A Brief Tutorial on Sparse Vector Technique

—— An Advanced Mechanism in Differential Privacy

Outline

- Recap of Differential Privacy
- Sparse Vector Technique
- Generalized SVT: An Enhanced Version [VLDB '17]
- Case Study 1: Mbeacon [NDSS '19]
- Case Study 2: PrivateSQL [VLDB '19]
- Case Study 3: Privacy-preserving Deep Learning [CCS '15]

- Lyu, M., Su, D., & Li, N. (2017). Understanding the sparse vector technique for differential privacy. Proceedings of the VLDB Endowment, 10(6), 637-648.
- Hagestedt, I., Zhang, Y., Humbert, M., Berrang, P., Tang, H., Wang, X., & Backes, M. (2019, February). MBeacon: Privacy-Preserving Beacons for DNA Methylation Data. In NDSS.
- Kotsogiannis, I., Tao, Y., He, X., Fanaeepour, M., Machanavajjhala, A., Hay, M., & Miklau, G. (2019). PrivateSQL: a differentially private SQL query engine. Proceedings of the VLDB Endowment, 12(11), 1371-1384.
- Shokri, R., & Shmatikov, V. (2015, October). Privacy-preserving deep learning. In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (pp. 1310-1321).

Differential Privacy

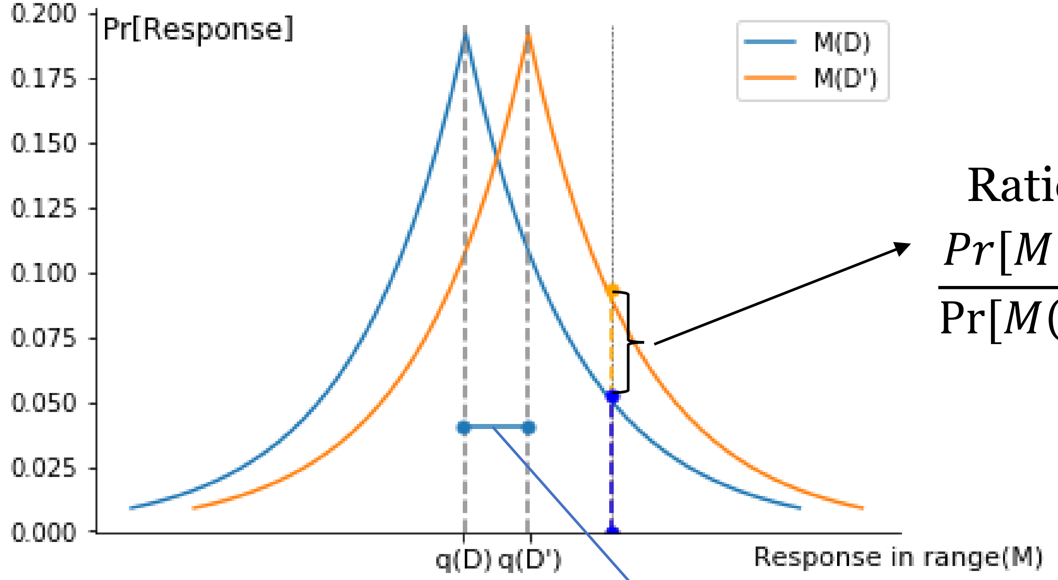
- For every pair of inputs, say $D \text{ and } D'$, which differ in one row, taking the output, the likelihood ratio between observing $D \text{ and } D'$ is bounded by e^{ϵ}.
  - Namely, the adversary cannot distinguish $D \text{ and } D'$ based on the output O.

ϵ-DP:

$$Pr[M(D) = S] \leq e^{\epsilon} Pr[M(D') = S]$$

Laplace Mechanism:

$$M_L(x, q(\cdot), \epsilon) = q(x) + Lap(\frac{\Delta}{\epsilon})$$



Ratio Bound:

$$\frac{Pr[M(D) = S]}{\Pr[M(D') = S]} \leq e^{\epsilon}$$

Sensitivity:

$$\Delta = \max |q(D) - q(D')|$$

* ϵ is called the privacy budget, a smaller ϵ indicates better privacy but often worse data utility.

