Reproducibility Report for ACM SIGMOD 2023 Paper: "DeltaBoost: Gradient Boosting Decision Trees with Efficient Machine Unlearning"

JALAL KHALIL, St. Cloud State University, USA
KAIQIANG YU, Nanyang Technological University, Singapore
HAOTIAN LIU, Sourthern University of Science and Technology, China

We have successfully replicated the findings of the original paper, including the performance results outlined in its experimental section. The GitHub repo was well organized and easy to follow to run the experiments and produce the results. We pose several results we reproduced (using three different platforms) in this report.

1 INTRODUCTION

This report summarizes the reproducibility evaluation for the paper entitled "DeltaBoost: Gradient Boosting Decision Trees with Efficient Machine Unlearning" [1] by Zhaomin Wu, Junhui Zhu, Qinbin Li, and Bingsheng He. The paper appeared in Proceedings of the 2023 ACM SIGMOD International Conference on Management of Data (SIGMOD'23)

The paper focuses on solving the data privacy issues in machine learning (ML) and study how to remove some specific data from ML models to preserve privacy. To achieve this, the authors present a robust machine learning model, namely *DeltaBoost*, for enabling efficient unlearning algorithms on Gradient Boosting Decision Trees, and further develop a training algorithm for the proposed DeltaBoost model.

The core experiments in this paper are successfully reproduced based on the data files and scripts provided by the authors. The main results we reproduced show similar trends as those in the original paper.

2 SUBMISSION

The authors have published the source code on GitHub. It has two options to reproduce the results: 1) a Docker-based master script that automatically downloads the datasets, builds the program, runs the experiments, and saves the results; 2) a step-by-step guide that provides clear details on how to install different components in the right order. We have chosen the Docker-based route.

The GitHub page which contains the source code and the README file is available here: https://github.com/Xtra-Computing/DeltaBoost.

3 HARDWARE AND SOFTWARE ENVIRONMENT

Three different platforms are used for reproducing experiments. We denote them as Platform A (Table 1), Platform B (Table 2) and Platform C (Table 3). In particular, Platform A is the Heterogeneous Accelerated Compute Cluster (HACC) from the School of Computing at the National University of Singapore, which is generously provided by the authors.

4 REPRODUCIBILITY EVALUATION

4.1 Process

The reviewers Kaiqiang Yu and Haotian Liu use docker to run the code, while Jalal Khalil (who uses the HACC cluster where the docker was not installed) uses apptainer[1]

The docker-based reproducing can be done by simply following the README file provided by the authors. Thus, we only provide the steps to reproduce the results using apptainer.

[1]https://apptainer.org/

Table 1. Hardware & Software Environment of Platform A

	Paper	Repro Review
CPU	Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz	AMD EPYC 7V13 @ 3.70GHz
cores	2 x 24 (x2 threads)	2 x 64 (x1 threads)
RAM	380GB	500GB
Storage	2TB SSD	1.7TB SSD

Table 2. Hardware & Software Environment of Platform B

	Paper	Repro Review
CPU	Intel(R) Xeon(R) Gold 6248R	Intel Xeon Gold 6230
cores	24	20
GHz	3.00	2.10
RAM	380GB	128GB

Table 3. Hardware & Software Environment of Platform C

	Paper	Repro Review
CPU	Intel(R) Xeon(R) Gold 6248R	Intel(R) Xeon(R) Gold 5318Y
cores	24	24
GHz	3.00	2.10
RAM	380GB	512GB

- `tmux`
  A command to start a tmux session. Tmux is a terminal multiplexer that allows users to create and manage multiple terminal sessions within a single window.
- `srun -p mi210_vck_u55c --cpus-per-task=32 --pty bash -i`
  Used to allocate an interactive job using Slurm[2]. Slurm is an open-source job scheduler and resource manager widely used in high-performance computing clusters to efficiently allocate and manage computing resources for parallel and distributed computing tasks.
- `git clone https://github.com/Xtra-Computing/DeltaBoost.git`
- `apptainer pull docker://jerrylife/deltaboost`
- `apptainer instance start`
- `apptainer shell instance://deltaboost`
- `cd DeltaBoost`
- `bash run.sh`

It took the system around two days to finish, and the results were saved in *out* and *fig* folders.

[2]https://slurm.schedmd.com/overview.html

Dataset	Metric	Remove 0.1%	Remove 1%
codrna	$H^2(M_r, M; D_{\text{test}})$	0.0002±0.0051	0.1046±0.2984
codrna	$H^2(M_r, M_d; D_{\text{test}})$	0.0000±0.0014	0.0070±0.0515
covtype	$H^2(M_r, M; D_{\text{test}})$	0.0162±0.1260	0.0300±0.1521
covtype	$H^2(M_r, M_d; D_{\text{test}})$	0.0000±0.0005	0.0069±0.0467
gisette	$H^2(M_r, M; D_{\text{test}})$	0.0007±0.0022	0.0070±0.0081
gisette	$H^2(M_r, M_d; D_{\text{test}})$	0.0000±0.0004	0.0051±0.0065
cadata	$H^2(M_r, M; D_{\text{test}})$	0.0058±0.0157	0.0087±0.0113
cadata	$H^2(M_r, M_d; D_{\text{test}})$	0.0034±0.0121	0.0033±0.0048
msd	$H^2(M_r, M; D_{\text{test}})$	0.0041±0.0044	0.0126±0.0101
msd	$H^2(M_r, M_d; D_{\text{test}})$	0.0028±0.0036	0.0093±0.0079

Table 4. Reproduction (excerpt) of Table 5 (DeltaBoost values only) in [1] on Platform A: The forgetfulness $F(M, D_{\text{test}})$ of each approach (single tree).

