USENIX'23 Artifact Appendix: HOLMES: Efficient Distribution Testing for Secure Collaborative Learning

Ian Chang
UC Berkeley

Katerina Sotiraki
UC Berkeley

Weikeng Chen
Discreet Labs & UC Berkeley

Murat Kantarcioglu
University of Texas at Dallas & UC Berkeley

Raluca Ada Popa
UC Berkeley

A Artifact Appendix

A.1 Abstract

We present HOLMES, a protocol for performing secure distribution testing efficiently. Our artifact includes:

1. the efficiency comparison of HOLMES against various baselines;
2. the efficiency evaluation of HOLMES in real-world datasets;
3. the accuracy evaluation for HOLMES' statistical tests against corruptions to simulated and real-world data.

This artifact reproduces the tables and figures in our paper.

A.2 Description & Requirements

HOLMES allows efficient distribution testing in a multiparty setting without revealing the dataset of any of the parties. A distribution test is a predicate over an individual or joint (i.e., from multiple parties) dataset. Examples include well-known statistical tests, such as mean equality z-test (when the variance of the dataset is known) and t-test (when the variance is unknown), variance equality F-test, and Pearson's χ^2-test. These tests check a property between two populations, or between a population and a public distribution.

We support major statistical tests including the t-test, z-test, F-test, and the chi-squared test for single and multiple dimensions. We also provide support for computing and checking dataset properties essential in distribution testing; specifically, we support computing mean, trimmed mean, variance, histogram, random linear combination, and range check.

HOLMES integrates zero-knowledge proofs and secure multiparty computation with a lightweight consistency check. Specifically, HOLMES uses QuickSilver as a framework for zero-knowledge proofs and SCALE-MAMBA for the MPC computation. The codebase also includes integration tests, unit tests, and individual benchmarks.

A.2.1 Security, privacy, and ethical concerns

We assume that t parties want to participate in secure collaborative learning based on a t-party MPC protocol (e.g., SCALE-MAMBA). Before they engage in the learning protocol, the parties wish to check the quality of the dataset using distribution tests. HOLMES offers privacy in the dishonest and malicious majority setting, where at most $t-1$ out of t parties can collude and arbitrary deviate from the protocol.

Note that any distribution testing leaks one-bit information, i.e., whether the test passed or failed. Hence, it is important that a party does not participate in distribution tests that may leak sensitive information.

A.2.2 How to access

HOLMES is available (at a stable URL) here. The repository includes instructions for compiling HOLMES and reproducing our results.

The artifacts for reproducing our experiments and graphs are available (at a stable URL) here. We have also published the AMIs as public, and prepared scripts to automatically launch the clusters, so users can launch their own cluster on their own AWS account at here.

A.2.3 Hardware dependencies

HOLMES does not require any specialized hardware. Our experiments were performed on AWS c5.9xlarge instances, each with 36 cores and 72 GB memory. Different hardware configurations will affect the performance of HOLMES, but will result in a similar performance gain over the baselines.

A.2.4 Software dependencies

We provide the software dependencies for the plots and creating the AMI cluster in https://github.com/holmes-inputcheck/holmes-artifacts/. To install HOLMES only on a local machine, a user has to perform the following steps:

1. Install GMP https://gmplib.org/
2. Install MPFR https://www.mpfr.org/

3. Install FLINT https://www.flintlib.org/
4. Install emp-tool https://github.com/emp-toolkit/emp-tool
5. Install emp-ot https://github.com/emp-toolkit/emp-ot
6. Install emp-zk https://github.com/holmes-inputcheck/emp-zk
7. Clone and compile HOLMES https://github.com/holmes-anonymous-submission/holmes-library

A.2.5 Benchmarks

We provide benchmarks for the following tasks.

- Efficiency comparison for the histogram, mean, variance, and trimmed mean of HOLMES with the generic MPC baseline;
- Overhead comparison of range checks and ZK-friendly sketching with three baselines; these are the two most expensive gadgets supported in HOLMES;
- Overhead of running sample distribution tests on a real-world dataset from bank marketing. This dataset is provided in https://github.com/holmes-inputcheck/holmes-library, and is pre-cleaned and provided as CSV files in https://github.com/holmes-inputcheck/holmes-library/blob/master/bench/dataset[1-3].csv;
- Computing the accuracy of HOLMES' distribution tests against specific types of corruptions on simulated and real-world dataset.

A.3 Set-up

In this section, we provide information about setting up and running the artifacts.

A.3.1 Installation

We provide instructions on how to install the dependencies and necessary configuration steps in https://github.com/holmes-inputcheck/holmes-artifacts/.

