Differential Privacy in Applications

Jiaqi Shao

The Chinese University of Hong Kong, Shenzhen

July 8, 2022

PACK

1. **Plan**
   - Centralized DP has privacy controller $\rightarrow$ *LDP*
2. **Achievement**
   - PEM with local differential privacy
   - Find heavy hitters under different client data distribution
3. **Challenge**
   - Data Distribution Bias
   - Noise reduces accuracy

PACK

1. **Plan**
   - Centralized DP has privacy controller $\rightarrow$ *LDP*

2. **Achievement**
   - PEM with local differential privacy
   - Find heavy hitters under different client data distribution

3. **Challenge**
   - Data Distribution Bias
   - Noise reduces accuracy

PEM with local differential privacy

Input: ClientSize: n, MaxBitLen: m, ε, BatchSize: g, TopK: k, Round: rnds

Output: Top-K Heavy Hitters: HH, Evaluation Score: score

BitsPerBatch: $b \leftarrow m/g$

Server Side:

for $rnd = 1$ to $rnds$

$C_0 = \{\}$

for $i = 1$ to g: # Divide clients into g batches

Construct $D_i = C_i \times \{0, 1\}^b$

Receive Reports from clients

Aggregate Responses to get D_i

Construct C_i from candidates in D_i

$HH_{rnd} = C_g$

$score_{rnd} =$ Evaluate(HH)

Client Side:

for $i = 1$ to g:

Report $v' =$ PrivacyMechanism($v[: i \cdot b]$) to Server

PEM with local differential privacy: Privacy Module

Direct Encoding (DE): There is no encoding, e.g. Random Response Techmique (GRR)

GRR

Given the domain D, domain size $|\mathrm{D}| = d$. Perturb input v as follows.

$$\Pr\left[\mathcal{M}_{\mathrm{GRR}}(x) = i\right] = \begin{cases} p = \frac{e^{\varepsilon}}{e^{\varepsilon}+d-1}, & \text{if } i = x \\ q = \frac{1-p}{d-1} = \frac{1}{e^{\varepsilon}+d-1}, & \text{if } i \neq x \end{cases}$$

ε-LDP Satisfaction: For any inputs v_1, v_2 and output y, we have:

$$\frac{\Pr\left[\mathcal{M}_{\mathrm{GRR}}\left(v_1\right) = y\right]}{\Pr\left[\mathcal{M}_{\mathrm{GRR}}\left(v_2\right) = y\right]} \leq \frac{p}{q} = \frac{e^{t/\left(e^{t}+d-1\right)}}{1/\left(e^{t}+d-1\right)} = e^{\epsilon}$$

PEM with local differential privacy: Privacy Module

Unary Encoding (UE): input v is encoded into a one-hot bit vector with length d, perturb 0 to 1. **Encode**:

$$B = Enc(v) = [0, ..., 1, 0, 0]$$

Perturb: $B' = Perturb(B)$

$$\Pr\left[B'[i] = 1\right] = \begin{cases} p, & \text{if } B[i] = 1 \\ q, & \text{if } B[i] = 0 \end{cases}$$

ε-**LDP**: $\varepsilon = ln(\frac{p(1-q)}{(1-p)q})$

- Symmetric Unary Encoding (SUE): $p = \frac{e^{\varepsilon/2}}{e^{\varepsilon/2}+1}, q = 1 - p = \frac{1}{e^{\varepsilon/2}+1}$
- Optimized Unary Encoding (OUE): $p = 1/2, q = \frac{1}{e^{\varepsilon/2}+1}$ (Optimized for Variance)

PEM with local differential privacy: Privacy Module

Local Hashing (LH): Reduce domain size to d', where $d' < d$
Let $\mathbb{H}$ be a universal hash function family, such that each hash function $H \in \mathbb{H}$ hashes an input in $[d']$
Optimized Local Hashing (OLH):
Encode: $Enc_{OLH}(v) = \langle H, x \rangle$, where $H \leftarrow_R \mathbb{H}$ is chosen uniformly at random from $\mathbb{H}$, and $x = H(v)$.
Perturb: $Perturb_{OLH}(\langle H, x \rangle) = \langle H, y \rangle$ where

$$\forall_{i \in [d']} \Pr[y = i] = \begin{cases} p = \frac{e^\epsilon}{e^\epsilon + d'1}, & \text{if } x = i \\ q = \frac{1}{e^\epsilon + d' + 1}, & \text{if } x \neq i \end{cases}$$

