A EXPERIMENTS FOR GOWALLA

In this section, we present the experimental results for the Gowalla dataset [7]. The false positive rate of all Bloom filters is set to 0.0001. The results consistently show that PPRC is several times more accurate and dozens of more efficient than the other protocols.

A.1 Accuracy Comparison

A.1.1 ***The Effect of*** I. Fig. 10(a)-(b) report MAE and MRE as the number of DHs varies over $I \in \{5, 10, 15, 20\}$. Here we set the $P = 0.1$ and $S = 2K$.

As I increases, MAE increases while MRE decreases for PPRC, FM-Count, and LL-count; for RCC, both the MAE and MRE increase. PPRC is several times more accurate than RCC, FM-Count, and LL-Count. For example, in Fig. 10(a), the MAE of PPRC is 23.65, 38.93, 46.39, and 51.31 for $I = 5, 10, 15, 20$, respectively. On average, PPRC is 19.07, 1.93, and 2.73 times more accurate than RCC, FM-Count, and LL-Count. Fig. 10(b) corroborates these findings.

A.1.2 ***The Effect of*** P. Fig. 10(c)-(d) show MAE and MRE as P varies over $\{0.05, 0.1, 0.15, 0.2\}$, where P denotes the proportion of each DH's private dataset relative to the whole Gowalla dataset. We fix $I = 10$ and $S = 2K$.

With increasing P, MAE increases and MRE decreases for PPRC, FM-Count, and LL-Count, whereas both MAE and MRE increase for RCC. Our PPRC remains several times more accurate than RCC, FM-Count, and LL-Count. Fig. 10(c) presents the MAE results. When P is 0.05, 0.1, 0.15, and 0.2, the MAE of PPRC is 21.95, 38.93, 43.91, and 56.85, respectively. On average, PPRC is 18.67, 1.92, and 2.76 times more accurate than RCC, FM-Count, and LL-Count. Fig. 10(d) shows consistent results.

A.1.3 ***The Effect of*** S. Fig. 10(e)-(f) evaluate the effect of the counter length of the count estimation sketches $S \in \{1K, 2K, 3K, 4K\}$. The results for RCC remain constant as they do not use the count estimation sketch. Here we set the $P = 0.1$ and $I = 10$.

MAE and MRE of PPRC, FM-Count, and LL-Count decrease with larger S. Similarly, PPRC is several times more accurate than RCC, FM-Count, and LL-Count. As shown in Fig. 10(e), the MAE of PPRC decreases from 46.51 to 31.59 as S increases. Furthermore, on average, our PPRC is 14.47, 1.85, and 2.66 times more accurate than RCC, FM-Count, and LL-Count, respectively. Similar trends are observed in Fig. 10(f).

A.2 Efficiency Comparison

A.2.1 ***Time Cost.*** Figs. 11(a) and 11(c) show the Time cost with respect to the number of DHs $I \in \{5, 10, 15, 20\}$ and the data proportion $P \in \{0.05, 0.1, 0.15, 0.2\}$. We set $I = 10$, $P = 0.1$, and $S = 2K$ as default.

As I and P increase, the Time cost of all protocols grows. Furthermore, PPRC is dozens of times faster than the other protocols. For example, in Fig. 11(a), the Time cost of PPRC is 1.957s, 1.962s, 1.966s, and 1.972s for $I = 5, 10, 15, 20$, respectively. On average, PPRC is 10.11, 22.00, 23.07, 22.83, and 42.67 times faster than RCC, LC-Count, FM-Count, LL-Count, and TVA, respectively. Consistent results are shown in Fig. 11(c).