Thunder	DB-Train	DB-Remove	Speedup (Thunder)
0.602	8.201 ±3.658	0.210 ±0.096	2.87x
0.759	7.978 ±3.416	0.201 ±0.061	3.78x
5.609	80.537 ±2.905	2.677 ±9.285	2.10x
4.408	81.029 ±2.759	4.207 ±13.245	1.05x
44.142	90.619 ±7.235	1.080 ±0.512	40.86x
45.757	90.575 ±6.867	1.262 ±0.518	36.26x
0.654	2.545 ±2.521	0.101 ±0.089	6.50x
0.644	2.216 ±1.709	0.084 ±0.080	7.64x
18.576	72.026 ±2.018	3.550 ±1.489	5.23x
12.069	73.174 ±2.220	3.618 ±1.507	3.34x

Table 5. Reproduction (excerpt) of Table 6 in [1] (only compared with Thunder) on Platform C: Training time and removal time of DeltaBoost

Dataset	Metric	Remove 0.1%	Remove 1%
codrna	$H^2(M_r, M; D_{\text{test}})$	0.0128±0.0099	0.0088±0.0081
codrna	$H^2(M_r, M_d; D_{\text{test}})$	0.0127±0.0099	0.0089±0.0079
covtype	$H^2(M_r, M; D_{\text{test}})$	0.0116±0.0093	0.0118±0.0096
covtype	$H^2(M_r, M_d; D_{\text{test}})$	0.0116±0.0095	0.0118±0.0096
gisette	$H^2(M_r, M; D_{\text{test}})$	0.0136±0.0093	0.0314±0.0175
gisette	$H^2(M_r, M_d; D_{\text{test}})$	0.0136±0.0093	0.0314±0.0169
cadata	$H^2(M_r, M; D_{\text{test}})$	0.0240±0.0167	0.0247±0.0159
cadata	$H^2(M_r, M_d; D_{\text{test}})$	0.0239±0.0154	0.0249±0.0149
msd	$H^2(M_r, M; D_{\text{test}})$	0.0196±0.0107	0.0249±0.0127
msd	$H^2(M_r, M_d; D_{\text{test}})$	0.0196±0.0107	0.0248±0.0126

Table 6. Reproduction (excerpt) of Table 7b (DeltaBoost values only) in [1] on Platform B: The forgetfulness $F(M, D_{\text{test}})$ of each approach (multiple trees).

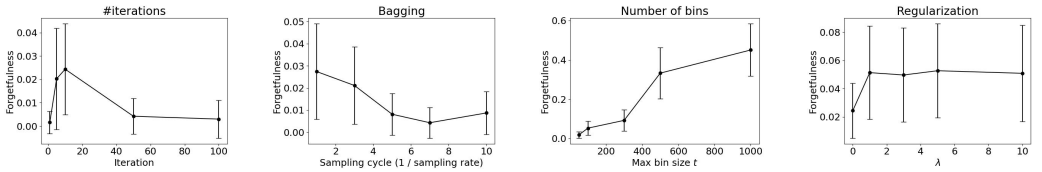

Fig. 1. Reproduction (excerpt) of Figure 10 in [1] Platform A: Effect of different hyperparameters on codrna dataset.

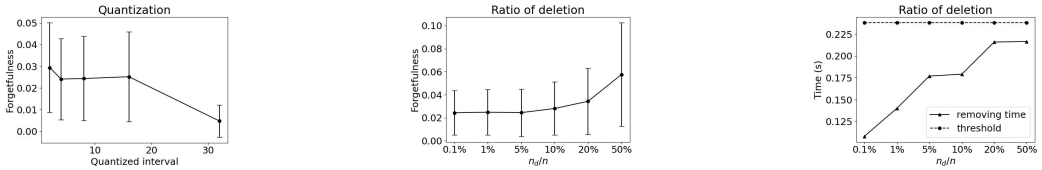

Fig. 2. Reproduction (excerpt) of Figure 11 in [1] on Platform A: Effect of quantization and removal ratio.

4.2 Results

The experiments in the paper evaluated four metrics, including *test error* and *forgetfulness* (see Section 5.3) for the accuracy evaluation, as well as *running time* and *memory usage* for the efficiency evaluation. We can reproduce the core results in the paper by running separate scripts as instructed. Specifically, in our reproduced results, we have the following findings.

- For the accuracy evaluation, we report our reproductions of *Table 5*, *Table 7b*, *Figure 10* and *Figure 11* of the paper [1] in Table 4, Table 6, Figure 1 and Figure 2, respectively. We observe that the reproduced results are similar to those presented in the paper.
- For the efficiency evaluation, we report our reproduction of *Table 6* of the paper [1] in Table 5. We observe that the reproduced running time results are different from those given in the paper, but the relative speedup ratios (between the proposed *DeltaBoost* and baselines) are consistent with those presented in the paper. We believe that the differences potentially come from the distinct hardware and software environments (e.g., CPU and the operating system).

In summary, we can verify that the proposed *DeltaBoost* outperforms other baselines in terms of the above four metrics as stated in the paper. The authors deserve praise for their hard work and careful guidance.

REFERENCES

[1] Zhaomin Wu, Junhui Zhu, Qinbin Li, and Bingsheng He. 2023. DeltaBoost: Gradient Boosting Decision Trees with Efficient Machine Unlearning. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–26.