Differential Privacy

- Exponential Mechanism
  - Answering non-numerical queries such as "most popular fruit" (Table 1).
  - Consider the "utility score" of a response: $u: N^{|D|} \times Range \to R$.
    - The utility score reflect the users' preference to the items.

<u>Exponential Mechanism:</u>

$M_E(x, u, Range)$ selects and outputs an element $r \in Range$ with prob. proportional to $\exp(\frac{\epsilon u(x,r)}{2\Delta u})$.

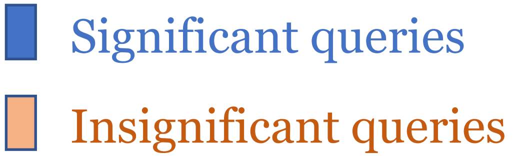
Table 1. An Example for Exponential Mechanism.

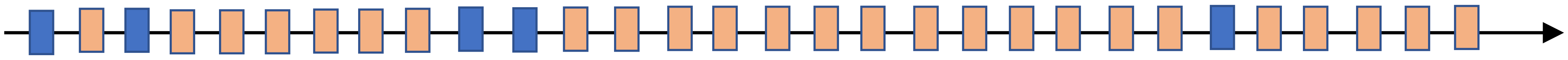
Category	Utility Score $\Delta u = 1$	$Pr[Response]$		
		$\epsilon = 0$	$\epsilon = 0.1$	$\epsilon = 1$
Apple	30	0.25	0.424	0.924
Orange	25	0.25	0.330	0.075
Pear	8	0.25	0.141	1.5E-05
Pineapple	2	0.25	0.105	7.7E-07

◆ McSherry, F., & Talwar, K. (2007, October). Mechanism design via differential privacy. In 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07) (pp. 94-103). IEEE.

Sparse Vector Technique

- Motivating Example
  - Consider a very large number, say k, of queries to answer. If using Laplace mechanism, ϵ would be proportional to k.
  - But what if the data analyst believe only a few queries are significant, and will take value above a certain threshold?





- Goal and Intuition:
  - Saving privacy budget.
    - Add less noise to achieve the same level of privacy.
    - Answer insignificant queries (with negative results) *"for free"*.
  - Only gives "positive/negative" response, not the noisy value.
    - The answer is *sparse*.

◆ Hardt, M., & Rothblum, G. N. (2010, October). A multiplicative weights mechanism for privacy-preserving data analysis. In 2010 IEEE 51st Annual Symposium on Foundations of Computer Science (FOCS'10) (pp. 61-70). IEEE.

Sparse Vector Technique

- Algorithm 1. Basic Sparse Vector Technique.
  - Input: A private database D, a stream of queries $Q = q_1, q_2, \ldots$ each with sensitivity no more than Δ, a sequence of thresholds $T = T_1, T_2, \ldots$, and the number c of queries to expect positive answers.
  - Output: A vector of indicators $A = a_1, a_2, \ldots$, where each $a_i \in \{\top, \bot\}$.

$\top$ — positive
$\bot$ — negative

Input: $D, Q, \Delta, \mathbf{T} = T_1, T_2, \cdots, c.$

$\epsilon_1 = \epsilon/2, \quad \rho = \mathsf{Lap}(\Delta/\epsilon_1)$
$\epsilon_2 = \epsilon - \epsilon_1, \quad \text{count} = 0$
for each query $q_i \in Q$ **do**
    $\nu_i = \mathsf{Lap}(2c\Delta/\epsilon_2)$
    if $q_i(D) + \nu_i \geq T_i + \rho$ **then**
        Output $a_i = \top$
        count = count + 1, **Abort** if count $\geq c$.
    else
        Output $a_i = \bot$

* Note that now we discuss SVT in an interactive setting.

Sparse Vector Technique

- **Algorithm 1. Basic Sparse Vector Technique.**
  - Input: A private database D, a stream of queries $Q = q_1, q_2, \ldots$ each with sensitivity no more than Δ, a sequence of thresholds $T = T_1, T_2, \ldots$, and the number c of queries to expect positive answers.
  - Output: A vector of indicators $A = a_1, a_2, \ldots$, where each $a_i \in \{\top, \bot\}$.