A.3.2 Basic Test

We provide instructions for running unit tests on all the statistical tests of HOLMES in your AMI cluster located here, or on your local machine, given that you have all of the prerequisites installed, located here.

The instructions for running the integration tests, which measure the overhead of the dataset testing workflows, for your AMI cluster are provided in here, or on your local machine, given that you have all of the prerequisites installed, located here.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): HOLMES achieves a speedup of up to 10x for classical distribution tests over the generic MPC baseline with $t = 2$ parties. This is proven by experiment (E1). This result is described in Section 4.4 of the full version of paper and is illustrated in Figures 7a-f.

(C2): HOLMES achieves a speedup up to 10000x for its ZK-friendly sketching multidimensional tests over the strawman one-hot encoding multidimensional tests with $t = 2$ parties. This is proven by experiment (E2). This result is described in Section 4.4.2 of the full version of paper and is illustrated in Figures 7g-h.

(C3): The generic MPC baseline is 10–256x and 35–198x slower than HOLMES (i.e., QuickSilver, which is the underlying IZK protocol in HOLMES) for the range check and the ZK-friendly sketching, respectively, with $t \in \{2, 6, 10\}$ parties. The pairwise 2PC baseline is 4–32x slower for the range check and 13–36x slower for the ZK-friendly sketching than HOLMES. Spartan$_{\text{NIZK}}$ is 1–16× slower for the range check and 4–45x slower for the ZK-friendly sketching than HOLMES. This is proven by experiment (E3). This result is described in Section 4.4.1 of the full version of paper and is illustrated in Table 1.

(C4): HOLMES' chi-squared test has approximately the same accuracy as the naive normalized and unnormalized chi-squared test. This is proven by experiment (E4). This result is described in Section 4.3 of the full version of paper and is illustrated in Figure 5.

(C5): HOLMES's approach outperforms the generic MPC baseline by 77–264x for a real-world testing workflow on the bank marketing dataset for $t \in \{2, 6, 10\}$ parties. This is proven by experiment (E5). This result is described in Section 4.4.3 of the full version of paper and is illustrated in Figure 6 and Table 2.

A.4.2 Experiments

(E1): **Classic distribution tests for two parties** (20 human-minutes + 2 compute-hours + 72GB disk): This experiment measures the overhead of classic distribution tests, i.e., the naive histogram check, the trimmed mean check, the mean check, and the variance check, on a fake dataset of all ones. We compare the overhead between the two setups: HOLMES and SCALE-MAMBA.
Preparation: Perform the set up as described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#setup.
Execution: Use the designated scripts described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#misc-bench-scripts-experiment-e1–e2 to run the benchmarks and retrieve

the results.

Results: To interpret the results, run the designated scripts described in https://github.com/holmes-inputcheck/holmes-artifacts#classical-distribution-tests-experiment-e1. These scripts produce six figures, one for the cost in HOLMES and one for the cost in SCALE-MAMBA, for the following tests: histogram, trimmed mean, mean and variance. The cost of histogram check is plotted for 10 buckets with varying range sizes and input sizes. The cost of trimmed mean is plotted for datasets with 100k and 200k entries with varying threshold θ. The cost of mean and variance is summed and plotted for varying dataset sizes from 1 million entries to 5 million entries. This experiment supports claim (C1).

(E2): HOLMES' Multidimensional Test vs. Naive Multidimensional Test in HOLMES for two parties (20 human-minutes + 2 compute-hours + 72GB disk): This experiment measures the overhead of multidimensional tests on a fake dataset of all ones. We compare the overhead between two approaches for the multidimensional χ^2-test for the canonical two-party case with HOLMES (QuickSilver). We show that using our ZK-friendly sketching approach to compute the χ^2-test is much more efficient than the standard strawman approach of naively computing the one-hot encoding for each multidimensional input.

Preparation: Perform the set up as described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#setup.

Execution: Use the designated scripts described in https://github.com/holmes-inputcheck/holmes-artifacts#misc-bench-scripts-experiment-e1–e2 to run the benchmarks and retrieve the results.

Results: To interpret the results, run the designated scripts described in https://github.com/holmes-inputcheck/holmes-artifacts#holmes-multidimensional-test-vs-naive-multidimensional-test-experiment-e2. These scripts produce two figures showing the computational cost with respect to the number of dimensions and with respect to the number of individual labels in each dimension. In the first figure, the baseline is the naive multidimensional χ^2-test, whereas HOLMES uses the ZK-friendly sketching multidimensional χ^2-test. We plot the cost for datasets with 100k, 200k, 500k entries and 10 individual labels per dimension with varying number of dimensions. In the second figure, the cost of the multidimensional χ^2-test in SCALE-MAMBA and in HOLMES is plotted for datasets with 100k, 200k, 500k entries and four dimensions with varying number of labels. This experiment supports claim (C2).

