LiFUSO: A Tool to Improve Library Selection in Maven

Camilo Velázquez-Rodríguez*, and Coen De Roover†
Vrije Universiteit Brussel, Belgium
*camilo.ernesto.velazquez.roodriguez@vub.be, †coen.de.roover@vub.be

Libraries in a software ecosystem target a well-defined domain (e.g., PDF, collections, etc.) and offer functionalities to client systems through their API. The instantiation of one or multiple APIs enables the implementation of a particular task within the domain (e.g., inserting an image, serialising to JSON, reversing a list, etc.). Maven, NPM or CRAN are examples of software ecosystems which provide a vast number of libraries offering multiple features for their reuse.

At library selection time from a vast ecosystem, it becomes essential for developers to know the enumeration of features offered by each library. Such a possibility is not currently available in existing software ecosystems. The selection of libraries is currently based on popularity metrics (which can be biased by the number of developers who have used the library) and not the features they provide, defining features as a set of API references $n : n >= 1$ and the textual description of the offered functionality [1], [2]. Furthermore, there is currently no way to compare libraries feature-wise, which can ultimately improve their selection.

To alleviate the absence of descriptions of features of libraries, developers often pose questions on Q&A websites like Stack Overflow[1] (SO) to get information about how to realise particular tasks. Therefore, the solutions on sites like SO provide a variety of feature examples that enhance library documentation, enable cross-library comparison by users, and improve recommendation tools of third-party libraries.

We present our tool for Library Feature Unveiling from Stack Overflow (LiFUSO) posts as an initial approach to improving the selection of libraries in the Maven software ecosystem. LiFUSO was developed based on the proposed approach by Velázquez-Rodríguez et al. [3]. The tool takes advantage of the natural language terms around API reference usages to discover features with their corresponding name.

The input of LiFUSO is a target library for which features need to be computed. Our tool requires that the *groupID* and *artifactID* (e.g., `com.google.guava` and `guava` respectively) of at least one version of the library is available from the Maven Central repository. The name of the target library needs also to be a valid SO tag[2].

In this abstract, we briefly describe the main components of LiFUSO and provide a simple example of its usage:

1) **Collecting Information**: The first step collects the names of API elements as well as answers from SO that might contain usages of these elements by downloading all versions of the library from Maven and processing their bytecode. SO answers are obtained from the SOTorrent dataset[3] by searching posts with the name of the library among their tags.
2) **Filtering**: This step first excludes answers without any code snippets. As SO code snippets are not necessarily complete nor syntactically correct, LiFUSO relies on a robust parser generated by a custom-built island grammar to this end. Our custom parser focuses on the syntactic constructs in which method invocations from the target library can occur.
3) **Clustering and Naming**: This step constructs a matrix of the Jaccard similarity between all SO answers based on their usage of the library's API elements. Next, it applies hierarchical clustering to the matrix and by means of a dynamic cut tree technique, selects the optimal cutting point to form clusters. Using the local outlier factor (LOF) technique, the most frequent API elements within the cluster are identified. In case LOF cannot determine such elements, the cluster is discarded. A semantic tree is computed for the sentences in the title of and the text surrounding each SO answer. For each noun and verb, LiFUSO extracts pairs of the form noun-verb or verb-noun. The most frequent verbs and nouns in the pairs (using LOF once again) are retained.
4) **Representing Features**: LiFUSO selects the top-5 most frequent pairs as representations for the name of the feature. In the case of the selection for API references in the clustered snippets, LiFUSO selects those extracted from the LOF technique in a previous step. Clusters with the most frequent name pairs and API references are outputted.
5) **Exploring Features**: Shared features of the libraries are calculated as well as the unique functionalities per pair of libraries. Finally, all the previously described information is displayed in a visual interface for feature exploration and library comparisons feature-wise.

At BENEVOL, we intend to present our tool, the limitations it currently shas, and the future avenues our tool might take.

[1] https://stackoverflow.com
[2] https://stackoverflow.com/tags
[3] November 16th, 2020 version from https://zenodo.org/record/4287411

References

[1] Kanda, Tetsuya and Manabe, Yuki and Ishio, Takashi and Matsushita, Makoto and Inoue, Katsuro. 2013. Semi-Automatically Extracting Features from Source Code of Android Applications. TIS E96.D, 12.
[2] Antoniol, Giuliano and Guéhéneuc, Yann-Gaël. 2005. Feature Identification: A Novel Approach and a Case Study ICSM.
[3] Velázquez-Rodríguez, Camilo and Constantinou, Eleni and De Roover, Coen. 2022. Uncovering Library Features from API Usage on Stack Overflow. SANER.