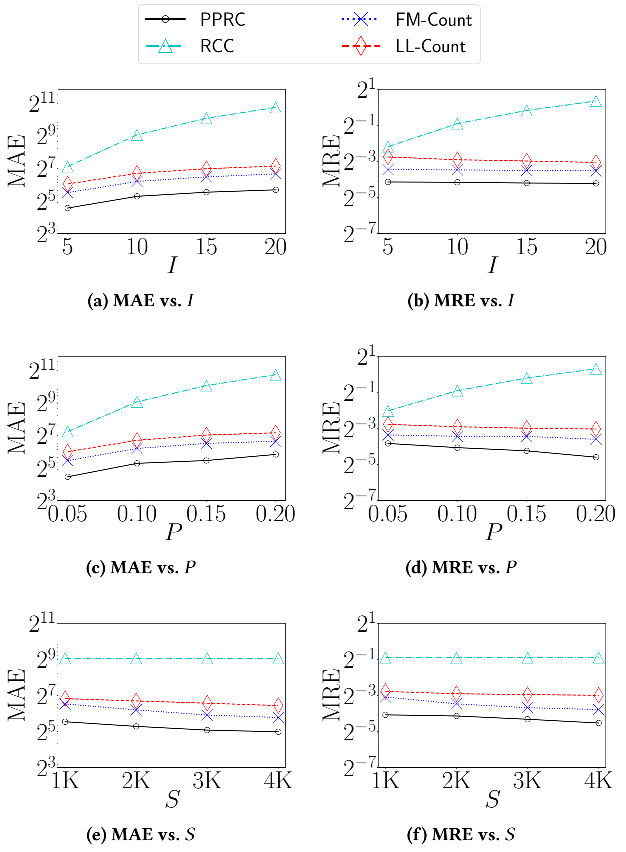

Figure 10: Accuracy comparison with various I, P, and S.

A.2.2 ***Communication Cost.*** Figs. 11(b) and 11(d) present the Communication cost concerning $I \in \{5, 10, 15, 20\}$ and $P \in \{0.05, 0.1, 0.15, 0.2\}$. We also set $I = 10$, $P = 0.1$, and $S = 2K$ as default.

As I increases, the communication cost of all protocols increases. As P increases, the communication of TVA increases while the communication cost of the other protocols remains constant. Furthermore, the communication cost of PPRC is several times lower than LC-Count, FM-Count, LL-Count, and TVA. Fig. 11(b) presents the Communication cost with various I. When I is 5, 10, 15, 20, the communication cost of PPRC is 34.52MB, 63.28MB, 92.05MB, and 120.81MB, respectively. On average, PPRC reduces the communication cost by 4.75, 4.79, 4.75, and 19.31 times compared with LC-Count, FM-Count, LL-Count, and TVA. Although RCC achieves lower communication costs, it does not provide the same level of accuracy and computational efficiency as PPRC. Consistent results are shown in Fig. 11(d).

A.2.3 ***The Effect of*** S. Fig. 11(e)-(f) evaluate the impact of the counter length of count estimation sketches S on Time and Communication costs. Here, we consider $S \in \{1K, 2K, 3K, 4K\}$, with $I = 10$ and $P = 0.1$. Since RCC and TVA do not use count estimation sketches, their costs remain constant regardless of S.

The Time cost of PPRC, LC-Count, FM-Count, and LL-Count increases with S. Furthermore, PPRC is dozens of times faster than RCC, LC-Count, FM-Count, LL-Count, and TVA. In Fig. 11(e), Time

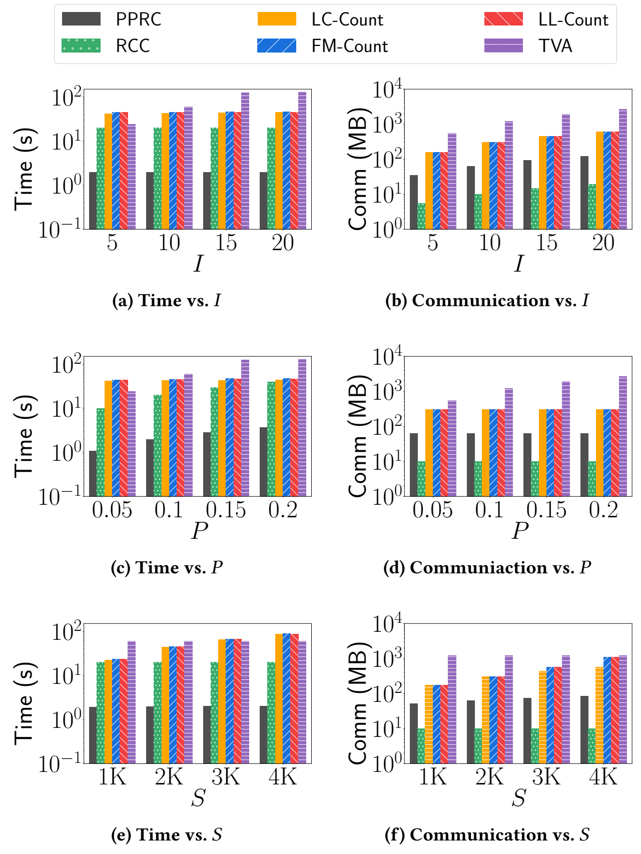

Figure 11: Efficiency comparison with various I, P, and S.

