# Reproducibility instructions for Pollock : A Data Loading Benchmark

The source code and the artifacts required to reproduce the results of the paper "Pollock: A Data Loading Benchmark" [1] can be found at the link https://github.com/HPI-Information-Systems/Pollock. The repository contains the pdf version of the paper.

## 1 PREREQUISITES

We provide docker containers that run different scripts – therefore one system requirement is the use of **docker** and **docker-compose**. Alternatively, to reproduce only the results using the intermediate artifacts provided, it is sufficient to have a Python environment installing the dependencies found in the requirements.txt file,

As a first step, please clone the repository on your machine by running:

```
git clone https://github.com/HPI-Information-Systems/Pollock.git
```

From scratch, the process to reproduce the results of the Pollock benchmark is composed of three steps:

1. Generating the polluted versions of the source file.
2. Loading the polluted files in each system under test.
3. Calculating the Pollock scores with the output files for each system under test.

For convenience, in this repository we already provide the intermediate artifacts necessary to run the Steps 2 and 3, so that our results can be reproduced at different stages. We include the resulting artifacts for 16 different systems:

- 6 csv loading modules: clevercsv, csvcommons, hypoparsr, opencsv, pandas, pycsv, rcsv, univocity
- 4 RDBMS: mariadb, mysql, postgres, sqlite
- 3 spreadsheet systems: libreoffice, spreaddesktop, spreadweb
- A data visualization tool, dataviz

Due to licensing issues, we cannot provide the scripts used in step 2 to benchmark the commercial systems spreaddesktop, spreadweb, and dataviz. However, for these systems, we provide the results of the loading processes.

## 2 Repository Overview

The structure of the repository is the following:

- `open_data_crawl` contains the scripts used to crawl the open data portals of different countries to sample the statistics about file types.
- `survey` contains the csv files used in the paper survey, and their annotations with respect to dialect and pollutions.
- `survey_sample` contains the sample of 100 files used in the paper experiment, with their clean versions and loading parameters.
- `polluted_files` contains the generated polluted files for the Pollock benchmark.
- pollock is the main source folder for the Pollock benchmark: it contains the files necessary to generate the polluted versions of an input file (`polluters_base.py` and `polluters_stdlib.py`) as well as the files with the metrics to evaluate results of data loading.
- `sut` is the source folder that contains the scripts used to benchmark given systems. These scripts can be in bash, python, or heterogeneous format, depending on the specific tool that is under test.
- `results` contains the results of loading both the polluted files and the survey files for each of the systems evaluated. The folder will also contain .csv files that summarize the evaluation results - for each of the systems under test and for all of them together (`aggregate_results_{dataset}.csv`, `global_results_{dataset}.csv`).
- The file `docker-compose.yml` contains a list of the docker images that are used to run the benchmark. The images are built from the Dockerfile files in the sut folder.
- The two files `pollute_main.py` and `evaluate.py` are used to run the pollution of a source file and to evaluate all systems under test that have a folder in results/loading

For convenience, we provide a script named `paper_tables.py`, which is the main script to obtain the tables with the experimental results from the paper (Tables 5, 6 and 7). This script takes as input the resulting csv files obtained from computing the Pollock scores in step 3 and prints the table in the terminal. To run it, from the root directory run:

```
docker-compose up paper-tables
```

We warn that the time measurement may be slightly different from the ones reported in the paper, considering the experiments were repeated and updated since the paper submission. The next sections go into detail about the three different steps to fully reproduce the results of the paper, starting from the generation of the pollution dataset.

## 3 GENERATING POLLUTED FILES

To generate the pollution dataset from the input source file, first delete the polluted_files folder to ensure the dataset is generated from scratch:

```
rm -rf polluted_files/
```

The following command will build the docker image to generate the polluted files:

```
docker-compose up pollution
```

After this step, the set of benchmark files are contained in the folder `polluted_files`. Inside this folder there are three sub-folders: `csv` containing the generated polluted files in the .csv format, `clean` containing the cleaned versions of the generated files, and `parameters` containing JSON files storing the corresponding loading parameters for each file.

## 4 LOADING POLLUTED FILES FOR A GIVEN SYSTEM UNDER TEST

Once the folder `polluted_files/` contains the polluted files, the next step is to run the benchmark for the different systems under test. For example, to benchmark the Pandas systems, users can run:

```
rm -rf results/pycsv/polluted_files/loading/*
docker-compose up pycsv-client
```

At the end of the loading stages, the results will be available in the folder `results/{sut}/polluted_files/loading/`, where `{sut}` stands for a given system under test name. The full benchmark can be run by launching all different docker-compose commands in a sequence. Alternatively, if a unix-based system is used, the following one-liner executes the loading for all SUTs:

```
chmod +x benchmark.sh; ./benchmark.sh
```

Considering that the process may be time-consuming, we already provide the intermediate artifacts of loading for each of the systems in the relative result folders. We note that, for the spreadweb, spreaddesktop, and dataviz systems, due to their commercial nature, we cannot share the scripts or docker images used to run the benchmark, but we provide the results of our experiments in the result folders.

## 5 Pollock score calculation

To finally run the evaluation step and calculate the Pollock scores along with the loading metrics, run the command:

```
docker-compose up evaluate
```

The script outputs the benchmark scores for each of the polluted files as a `csv` file under `results/{sut}/polluted_files/` for each of the systems. Moreover, all results are saved in the file `results/global_results_polluted_files.csv` and aggregated in the file `results/aggregate_results_polluted_files.csv`. The script also outputs the results of the benchmark with the simple and weighted Pollock scores.

As a convenience, to obtain the full experimental tables of the paper, users can run:

```
docker-compose up paper-tables
```

## 6 Run experiments for real world files survey

To run the benchmark using the real-world survey files instead of the polluted files, it is sufficient to change the `.env` file in the project root directory, commenting the first line and uncommenting the second line. The file should have the following content:

```
#DATASET=polluted_files
DATASET=survey_sample
```

If the environment variable `DATASET` is set to survey_sample, all of the previous steps can be repeated, and the Pollock benchmark would use as input the files contained in the corresponding `survey_sample/` folder.

## 7 References

[1] Gerardo Vitagliano, Mazhar Hameed, Lan Jiang, Lucas Reisener, Eugene Wu, and Felix Naumann. Pollock: A Data Loading Benchmark. PVLDB, 16(8): 1870-1882, 2023.