$\top$ — positive

$\bot$ — negative

Input: $D, Q, \Delta, \mathbf{T} = T_1, T_2, \cdots, c.$

$\epsilon_1 = \epsilon/2$, $\rho = \mathsf{Lap}(\Delta/\epsilon_1)$

$\epsilon_2 = \epsilon - \epsilon_1$, count = 0

for each query $q_i \in Q$ **do**

  $\nu_i = \mathsf{Lap}(2c\Delta/\epsilon_2)$

  if $q_i(D) + \nu_i \geq T_i + \rho$ **then**

    Output $a_i = \top$

    count = count + 1, **Abort** if count $\geq c$.

  else

    Output $a_i = \bot$

Generate (the same) noise ρ and add to each threshold.

Generate noise ν_i and add to each query q_i.

* Note that now we discuss SVT in an interactive setting.

Sparse Vector Technique

- Algorithm 1. Basic Sparse Vector Technique.
  - Input: A private database D, a stream of queries $Q = q_1, q_2, \ldots$ each with sensitivity no more than Δ, a sequence of thresholds $T = T_1, T_2, \ldots$, and the number c of queries to expect positive answers.
  - Output: A vector of indicators $A = a_1, a_2, \ldots$, where each $a_i \in \{\top, \bot\}$.

$\top$ — positive
$\bot$ — negative

Input: $D, Q, \Delta, \mathbf{T} = T_1, T_2, \cdots, c$.

$\epsilon_1 = \epsilon/2$, $\rho = \mathsf{Lap}(\Delta/\epsilon_1)$

$\epsilon_2 = \epsilon - \epsilon_1$, count = 0

for each query $q_i \in Q$ **do**

  $\nu_i = \mathsf{Lap}(2c\Delta/\epsilon_2)$

  if $q_i(D) + \nu_i \geq T_i + \rho$ **then**

    Output $a_i = \top$

    count = count + 1, **Abort** if count $\geq c$.

  else

    Output $a_i = \bot$

Generate (the same) noise ρ and add to each threshold.

Generate noise v_i and add to each query q_i.

Stop when the number of $\top$s outnumbers c.

* Note that now we discuss SVT in an interactive setting.

Sparse Vector Technique

- Analysis
  - **Theorem.** Algorithm 1 satisfies ϵ-DP.

$Pr[M(D) = S] \leq e^{\epsilon} Pr[M(D') = S]$

Sparse Vector Technique

$Pr[M(D) = S] \leq e^{\epsilon} Pr[M(D') = S]$

- Analysis
  - **Theorem.** Algorithm 1 satisfies ϵ-DP.
  - **Proof.** Consider any $a_i \in \{\top, \bot\}^l$. Let $a = < a_1, \dots, a_l >$, $I_\top = \{i: a_i = \top\}, and\ I_\bot = \{i: a_i = \bot\}$. Let

$$f_i(D,z) = \Pr[q_i(D) + \nu_i < T_i + z]$$
$$g_i(D,z) = \Pr[q_i(D) + \nu_i \geq T_i + z].$$

  - We have:

$$\Pr[\mathcal{A}(D) = \boldsymbol{a}] = \int_{-\infty}^{\infty} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\top} \Pr[q_i(D) + \nu_i \geq T_i + z] \prod_{i \in \mathbf{I}_\bot} \Pr[q_i(D) + \nu_i < T_i + z] \ dz$$
$$= \int_{-\infty}^{\infty} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\top} \Pr[q_i(D) + \nu_i \geq T_i + z] \ dz \times \int_{-\infty}^{\infty} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\bot} \Pr[q_i(D) + \nu_i < T_i + z] \ dz$$

- Integrate all possible values for ρ, the noise added to the threshold.
- The same logic for D'.

Sparse Vector Technique

- Analysis
  - **Proof.** (Cont.)

$$f_i(D,z) = \Pr[q_i(D)+\nu_i < T_i+z]$$
$$g_i(D,z) = \Pr[q_i(D)+\nu_i \geq T_i+z]\,.$$

$$\frac{\Pr[\mathcal{A}(D)=\boldsymbol{a}]}{\Pr[\mathcal{A}(D')=\boldsymbol{a}]}$$
$$= \frac{\int_{-\infty}^{\infty} \Pr[\rho=z] \prod_{i\in\mathbf{I}_\perp} f_i(D,z) \prod_{i\in\mathbf{I}_\top} g_i(D,z)\, dz}{\int_{-\infty}^{\infty} \Pr[\rho=z] \prod_{i\in\mathbf{I}_\perp} f_i(D',z) \prod_{i\in\mathbf{I}_\top} g_i(D',z)\, dz}$$