of PPRC is 1.923s, 1.965s, 2.000s, and 2.013s for $S \in \{1\text{K}, 2\text{K}, 3\text{K}, 4\text{K}\}$, respectively. On average, PPRC is 10.05, 27.43, 28.15, 27.66, and 30.01 times faster than RCC, LC-Count, FM-Count, LL-Count, and TVA. Fig. 11(f) shows similar patterns for the Communication cost.

B LC-COUNT

In this section, we describe the LC-Count protocol. The Setup and Range Evaluation phases are identical to those in PPRC; thus, we focus on the Aggregation phase, whose pseudocode is shown in Algorithm 4. This phase consists of three steps:

- **Step 1 (Lines 1-8): Sketch Construction.** Each data holder i initializes its LC sketch LC_i with S encrypted counters, each set to $\text{E}(0)$ using the public key pk. All data holders jointly agree on a hash function h that uniformly maps each record $(x_n^{(i)}, y_n^{(i)})$ to an index in $\{1, \ldots, S\}$. For each record, the data holder adds its encrypted range indicator $\text{E}(L_n^{(i)})$ to the corresponding counter as $\text{E}(\text{LC}_i[h(x_n^{(i)}, y_n^{(i)})]) = \text{E}(\text{LC}_i[h(x_n^{(i)}, y_n^{(i)})]) + \text{E}(L_n^{(i)})$. After all updates, $\text{E}(\text{LC}_i)$ is normalized using the $\texttt{ZeroTest}(\cdot)$ operation [41], which is based on Lagrange interpolation and transforms $\text{E}(0) \rightarrow \text{E}(0)$ while mapping all non-zero ciphertexts to $\text{E}(1)$. This normalization ensures that each counter reflects whether any record was hashed into its position.
- **Step 2 (Lines 9-13): Secure Aggregation.** Each data holder sends its encrypted sketch $\text{E}(\text{LC}_i)$ to the central aggregator. The aggregator performs element-wise homomorphic addition to aggregate all sketches: $\text{E}(\text{LC}[s]) = \sum_{i=1}^{I} \text{E}(\text{LC}_i[s]), s \in [1, S]$. It then applies $\texttt{ZeroTest}(\cdot)$ again to normalize all counters and computes the encrypted number of zero counters as $\text{E}(S') = S - \sum_{s=1}^{S} \text{E}(\text{LC}[s])$.
- **Step 3 (Lines 14-16): Decryption and Estimation.** The central aggregator sends $\text{E}(S')$ to the query user, who decrypts it to obtain S'. The query user then estimates the final range count using the standard LC estimator $RC = -S \ln(S'/S)$.

Algorithm 4: Aggregation Phase of LC-Count

```
Input  : Dataset $\{(x_n^{(i)}, y_n^{(i)}, \text{E}(L_n^{(i)}))\}_{n=1}^{N_i}$ held by each data
         holder $i \in [1, I]$.
Output : Range count $RC$ obtained by the query user.
/* - Local Sketch Construction by data holders - */
1  Each data holder $i$ initializes each counter of its LC sketch
     $\text{LC}_i[s] = 0$ for all $s \in [1, S]$ and encrypts each counter into
     $\text{E}(\text{LC}_i[s])$ using $pk$;
2  All data holders jointly agree on a hash function $h$;
3  for DH $i = 1$ to $I$ do
4      for $n = 1$ to $N_i$ do
5          $s = h(x_n^{(i)}, y_n^{(i)})$;
6          $\text{E}(\text{LC}_i[s]) = \text{E}(\text{LC}_i[s]) + \text{E}(L_n^{(i)})$;
7      for $s = 1$ to $S$ do
8          $\text{E}(\text{LC}_i[s]) = \texttt{ZeroTest}(\text{E}(\text{LC}_i[s]))$; // ZeroTest:
             maps E(0) → E(0), others → E(1)
/* - Secure Aggregation by the Central
   Aggregator - */
9  Each data holder $i$ sends $\text{E}(\text{LC}_i)$ to the central aggregator;
10 for $s = 1$ to $S$ do
11     $\text{E}(\text{LC}[s]) = \sum_{i=1}^{I} \text{E}(\text{LC}_i[s])$;
12     $\text{E}(\text{LC}[s]) = \texttt{ZeroTest}(\text{E}(\text{LC}[s]))$;
13 Compute $\text{E}(S') = S - \sum_{s=1}^{S} \text{E}(\text{LC}[s])$;
/* - Decryption and Estimation - */
14 The central aggregator sends $\text{E}(S')$ to the query user;
15 The query user decrypts $\text{E}(S')$ into $S'$ using $sk$;
16 The query user computes $RC = -S \ln\left(\frac{S'}{S}\right)$. ;
```

