USENIX Security '25 Artifact Appendix: Addressing Sensitivity Distinction in Local Differential Privacy: A General Utility-Optimized Framework

Xingyu He[1], Youwen Zhu[1], Rongke Liu[1], Gaoning Pan[2], Changyu Dong[3]

[1]Nanjing University of Aeronautics and Astronautics
[2]Hangzhou Dianzi University, [3]Guangzhou University

A Artifact Appendix

A.1 Abstract

This artifact contains Python-based code designed to evaluate ULDP protocols using both real-world and synthetic datasets. The artifact includes three main components: (1) the source code of all ULDP frequency protocols evaluated in the paper (including existing mechanisms uRR, uRAP, and uHR, as well as our proposed mechanisms uSS, uUE, and uLH); (2) scripts for running the experiments; (3) a complete README file. With all these components, the results presented in our paper can be reproduced.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

None.

A.2.2 How to access

The artifact is publicly available on Zenodo at `https://zenodo.org/records/15614307`. Please use the latest version of this record.

A.2.3 Hardware dependencies

This artifact is compatible with any machine equipped with at least 16GB of RAM. GPU resources are not required to run the code; however, improvements in CPU performance and memory capacity can reduce overall runtime. We conducted our experiments on two machines: 1. i9-13980HX with 16GB RAM; 2. i7-9700K with 16GB RAM.

A.2.4 Software dependencies

This artifact was developed using Python 3.9 and should be compatible with the latest versions of Python. While we ran our experiments on Windows, Python is a cross-platform language, which means the artifact can also run on macOS and Linux. The artifact only requires three Python packages: NumPy, xxHash, and Matplotlib. All of these packages can be installed via pip. In our experiments, we used Anaconda to manage the Python environment and provide an *environment.yml* file that fully describes the environment we used.

A.2.5 Benchmarks

The three real-world datasets and one synthetic dataset used in the experiments have been integrated into the artifact.

A.3 Set-up

A.3.1 Installation

Before installation, please ensure that Python is available on your system. We recommend using Anaconda/conda, and you can reproduce our environment using the following command:

```
conda env create -f environment.yml
```

Alternatively, you may manually install the required packages for the artifact using pip:

```
pip install numpy xxhash matplotlib
```

A.3.2 Basic Test

Open the "GLUTF-Pure ULDP Experiments (without frequency item mining)\src\exp1_impact_of_epsilon.py" file in the src directory, change

```
replication = 50
```

in the four functions to

```
replication = 1
```

, and change

```
for epsilon in [0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5]:
```

in the main function at the bottom to

```
for epsilon in [0.5]:
```

Then run the file.

```
python exp1_impact_of_epsilon.py
```

If the experimental and theoretical MSE of each mechanism are successfully displayed, it means success.

A.4 Evaluation workflow

Important Tips: To improve result stability, each experiment in our code is executed 50 times by default, and the average MSE is reported. While this setting ensures reliable results, it can be time-consuming. To speed up the evaluation, we recommend setting the number of repetitions (the corresponding parameters in the code are *replication* or *loop*) to 5 or 10 during initial testing. If the observed results deviate significantly from the theoretical MSEs due to randomness, increasing the replication count can help stabilize the outcomes.

A.4.1 Major Claims

(C1): *The experimental MSE of each mechanism decreases as the epsilon increases. Moreover, uSS is definitely better than uRR, and uUE is definitely better than uRAP. The utility of uLH is almost the same as that of uUE. This is proven by the experiment (E1) described in Section 6.2 whose results are illustrated in Figure 4.*

(C2): *The experimental MSE of each mechanism increases as the sensitive ratio increases. This is proven by the experiment (E2) described in Section 6.2 whose results are illustrated in Figure 5.*

(C3): *The MSE of the mechanism using collaborative sampling is almost the same as the MSE of the mechanism using the true theta. This is proven by the experiment (E3) described in Section 6.3 whose results are illustrated in Figure 6.*

(C4): *When we reduce z to 0, the mse will decrease in most cases (if the maximum value of z is 0, this action will have no effect). This is proven by the experiment (E4) described in Section 6.4 whose results are illustrated in Figure 7.*

(C5): *The experimental MAE of each mechanism decreases as the epsilon increases. Moreover, uSS is definitely better than uRR, and the utility of uLH is almost the same as that of uUE. This is proven by the experiment (E5) described in Section 6.5 whose results are illustrated in Figure 8.*

(C6): *Both the F1 and NDCG of the mechanisms increase with the increase of privacy budget. uRR performs the worst, while the other mechanisms perform similarly. This is proven by the experiment (E6) described in Section 6.5 whose results are illustrated in Figure 9.*