$$= \frac{\int_{-\infty}^{\infty} \Pr[\rho=z-\Delta] \prod_{i\in\mathbf{I}_\perp} f_i(D,z-\Delta) \prod_{i\in\mathbf{I}_\top} g_i(D,z-\Delta)\, dz}{\int_{-\infty}^{\infty} \Pr[\rho=z] \prod_{i\in\mathbf{I}_\perp} f_i(D',z) \prod_{i\in\mathbf{I}_\top} g_i(D',z)\, dz}$$

- The change of integration variable from z to $z-\Delta$.

$$\leq \frac{\int_{-\infty}^{\infty} e^{\epsilon_1} \Pr[\rho=z] \prod_{i\in\mathbf{I}_\perp} f_i(D',z) \prod_{i\in\mathbf{I}_\top} g_i(D,z-\Delta)\, dz}{\int_{-\infty}^{\infty} \Pr[\rho=z] \prod_{i\in\mathbf{I}_\perp} f_i(D',z) \prod_{i\in\mathbf{I}_\top} g_i(D',z)\, dz}$$

- Since $\rho = Lap\left(\frac{\Delta}{\epsilon_1}\right)$, $\Pr[\rho = z-\Delta] \leq e^{\epsilon_1} \Pr[\rho = z]$. We leave $f_i(D,z-\Delta) \leq f_i(D',z)$ later.

Sparse Vector Technique

- Analysis
  - **Proof.** (Cont.)

$$\frac{\Pr[\mathcal{A}(D) = \boldsymbol{a}]}{\Pr[\mathcal{A}(D') = \boldsymbol{a}]}$$

$$\leq \frac{\int_{-\infty}^{\infty} e^{\epsilon_1} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\perp} f_i(D', z) \prod_{i \in \mathbf{I}_\top} g_i(D, z - \Delta)\, dz}{\int_{-\infty}^{\infty} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\perp} f_i(D', z) \prod_{i \in \mathbf{I}_\top} g_i(D', z)\, dz}$$

▸ The result from last page.

$$\leq \frac{\int_{-\infty}^{\infty} e^{\epsilon_1} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\perp} f_i(D', z) \prod_{i \in \mathbf{I}_\top} e^{\frac{\epsilon_2}{c}} g_i(D', z)\, dz}{\int_{-\infty}^{\infty} \Pr[\rho = z] \prod_{i \in \mathbf{I}_\perp} f_i(D', z) \prod_{i \in \mathbf{I}_\top} g_i(D', z)\, dz}$$

▸ $g_i(D, z - \Delta) \leq e^{\frac{\epsilon_2}{c}} g_i(D', z)$, we will show this later.

$$\leq e^{\epsilon_1} \left(e^{\frac{\epsilon_2}{c}} \right)^c = e^{\epsilon_1 + \epsilon_2} = e^{\epsilon}$$

▸ We have at most c positive outcomes, i.e. $|I_\top| \leq c$.

Sparse Vector Technique

- Analysis
  - **Proof.** (Cont.)

$$f_i(D,z) = \Pr[q_i(D)+\nu_i < T_i+z]$$
$$g_i(D,z) = \Pr[q_i(D)+\nu_i \geq T_i+z]\,.$$

$$\begin{aligned} g_i(D, z-\Delta) &= \Pr[q_i(D)+\nu_i \geq T_i+z-\Delta] \\ &\leq \Pr\left[q_i(D')+\Delta+\nu_i \geq T_i+z-\Delta\right] \\ &= \Pr\left[q_i(D')+\nu_i \geq T_i+z-2\Delta\right] \\ &\leq e^{\frac{\epsilon_2}{c}} \Pr\left[q_i(D')+\nu_i \geq T_i+z\right] \\ &= e^{\frac{\epsilon_2}{c}} g_i(D', z). \end{aligned}$$

- Global sensitivity, by definition: $q_i(D') - \Delta \leq q_i(D) \leq q_i(D') + \Delta$.
- v_i is sampled from $Lap(\frac{2c\Delta}{\epsilon_2})$.

$$\begin{aligned} f_i(D, z-\Delta) &= \Pr[q_i(D) + v_i < T_i + z - \Delta] \\ &\leq \Pr[q_i(D') - \Delta + v_i < T_i + z - \Delta] \\ &\leq \Pr[q_i(D') + v_i < T_i + z] = f_i(D', z) \end{aligned}$$

- Same logic for $f_i(D, z-\Delta)$.