ε-**LDP**:

$$\frac{\Pr[\langle \mathrm{H}, \mathrm{y} \rangle | \mathrm{v}_1]}{\Pr[\langle \mathrm{H}, \mathrm{y} \rangle | \mathrm{v}_2]} = \frac{\Pr[\mathrm{Perturb}(\mathrm{H}(\mathrm{v}_1)) = \mathrm{y}]}{\Pr[\mathrm{Perturb}(\mathrm{H}(\mathrm{v}_2)) = \mathrm{y}]} \leq \frac{p}{q} = e^\varepsilon$$

Aggregation: I_v: the number of reports that "supports" the input v

$$I_v = |\{j | H^j(v) = y^j\}|$$

PEM with local differential privacy: Evaluation Module

F1 Score: C_T: truth heavy hitters, C_g: output heavy hitters (no order)

$$F1 = \frac{2 \cdot |C_T \cap C_g|}{2 \cdot |C_T \cap C_g| + |C_g - C_T \cap C_g|}$$

Normalized Cumulative Rank (NCR)

$$\mathrm{NCG} = \frac{\sum_{i \in [k]} rel_i}{\sum_{i \in [REL_k]} rel_i}$$

v in position i with relevant score $rel_i = k - i + 1$

Normalized Discount Cumulative Gain (NDCG)

$$\mathrm{nDCG_k} = \frac{DCG_k}{IDCG_k}$$

$$where \quad \mathrm{DCG_k} = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}, \quad \mathrm{IDCG_j} = \sum_{i=1}^{|REL_k|} \frac{rel_i}{\log_2(i+1)}$$

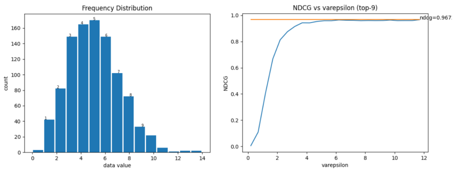
PACK

1. **Plan**
   - Centralized DP has privacy controller $\rightarrow$ *LDP*
2. **Achievement**
   - PEM with local differential privacy
   - Find heavy hitters under different client data distribution
3. **Challenge**
   - Data Distribution Bias
   - Noise reduces accuracy

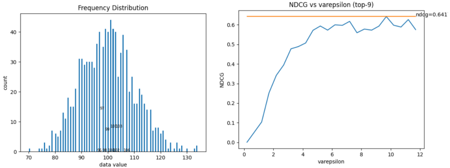
Find heavy hitters under different client data distribution

Input: $\textsc{ClientSize} = 1000$, $\textsc{PrivacyModule} = \text{GRR}$, $\textsc{EvaluationModule} = \text{NDCG}$, $\textsc{MaxBitLen} = 16$, $\textsc{TopK} = 9$

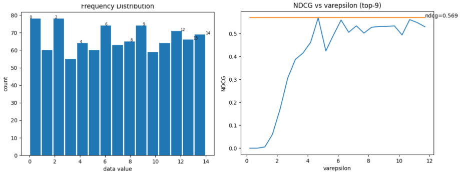
Distribution vs. NDCG



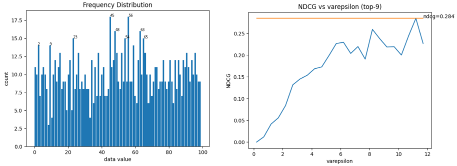
Poisson Distribution with $\lambda = 5$



Poisson Distribution with $\lambda = 100$



Uniform Discrete Distribution [0, 14]



Uniform Discrete Distribution [0, 100]

PACK

1. **Plan**
   - Centralized DP has privacy controller $\rightarrow$ *LDP*
2. **Achievement**
   - PEM with local differential privacy
   - Find heavy hitters under different client data distribution
3. **Challenge**
   - Data Distribution Bias
   - Noise reduces accuracy

PACK

1. **Plan**
   - Centralized DP has privacy controller $\rightarrow$ *LDP*
2. **Achievement**
   - PEM with local differential privacy
   - Find heavy hitters under different client data distribution
3. **Challenge**
   - Data Distribution Bias
   - Noise reduces accuracy

PEM with local differential privacy: Evaluation Module

Observation:

- More concentrated distributed (Poisson) data evaluates much better than Uniformed distributed client data. $\swarrow$
- Noise Based LDP reduce the utility since noise is accumulated during aggregation. So, large Privacy Budget is required for the small data set.
  $\Rightarrow$ Is it possible for achieving LDP without noise? (Like CDP)
- Client Sampling affects the evaluation, since clients in prior batches will control the prefix extending procedure.
  $\Rightarrow$ Diminishing clients size per batch.