A.4.2 Experiments

(E1): *[exp1_impact_of_epsilon.py] [1.6 * replication (default is 50) compute-hour]: This experiment aims to evaluate the impact of privacy budget on the MSE of the ULDP mechanism.*

Preparation: *For faster verification, users can customize the parameter "replication" to a smaller number (such as 5 or 10), but the results will fluctuate more.*

Execution: *Enter the GLUTF-Pure ULDP Experiments (without frequency item mining)\src directory and execute:*

```
python exp1_impact_of_epsilon.py
```

Results: *The MSE of the ULDP mechanism will be output to the console. You are free to plot the results. You can feed the MSE into the exp_1_all_exp function in experiment_visualization.py to generate the graph, or you can use any method you like.*

(E2): *[exp1_impact_of_sensitive.py] [1.6 * replication (default is 50) compute-hour]: This experiment aims to evaluate the impact of sensitive ratio on the MSE of the ULDP mechanism.*

Preparation: *For faster verification, users can customize the parameter "replication" to a smaller number (such as 5 or 10), but the results will fluctuate more.*

Execution: *Enter the GLUTF-Pure ULDP Experiments (without frequency item mining)\src directory and execute:*

```
python exp1_impact_of_sensitive.py
```

Results: *The MSE of the ULDP mechanism will be output to the console. You are free to plot the results. You can feed the MSE into the exp_1_vary_xs function in experiment_visualization.py to generate the graph, or you can use any method you like.*

(E3): *[exp2_collaborative_sampling.py] [0.8 * replication (default is 50) compute-hour]: This experiment aims to evaluate whether the use of cooperative sampling has an impact on the MSE of the ULDP mechanism.*

Preparation: *For faster verification, users can customize the parameter "replication" to a smaller number (such as 5 or 10), but the results will fluctuate more.*

Execution: *Enter the GLUTF-Pure ULDP Experiments (without frequency item mining)\src directory and execute:*

```
python exp2_collaborative_sampling.py
```

Results: *The MSE of the ULDP mechanism with collaborative sampling will be output to the console. The MSE of the ULDP mechanism with true theta can be found in E1. You are free to plot the results. You can feed the MSE into the exp_2_uSS, exp_2_uOUE and exp_2_uOLH function in experiment_visualization.py to generate the graph, or you can use any method you like.*

(E4): *[exp3_without_z.py] [0.7 * replication (default is 50) compute-hour]: This experiment aims to evaluate the impact of z on the MSE of the ULDP mechanism.*

Preparation: *Before executing this file, the protocol code needs to be modified. Specifically, when initializing the definition parameters, add the following code at the end of the __init__ function:*

```
self.z = 0
self.z_star = 1-self.f
```

For faster verification, users can customize the parameter "replication" to a smaller number (such as 5 or 10), but the results will fluctuate more.

Execution: *Enter the GLUTF-Pure ULDP Experiments (without frequency item mining)\src directory and execute:*

```
python exp3_without_z.py
```

Results: *The MSE of the ULDP mechanism will be output to the console. You are free to plot the results. You can feed the MSE into the exp_3_without_z_uSS, exp_3_without_z_uUE and exp_3_without_z_uLH function in experiment_visualization.py to generate the graph, or you can use any method you like.*

(E5): *[exp4_mae.py] [1.6 * replication (default is 50) compute-hour]: This experiment aims to evaluate the impact of privacy budget on the MAE of the ULDP mechanism.*

Preparation: *For faster verification, users can customize the parameter "replication" to a smaller number (such as 5 or 10), but the results will fluctuate more.*

Execution: *Enter the GLUTF-Pure ULDP Experiments (without frequency item mining)\src directory and execute:*

```
python exp4_mae.py
```

Results: *The MAE of the ULDP mechanism will be output to the console. You are free to plot the results. You can feed the MSE into the exp_4_mae function in experiment_visualization.py to generate the graph, or you can use any method you like.*

(E6): *[Folder:(only for frequency item mining)] [0.16 * replication (default is 50) compute-hour]: This experiment aims to evaluate the impact of privacy budget on the F1 and NDCG of the ULDP mechanism.*

Preparation: *For faster verification, users can customize the parameter "loop" to a smaller number (such as 5 or 10), but the results will fluctuate more.*

Execution: *Enter the GLUTF-Pure ULDP Experiments (only for frequency item mining)\src directory and execute:*

```
python exp-<mechanism>-<dataset>.py
```

Results: *The results will be stored in the results folder in txt format. You are free to plot the results. You can feed the result into the exp_4_item_mining function in experiment_visualization.py to generate the graph, or you can use any method you like.*

A.5 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2025/.