Herewith we finish the proof.

Generalized SVT

- Algorithm 2. Generalized SVT in [VLDB '17].

Input: $D, Q, \Delta, \mathbf{T} = T_1, T_2, \cdots, c$ and ϵ_1, ϵ_2 and ϵ_3.
Output: A stream of answers $a_1, a_2, \cdots$

$\rho = \mathsf{Lap}\left(\frac{\Delta}{\epsilon_1}\right)$, count = 0
for Each query $q_i \in Q$ **do**
  $\nu_i = \mathsf{Lap}\left(\frac{2c\Delta}{\epsilon_2}\right)$
  if $q_i(D) + \nu_i \geq T_i + \rho$ **then**
    if $\epsilon_3 > 0$ **then**
      Output $a_i = q_i(D) + \mathsf{Lap}\left(\frac{c\Delta}{\epsilon_3}\right)$
    else
      Output $a_i = \top$
    count = count + 1, **Abort** if count $\geq c$.
  else
    Output $a_i = \bot$

Theorem. Algorithm 2 satisfies $(\epsilon_1 + \epsilon_2 + \epsilon_3)$-DP.

Part 1. $(\epsilon_1 + \epsilon_2)$-DP as shown in analysis of Algorithm 1.

Part 2. **If** provide noisy answer, **then** consume ϵ_3-DP.

* Part 2 is taken into account in algorithm 2 because in many variants of SVT they output the noisy answers. This part is to explicitly show that outputting noisy answers needs additional privacy budget.

Generalized SVT

- Budget Allocation
  - Different strategy in allocating $\epsilon_1 + \epsilon_2$ results in different Accuracy.
  - Recap the comparing part of SVT:

$$q_i(D) + Lap\left(\frac{2c\Delta}{\epsilon_2}\right) \geq T + Lap(\frac{\Delta}{\epsilon_1})$$

  - If minimize the variance of $Lap\left(\frac{\Delta}{\epsilon_1}\right) - Lap(\frac{2c\Delta}{\epsilon_2})$, we can optimize the accuracy without sacrificing privacy. That is:

$$\begin{aligned} \min \quad & [2\left(\frac{\Delta}{\epsilon_1}\right)^2 + 2\left(\frac{2c\Delta}{\epsilon_2}\right)^2] \\ \text{s.t.} \quad & \epsilon_1 + \epsilon_2 = \text{t} \end{aligned}$$

  - Solve it and you can get $\epsilon_1 : \epsilon_2 = 1 : (2c)^{2/3}$

* Note that in the optimization problem, t denotes a fixed constant, which in fact, is $\epsilon - \epsilon_3$.

Generalized SVT

- SVT for Monotonic Queries (MQ)
  - MQ*: for any changes from D to D', the change in answers of all queries is in the same direction (i.e. either $\forall_i \, q_i(D) \geq q_i(D'), or \; \forall_i \, q_i(D) \leq q_i(D')$).
  - For monotonic queries, the optimization of privacy budget allocation becomes $\epsilon_1 : \epsilon_2 = 1 : c^{2/3}$.
- SVT vs. EM
  - In a non-interactive setting, EM can achieve the same goal.
    - Runs EM c times, each with budget $\frac{\epsilon}{c}$; the quality of the query is its answer; each query is selected with prob. proportional to $\exp(\frac{\epsilon}{2c\Delta})$.
  - EM can be proven to achieve better accuracy.

* This is common in the data mining field, e.g. using SVT for frequent itemset mining.

Recommendation from [VLDB ’17]

- In interactive settings, use the generalized SVT with optimal privacy budget allocation.

- In non-interactive settings, do not use SVT and use EM instead.
  - If one gets better performance using SVT than using EM,
  - then it is likely that one’s usage of SVT is *non-private*.

◆ Lyu, M., Su, D., & Li, N. (2017). Understanding the sparse vector technique for differential privacy. Proceedings of the VLDB Endowment, 10(6), 637-648.

