Reproducibility Report for ACM SIGMOD 2021 Paper: "SliceLine: Fast, Linear-Algebra-based Slice Finding for ML Model Debugging"

ILIAS KOTSAMPOUGIOUKOGLOU, National Technical University of Athens, Greece
DIMITRIOS TSOUMAKOS, National Technical University of Athens, Greece

The reproducibility submission by the authors, their assistance in providing help when needed, and the provisioning of free access to required hardware resources have substantially eased the task of reproducing the results of this work. In this report, we describe the steps taken and the results that show the agreement in principle and detail of the reproducibility study to the original findings.

1 INTRODUCTION

In this report, reproducibility results are presented for ACM SIGMOD 2021 paper: "SliceLine: Fast, Linear-Algebra-based Slice Finding for ML Model Debugging" by Svetlana Sagadeeva and Matthias Boehm from Graz University of Technology [1]. The main goal of the paper is to identify (also in a scalable way) top-K slices from a training/test set that have worse accuracy than the whole set in a trained ML model in order to improve its accuracy. Our efforts have shown that the findings of this work can be fully reproduced and validated in different settings and input sets as those described in the original submission.

2 SUBMISSION

From a code perspective, Java, R, and SystemDS DML programming languages are utilized to run the experiments and output the results. There exists a central `runAll.sh` bash script, which can execute all the experiments at once. However, the authors provide and recommend the use of eight (8) individual bash scripts which make up `runAll.sh` in order to monitor and control the execution of each stage in the pipeline:

- `./run1SetupDependencies.sh`; (for installing and setup the dependencies)
- `./run2SetupSystemDS.sh`; (for setup the Apache SystemDS which has been cloned from https://github.com/apache/systemds)
- `./run3DownloadData.sh`; (this stage downloads the data)
- `./run4PrepareLocalData.sh`; (this step prepares data for local – single machine execution)
- `./run5LocalExperiments.sh`; (this executes single-machine experiments)
- `./run6PrepareDistData.sh`; (this stage prepares data for distributed – cluster-based execution)
- `./run7DistExperiments.sh`; (for executing the distributed experiments)
- `./run8PlotResults.sh`; (this plots the results)

The `run5LocalExperiments.sh` script is divided into seven (7) bash scripts which are responsible for executing local (single-node) experiments separately. This facilitates more control of the execution pipeline and better reviewing of the results. The same applies for `run7DistExperiments.sh` which splits into two (2) bash scripts responsible for running the distributed experiments. Furthermore, it is possible to execute R scripts to plot the results. These R scripts have been included into `run8PlotResults.sh`.

From a replicating experiments point of view, the reproducibility code is close to an ideal submission. There is the freedom to choose between running compact parts of the code or inspecting stage-by-stage and monitoring individual results. The Readme file, along with some comments in the code, provides enough detail on how the execution pipeline works. As a minor note, it would

be preferable for some comments in the reproducibility scripts that explain the importance/identity of some of the input parameters in tuning the system and structure of the scripts themselves (as many scripts call others).

A list with a summary of the submission contents is the following:

- github source code repository at https://github.com/apache/systemds, commit 627825c25d5a5938a772a78ce037c57e68611998
- github reproducibility repository with code and scripts at https://github.com/damslab/reproducibility/tree/master/sigmod2021-sliceline-p218
- readme file at URL: https://github.com/damslab/reproducibility/blob/master/sigmod2021-sliceline-p218/README.md

Data sets used in the reviewed paper:

- Adult https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data
- Covtype https://archive.ics.uci.edu/ml/machine-learning-databases/covtype/covtype.data.gz
- KDD'98 https://archive.ics.uci.edu/ml/machine-learning-databases/kddcup98-mld/epsilon_mirror/cup98lrn.zip
- US Census https://archive.ics.uci.edu/ml/machine-learning-databases/census1990-mld/USCensus1990.data.txt
- CriteoD21 http://azuremlsampleexperiments.blob.core.windows.net/criteo/day_21.gz
- Salaries https://forge.scilab.org/index.php/p/rdataset/source/file/master/csv/car/Salaries.csv

3 HARDWARE AND SOFTWARE ENVIRONMENT

Table 1 and Table 2 present the Hardware and Software Environment that is used both in the paper's experimental section and during the reproducibility tests. We thank the authors for providing us access to both systems.