C FM-COUNT

In this section, we present the FM-Count protocol. Similar to LC-Count, its Setup and Range Evaluation phases remain unchanged from PPRC; hence, we focus on describing the Aggregation phase, whose pseudocode is provided in Algorithm 5. This phase consists of three steps:

- **Step 1 (Lines 1-10): Sketch Construction.** Each data holder i constructs an encrypted FM sketch $\text{E}(\text{FM}_i)$ composed of C arrays, and each array contains W encrypted counters. All counters are initialized to $\text{E}(0)$ using the public key pk. All data holders jointly agree on C independent hash functions $\{h_1, \ldots, h_C\}$. For each record $(x_n^{(i)}, y_n^{(i)}, \text{E}(L_n^{(i)}))$, data holder i computes C

Algorithm 5: Aggregation Phase of FM-Count

```
Input  : Dataset $\{(x_n^{(i)}, y_n^{(i)}, \mathrm{E}(L_n^{(i)}))\}_{n=1}^{N_i}$ held by each data
         holder $i \in [1, I]$.
Output : Range count $RC$ obtained by the query user.
/* – Local Sketch Construction by data holders – */
1  Each data holder $i$ initializes each counter of its FM sketch
   $\mathrm{FM}_i[c][w] = 0$ for all $c \in [1, C], w \in [1, W]$ and encrypts
   each counter into $\mathrm{E}(\mathrm{FM}_i[c][w])$ using $pk$;
2  All data holders jointly agree on $C$ independent hash
   functions $\{h_1, \ldots, h_C\}$;
3  for DH $i = 1$ to $I$ do
4      for $n = 1$ to $N_i$ do
5          for $c = 1$ to $C$ do
6              $w_c = h_c(x_n^{(i)}, y_n^{(i)})$;
7              $\mathrm{E}(\mathrm{FM}_i[c][w_c]) = \mathrm{E}(\mathrm{FM}_i[c][w_c]) + \mathrm{E}(L_n^{(i)})$;
8      for $c = 1$ to $C$ do
9          for $w = 1$ to $W$ do
10             $\mathrm{E}(\mathrm{FM}_i[c][w]) = \mathtt{ZeroTest}(\mathrm{E}(\mathrm{FM}_i[c][w]))$ ;
               // ZeroTest: maps E(0) → E(0), others → E(1)
/* – Secure Aggregation by the Central Aggregator – */
11 Each data holder $i$ sends $\mathrm{E}(\mathrm{FM}_i)$ to the central aggregator;
12 for $c = 1$ to $C$ do
13     for $w = 1$ to $W$ do
14         $\mathrm{E}(\mathrm{FM}[c][w]) = \sum_{i=1}^{I} \mathrm{E}(\mathrm{FM}_i[c][w])$;
15         $\mathrm{E}(\mathrm{FM}[c][w]) = \mathtt{ZeroTest}(\mathrm{E}(\mathrm{FM}[c][w]))$;
16 for $c = 1$ to $C$ do
17     for $w = 2$ to $W$ do
18         $\mathrm{E}(\mathrm{FM}[c][w]) = \mathrm{E}(\mathrm{FM}[c][w]) \cdot \mathrm{E}(\mathrm{FM}[c][w-1])$;
19 Compute $\mathrm{E}(Z_N) = \sum_{c=1}^{C} \sum_{w=1}^{W} \mathrm{E}(\mathrm{FM}[c][w])$;
/* – Decryption and Estimation – */
20 The central aggregator sends $\mathrm{E}(Z_N)$ to the query user;
21 The query user decrypts $\mathrm{E}(Z_N)$ into $Z_N$ using $sk$;
22 The query user computes $RC = 2^{\frac{Z_N}{C}}/\phi$, where $\phi \approx 0.77351$;
```