Case study 1: MBeacon

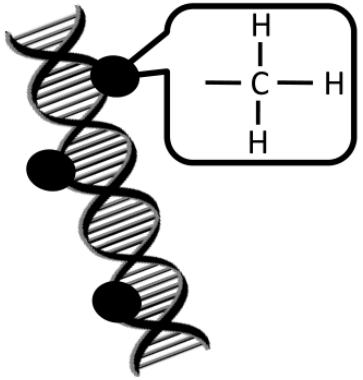
- Title: MBeacon: Privacy-Preserving Beacons* for DNA Methylation (甲基化) Data
  - Authors: Inken Hagestedt, Yang Zhang†, Mathias Humbert, Pascal Berrang, Haixu Tang, XiaoFeng Wang, Michael Backes
  - In NDSS 2019, distinguished paper award

- Highlights:
  - Attacked a biomedical data search engine system.
  - Proposed defense mechanism based on a tailored SVT algorithm.

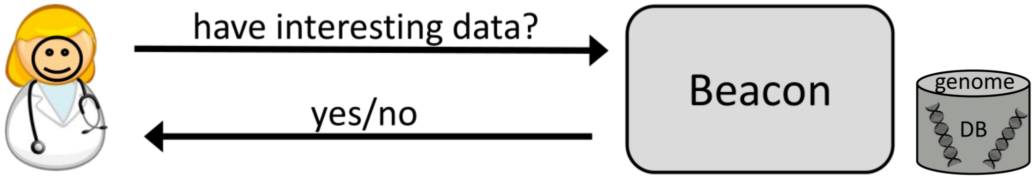
◆ Hagestedt, I., Zhang, Y., Humbert, M., Berrang, P., Tang, H., Wang, X., & Backes, M. (2019, February). MBeacon: Privacy-Preserving Beacons for DNA Methylation Data. In NDSS.

* A kind of molecular probe (分子探针), also the name of a search engine in this paper.

Case study 1: MBeacon



- Background
  - Methylation Data
    - A kind of important molecule located on DNA that influence cell life (on how to copy, express, etc.).
    - For privacy research, privacy breach exists since attacker may infer target's sensitive information (e.g. cancer, smokes, stressed).
  - Beacon system
    - A search engine for biomedical researchers that answers: *whether its database contains any record with the specified nucleotide* (核苷酸) *at a given position*
    - Only gives Yes/No response



https://en.wikipedia.org/wiki/DNA_methylation
https://beacon-network.org/

Case study 1: MBeacon

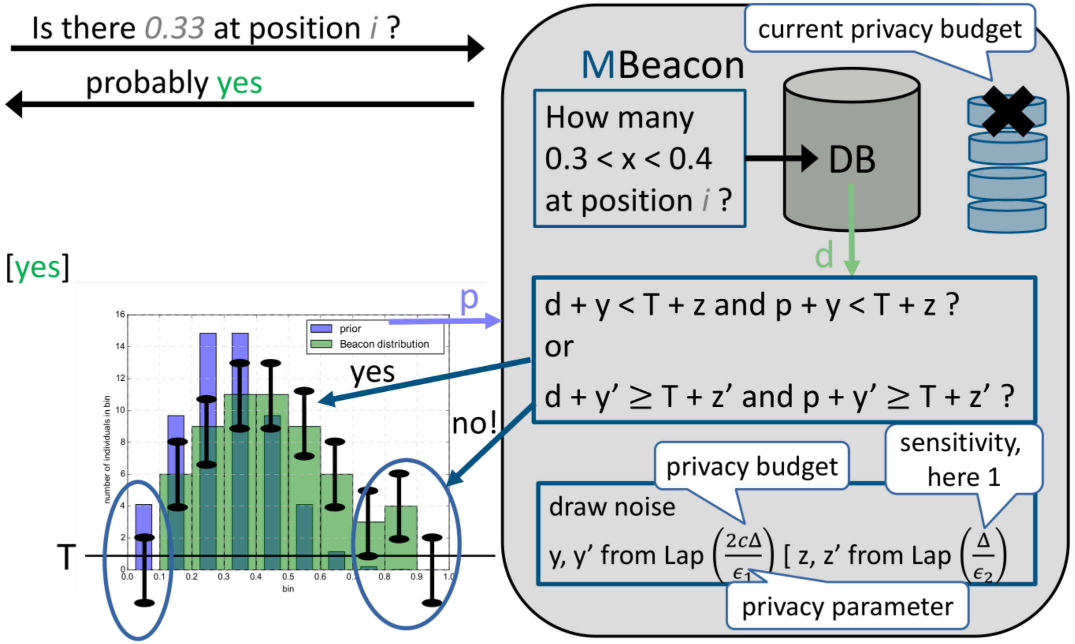
- Modeling
  - DNA methylation data
    - A sequence of real numbers[1], each between 0-1, i.e. $m(v) \in R_{[0,1]}^{M}$.
  - Query type
    - Are there any patients with this methylation value at a specific methylation position?
    - → Are there any patients with methylation value above some threshold for a specific position?
    - $B_I\colon\ q \to \{0,1\},\ q \ := (pos, val)$
  - Threat Model
    - Membership inference attack.
    - Adversary with access to the victim's methylation data $m(v)$ aims to infer whether the victim is in a certain database. In this case, database is with specified disease tags.
    - $A\colon\ (m(v), B_I, K) \to \{0,1\}$, K denotes some additional knowledge (i.e. means and std deviations of the general population at the methylation positions).