Table 1. Hardware environment of Paper

	Scale-up node	Scale-out cluster (per node)
CPU	2 Xeon Gold 6238	AMD EPYC 7302
cores	56 physical/112 virtual cores	16 physical/32 virtual cores
GHz	2.2-2.5 GHz	3.0-3.3 GHz
RAM at 2.933 GHz balanced	768GB DDR4	128GB DDR4
across memory channels	6	8
Storage (system/home)	2 × 480GB SATA SDDs	2 × 480GB SATA SDDs
Storage (data)	12 × 2 TB SATA SDDs	12 × 2 TB SATA HDDs
Ethernet	2 × 10Gb	2 × 10Gb

Table 2. Software environment of Paper

	Version
Ubuntu	20.04.1
OpenJDK Java	1.8.0_265 with -Xmx600g -Xms600g
Apache Hadoop	2.7.7
Apache Spark	2.4.7
R	4.0.4 with the doMC package
Apache SystemDS	2.0.0

4 REPRODUCIBILITY EVALUATION

4.1 Process

The reproducibility tests are split into two phases: The local experiments, where alpha (the scale-up node) is utilized, and the distributed experiments, where charlie (the Spark/HDFS master) is utilized for submitting distributed jobs.

For each phase, the first step was the configuration and setup for the experiments to run. Most of the time, this was the most tedious task due to the misconfiguration of the system. The authors' intervention was needed in these cases to clarify and give instructions on how to set up the system or how to resolve these problems. Each experiment has been run separately and its result was used to get the corresponding plot/output. The vast majority of the reproduced results were close/same to the ones reported in the published work. In a couple of cases, some deviation was reported. These were communicated with the authors and resolved with their intervention, as they were due either to misconfiguration or bugs in the reproducibility code (and not the code tied with the publication).

4.2 Results

Our first task was to reproduce the results on pruning effectiveness. These are examined in two axes, namely pruning techniques and different datasets for evaluation. About the pruning techniques part, the conclusion we reached is the same as of paper's, obtaining plots that correspond to Figure 3 of the paper and support the authors' findings. For the second part, the datasets which are utilized for evaluating pruning effectiveness are Adult, KDD98, USCensus, and Covtype. The results here are also a strong match with the findings in Figure 4 of the paper.

The second series of experiments refers to the weight parameter α. The datasets which are utilized are Adult, KDD98, USCensus, and Covtype. In this case, too, our findings completely agree with the ones reported in Figure 5 of the original paper: As α increases, scores increase, and sizes get reduced. This affects the slice sizes, which decrease as well.

The last part concerns execution run time for both local and distributed experiments. For local experiments, Adult, KDD98, USCensus, and Covtype have been used, and our results have been able to reproduce Figure 6 of the paper. For distributed experiments we run `runExperiment5b.sh` for validating the results presented in Figure 7(b). Finally, the Criteo dataset is used to execute `runExperiment6.sh` and produces Table 3 (to be compared to Table 2 from the original paper). For those experiments, the produced trends and actual results fully agree with the findings of the paper. It is worth noting that results in running time in distributed setups are hard to replicate precisely, so this is taken into consideration.

It is worth mentioning that the first three experiments (regarding Figures 3, 4, and 5) have also been executed in a local, single-node machine (AMD Ryzen threadripper 1950x, 16-core CPU at

Table 3. Criteo Slice Enumeration Statistics

Lattice level	1(Init)	2	3	4	5	6
Candidates	75573541	1302	1590	4260	8769	13239
Valid Slices	209	666	1570	4260	8769	13239
Elapsed Time	1115	1450	1554	1757	2198	3155

3.4GHz and 32 threads, 128GB RAM, 2TB Disk running Ubuntu 21.10). Our findings confirmed the paper's results except for the USCencus dataset, for which it was impossible to get the final output due to lack of memory.

5 SUMMARY

In the reviewers' opinion, the authors have made considerable efforts to both submit a user-friendly and functional reproducibility package and also provide us with the help and access to the hardware needed. Our findings agreed both in principle and detail with the ones reported in the accepted paper, marking this as a fully reproducible submission.

REFERENCES

[1] Svetlana Sagadeeva and Matthias Boehm. 2021. SliceLine: Fast, Linear-Algebra-based Slice Finding for ML Model Debugging. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*. ACM, 2290–2299. https://doi.org/10.1145/3448016.3457323