hash indices $w_c = h_c(x_n^{(i)}, y_n^{(i)})$ for $c = 1, \ldots, C$, and updates the sketch as $\mathrm{E}(\mathrm{FM}_i[c][w_c]) = \mathrm{E}(\mathrm{FM}_i[c][w_c]) + \mathrm{E}(L_n^{(i)})$ for $c = 1, \ldots, C$. After processing all records, $\mathtt{ZeroTest}(\cdot)$ is applied to normalize each counter such that $\mathrm{E}(\mathrm{FM}_i[c][w]) = \mathrm{E}(1)$ if any record contributes to the position, and $\mathrm{E}(0)$ otherwise.

- **Step 2 (Lines 11-19): Secure Aggregation.** Each data holder i sends $\mathrm{E}(\mathrm{FM}_i)$ to the central aggregator, which aggregates them counter-wise as $\mathrm{E}(\mathrm{FM}[c][w]) = \sum_{i=1}^{I} \mathrm{E}(\mathrm{FM}_i[c][w])$ for $c = 1, \ldots, C$ and $w = 1, \ldots, W$. The aggregator applies $\mathtt{ZeroTest}(\cdot)$ again to ensure that all aggregated counters remain binary. It then performs a prefix-wise cumulative multiplication over each row to propagate leading zeros, as in the original FM sketch design. Finally, it computes $\mathrm{E}(Z_N) = \sum_{c=1}^{C} \sum_{w=1}^{W} \mathrm{E}(\mathrm{FM}[c][w])$.
- **Step 3 (Lines 20-22): Decryption and Estimation.** The central aggregator sends $\mathrm{E}(Z_N)$ to the query user, who decrypts it to obtain Z_N. The final range count is estimated using the standard FM sketch estimator $RC = 2^{\frac{Z_N}{C}}/\phi$, where $\phi \approx 0.77351$ is a bias-correction constant.

D LL-COUNT

Algorithm 6: Aggregation Phase of LL-Count

```
Input  : Dataset $\{(x_n^{(i)}, y_n^{(i)}, \mathrm{E}(L_n^{(i)}))\}_{n=1}^{N_i}$ held by each data
         holder $i \in [1, I]$.
Output : Range count $RC$ obtained by the query user.
/* – Local Sketch Construction by data holders – */
1  Each data holder $i$ initializes each counter of its LL sketch
   $\mathrm{LL}_i[c][w] = 0$ for all $c \in [1, C], w \in [1, W]$ and encrypts
   each counter using $pk$;
2  All data holders jointly agree on $C + 1$ independent hash
   functions $\{h, h_1, \ldots, h_C\}$;
3  for DH $i = 1$ to $I$ do
4      for $n = 1$ to $N_i$ do
5          $c = h(x_n^{(i)}, y_n^{(i)})$;
6          $w = h_c(x_n^{(i)}, y_n^{(i)})$;
7          $\mathrm{E}(\mathrm{LL}_i[c][w]) = \mathrm{E}(\mathrm{LL}_i[c][w]) + \mathrm{E}(L_n^{(i)})$;
8      for $c = 1$ to $C$ do
9          for $w = 1$ to $W$ do
10             $\mathrm{E}(\mathrm{FM}_i[c][w]) = \mathtt{ZeroTest}(\mathrm{E}(\mathrm{FM}_i[c][w]))$ ;
               // ZeroTest: maps E(0) → E(0), others → E(1)
/* – Secure Aggregation by the Central Aggregator – */
11 Each data holder sends $\mathrm{E}(\mathrm{LL}_i)$ to the central aggregator;
12 for $c = 1$ to $C$ do
13     for $w = 1$ to $W$ do
14         $\mathrm{E}(\mathrm{LL}[c][w]) = \sum_{i=1}^{I} \mathrm{E}(\mathrm{LL}_i[c][w])$;
15         $\mathrm{E}(\mathrm{LL}[c][w]) = 1 - \mathtt{ZeroTest}(\mathrm{E}(\mathrm{LL}[c][w]))$;
16 for $c = 1$ to $C$ do
17     for $w = W-1$ to $1$ do
18         $\mathrm{E}(\mathrm{LL}[c][w]) = \mathrm{E}(\mathrm{LL}[c][w]) \cdot \mathrm{E}(\mathrm{LL}[c][w+1])$;
19 Compute $\mathrm{E}(Z_N) = C(W-1) - \sum_{c=1}^{C} \sum_{w=1}^{W} \mathrm{E}(\mathrm{LL}[c][w])$;
/* – Decryption and Estimation – */
20 The central aggregator sends $\mathrm{E}(Z_N)$ to the query user;
21 The query user decrypts $\mathrm{E}(Z_N)$ into $Z_N$ using $sk$;
22 The query user computes $RC = \alpha_C \cdot C \cdot 2^{Z_N/C}$, where
   $\alpha_C = \left(\Gamma(-\frac{1}{C}) \cdot \frac{1-2^{\frac{1}{C}}}{\log 2}\right)^C$;
```