(E3): Efficiency comparison of range checks and ZK-friendly sketching against the baselines (30 human-minutes + 20 compute-hours + 72GB disk): This experiment measures the overhead of range check and ZK-friendly sketching on a fake dataset of all ones. We compare the overhead between the following setups: HOLMES (i.e., QuickSilver), t-party SCALE-MAMBA, pairwise 2-party SCALE-MAMBA, Spartan$_{\text{NIZK}}$ for datasets with 100k, 200k, and 500k entries. For experiments that take too long or require more memory than available, e.g. 500k entries of ZK-friendly sketching for SCALE-MAMBA and Spartan$_{\text{NIZK}}$, we perform the ZK-friendly sketching for a smaller number of entries and extrapolate to larger entries using Euler's method. Spartan$_{\text{SNARK}}$ is only plotted for up to 100k entries.

Preparation: Perform the set up as described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#setup.

Execution: Use the designated scripts described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#range-checks-and-zk-friendly-sketching-against-the-baselines-experiment-e3 to run the benchmarks and retrieve the results.

Results: To interpret the results, run the designated scripts described in https://github.com/holmes-inputcheck/holmes-artifacts#range-checks-and-zk-friendly-sketching-against-the-baselines-experiment-e3. These scripts compute the cost for each setup. We produce a csv file, which can be compared with Table 2 and Table 3 in the paper. For each setup, we compute the cost for datasets with 100k, 200k, 500k entries for $t \in \{2, 6, 10\}$ parties. This experiment supports claim (C3), and as expected, QuickSilver runs the fastest in all setups.

(E4): Accuracy of distribution tests against corrupted datasets (5 human-minutes + 3 compute-hours): This experiment gradually corrupts up to 30% of the input dataset, and plots the statistical p-value of various tests as a function of the percentage of input corruptions.

Preparation: Perform the set up as described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#setup.

Execution: Use the designated scripts described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#statistical-corruption-accuracy-graphs-experiment-e4 to run the benchmarks and retrieve the results.

Results: To interpret the results, run the designated scripts described in https://github.com/holmes-inputcheck/holmes-artifacts#statistical-p-value-accuracy-graphs-experiment-e4. These scripts produce two plots: one for a simulated dataset and one for the bank marketing dataset. The corruption model is described in Section 4.3 of the full version of the paper. Due to randomness in sampling the dataset before corruption, the initial p-values might vary; in expectation, the chi-squared tests hit the p-value of 0.05

before all other tests, the z-test and t-test hit the p-value of 0.05 next followed by the F-test. This experiment supports claim (C4).

(E5): Marketing dataset overhead and cost breakdown (5 human-minutes + 1 compute-hour): This experiment measures the overhead of HOLMES and SCALE-MAMBA on the bank marketing dataset for $t \in \{2, 6, 10\}$ parties.

Preparation: Perform the set up as described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#setup.

Execution: Use the designated scripts described in the following link https://github.com/holmes-inputcheck/holmes-artifacts#marketing-dataset-testing-workflow-benchmarking-experiment-e5 to run the benchmarks and retrieve the results.

Results: To interpret the results, run the designated scripts described in https://github.com/holmes-inputcheck/holmes-artifacts#marketing-dataset-graphs-holmes-vs-mpc-baseline-experiment-e. These scripts produce a figure for the computational overhead as a function of the number of parties and a file with the breakdown of the cost. This experiment supports claim (C5).

A.5 Notes on Reusability

HOLMES' assumes that the highest degree of security (malicious security) is required and is best applied when all but one of the parties are untrusted. The parties most practically represent a powerful entity with lots of data and computing power. In example, competing bank conglomerates might want to jointly train their data over a specific model but do not trust each other, and are reasonably confident that other competing banks will collude. In this setting, HOLMES can securely perform distribution tests and securely compute aggregate statistics and analytics in a much faster speed than previous multiparty computation techniques.

HOLMES is flexible such that any future developer who chooses to use their own dataset, add their own custom checks, or add their own distribution tests can do so easily. An exciting future direction of HOLMES is that parties who wish to jointly train a model over their data can implement checks that prevent data poisoning attacks, such as algorithms from robust statistics. We encourage future users and developers to implement their own checks through HOLMES and use the existing checks to expedite the secure computation over their own selection of data.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20220926. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at `https://secartifacts.github.io/usenixsec2023/`.