stage of SLR. A complete description of the features of Scholar have been done by [7]. As stated in their user guide [2], Mendely also lets you store pdf files of papers, and take notes on each study. Thus Mendeley is more advanced than Google Scholar for helping SLR. Recently, the explosion in natural language processing algorithms has led to the emergence of new tools such as ChatGPT and HyperwriteAI.

These tools can effectively summarize papers or provide an initial overview of the state of the art. For example, if we ask hyperwriter AI [1] to provide us with a state of the art on biometric authentication using PPG, it will provide us with a summary of around one page, containing 5 or 6 references to academic studies.

Some add-onsn such as UPDF [3] allow ChatGPT to read and extract information from a pdf. However, processing is limited, and tedious for the user, who must manually process each pdf and the extracted information. Moreover, these add-ons are usually not free of charge.

4.2 Our main contribution

The SLRIA software enabled us to organize our systematic literature [8] [9] review more easily and make it reproducible. It also makes it easier to generate review statistics. Our software is currently being used in our laboratory to speed up other literature reviews.

Our tool enables automated management of all papers in PDF format. Using a single command, you can extract all the references of all the papers to be studied for the snowball phase. Then, all duplicates will be automatically merged, and statistics on the number of appearances of each paper will be calculated. Finally, the correctly formatted bibtex file will be generated.

To improve our tool, we will combine it with an LLM model to enhance reference extraction and offer automatic extraction of certain user-selected data. For example, in our case, we would like to be able to automatically extract quantitative information from studies, such as precision of the algorithms, number of patients, etc.

5. Comparison with other works

As stated in the previous section, some software already implements some of the SLRIA software's features . These include Google Scholar [7] and the Mendeley bookshop [2].

However, these tools are not capable of analysing all online literature automatically. Nor they can't keep track of SLR steps, neither automatically extract a paper's references in PDF format and generate statistics on each step. Nor can they sort the reading order according to the potential interest of each paper.

Regarding our second contribution, our Jupyter Notebook collection allow us to reproduce our analyses and experiments. They will also serve as a basis for future benchmarking software linked to biometric authentication. Finally, these notebooks can be re-used without modification for future literature reviews on any subjects, simply by modifying the columns used so that they correspond to the data extracted from other literature reviews.

Thanks to our SLR software suite, we propose to treat an SLR as a data science problem. We define a broad set of criteria to be analyzed, depending on the subject, and we perform a set of statistical analyses to help understand the state of the art, in an automated way. The aim is to provide an analysis of the evolution of a community through the experiments carried out.

6. Future works

In recent years, machine learning researchers have become increasingly interested in the biases of algorithms [6] . These biases often have their origins in the data used to create the algorithms[4] . With our method, during an SLR, we can extract the data used for each experiment carried out in a domain. We can then use our metric of data consumed to determine the data that has contributed most to the community. If these data are biased, we can then quantify the possible biases in a community, propose a less biased data set and monitor the evolution of the community's productions.

7. Conclusion

In conclusion, we have created two software to help us with a variety of tasks, both tedious and critical in SLR. This can help community to create SLR using the backward snowball method. Our SLRIA software will continue to be developed to facilitate systematic literature reviews, regardless of the topic. In the future, we will merge our two software to provide one unified system to help community to create SLR.

References

[1] HyperWrite — app.hyperwriteai.com. https://app.hyperwriteai.com/personalassistant. [Accessed 28-05-2024].

[2] mendeley.com. https://www.mendeley.com/guides/mendeley-reference-manager/. [Accessed 28-05-2024].

[3] UPDF — updf.com. https://updf.com/fr/pricing-individuals/?source=website_updf&channel=&currency=EUR&language=fr-FR. [Accessed 28-05-2024].

[4] B. Cowgill, F. Dell'Acqua, S. Deng, D. Hsu, N. Verma, and A. Chaintreau. Biased programmers? or biased data? a field experiment in operationalizing ai ethics. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 679–681, 2020.

[5] B. Kitchenham, S. Charters, et al. Guidelines for performing systematic literature reviews in software engineering, 2007.

[6] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6):1–35, 2021.

[7] R. L. Miller. An introduction to google scholar. 2019.

[8] B. Vignau, P. Berthomé, and P. Clemente. Les systèmes d'authentification biométriques cardiaques. Rendez-Vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information, May 2023. Poster.

[9] B. Vignau, P. Clemente, and P. Berthome. The biometric authentication using photoplethysmography (ppg): A twenty years systematic literature review. *Available at SSRN 4730225*, 2023.

[10] C. Zhao and S. Sahni. String correction using the damerau-levenshtein distance. *BMC bioinformatics*, 20:1–28, 2019.