In this section, we present the LL-Count protocol. Similar to LC-Count, its Setup and Range Evaluation phases remain unchanged from PPRC, and we thus focus on the Aggregation phase. The pseudocode is shown in Algorithm 6, consisting of three main steps:

- **Step 1 (Lines 1-10): Sketch Construction.** Each data holder i constructs an encrypted LL sketch $\mathrm{E}(\mathrm{LL}_i)$ composed of C arrays, and each array contains W encrypted counters. All counters are initialized to $\mathrm{E}(0)$ using the public key pk. The data holders jointly agree on $C+1$ independent hash functions $\{h, h_1, \ldots, h_C\}$, where h selects the target array and h_c determines the position within that array. For each $(x_n^{(i)}, y_n^{(i)}, \mathrm{E}(L_n^{(i)}))$, data holder i computes $c = h(x_n^{(i)}, y_n^{(i)})$ and $w = h_c(x_n^{(i)}, y_n^{(i)})$, and updates the encrypted counter as $\mathrm{E}(\mathrm{LL}_i[c][w]) = \mathrm{E}(\mathrm{LL}_i[c][w]) + \mathrm{E}(L_n^{(i)})$. Finally, $\texttt{ZeroTest}(\cdot)$ is applied to normalize each counter such that $\mathrm{E}(\mathrm{LL}_i[c][w]) = \mathrm{E}(1)$ if any record contributes to the position, and $\mathrm{E}(0)$ otherwise.
- **Step 2 (Lines 11-19): Secure Aggregation.** Each data holder sends $\mathrm{E}(\mathrm{LL}_i)$ to the central aggregator, which aggregates them counter-wise: $\mathrm{E}(\mathrm{LL}[c][w]) = \sum_{i=1}^{I} \mathrm{E}(\mathrm{LL}_i[c][w])$. The central aggregator then converts the counters via $\mathrm{E}(\mathrm{LL}[c][w]) = 1 - \texttt{ZeroTest}(\mathrm{E}(\mathrm{LL}[c][w]))$, ensuring that each position reflects whether at least one record across all data holders was mapped there. Next, a backward cumulative multiplication is applied along each row to propagate the rightmost one bit toward the left, consistent with the LL sketch procedure. Finally, the central aggregator computes $\mathrm{E}(Z_N) = C(W-1) - \sum_{c=1}^{C} \sum_{w=1}^{W} \mathrm{E}(\mathrm{LL}[c][w])$.
- **Step 3 (Lines 20-22): Decryption and Estimation.** The central aggregator sends $\mathrm{E}(Z_N)$ to the query user, who decrypts it using the secret key sk to obtain Z_N. The query user then estimates the range count using the standard LL sketch estimator $RC = \alpha_C \cdot C \cdot 2^{Z_N/C}$, where $\alpha_C = \left(\Gamma(-\frac{1}{C}) \cdot \frac{1-2^{1/C}}{\log 2}\right)^C$ is the standard bias-correction factor.