1. Each value represents the fraction of methylated dinucleotides (二核苷酸) at this position.

Case study 1: MBeacon

- Defense Mechanism
  - Intuition
    - Adversary successfully attacks the system, iff the output of the query deviate his background knowledge, which means he learns additional info from the query.
    - According to biomedical research, only a few methylation regions differ from the general population. —— Sparse vector technique.
  - Double SVT: SVT²
    - The ith query is not privacy-sensitive if:

$$((\alpha_i + y_i < T + z_1) \ and \ (\beta_i < T + z_1)) \\ or \ \ ((\alpha_i + y_i' \geq \mathrm{T} + z_2) \ and \ (\beta_i \geq T + z_2))$$



    - The algorithm answers negative for these non-privacy-sensitive queries; and positive otherwise.

* α_i is the number of patients in the MBeacon that corresponds to the query q_i;
β_i is the estimated number of patients given by the general population.

Case study 1: MBeacon

- Defense Mechanism
  - Part 1. Tailored SVT (right figure).
  - Part 2. Transform SVT result to MBeacon results (left figure).

Input: base threshold T, privacy parameters ϵ_1, ϵ_2 and c, query sensitivity Δ, query vector $\overrightarrow{Q}$, database $\mathbb{I}$ and prior frequency $\mathtt{P}$

Result: sanitized MBeacon responses $B_{\mathbb{I}}(\overrightarrow{Q})$

$\overrightarrow{R} = \mathcal{A}(T, \epsilon_1, \epsilon_2, c, \Delta, \overrightarrow{Q}, \mathbb{I}, \mathtt{P})$;
for *each query* q_i *in* $\overrightarrow{Q}$ **do**
  get r_i from $\overrightarrow{R}$; get β_i from $\mathtt{P}$;
  if $r_i = \bot$ **then**
    $B_{\mathbb{I}}(q_i) = \beta_i \geq T$;
  else
    $B_{\mathbb{I}}(q_i) = \neg(\beta_i \geq T)$;
  end
end

Input: base threshold T, privacy parameters ϵ_1, ϵ_2 and c, query sensitivity Δ, query vector $\overrightarrow{Q}$, database $\mathbb{I}$ and prior frequency $\mathtt{P}$

Result: sanitized responses R such that $r_i \in \{\bot, \top\}$ for each i

$z_1 = \text{Lap}(\frac{\Delta}{\epsilon_1})$; $z_2 = \text{Lap}(\frac{\Delta}{\epsilon_1})$;
count = 0;
for *each query* q_i *in* $\overrightarrow{Q}$ **do**
  $y_i = \text{Lap}(\frac{2c\Delta}{\epsilon_2})$; $y'_i = \text{Lap}(\frac{2c\Delta}{\epsilon_2})$;
  get α_i from $\mathbb{I}$ and β_i from $\mathtt{P}$;
  if $(\alpha_i + y_i < T + z_1$ *and* $\beta_i + y_i < T + z_1)$ *or* $(\alpha_i + y'_i \geq T + z_2$ *and* $\beta_i + y'_i \geq T + z_2)$ **then**
    $r_i = \bot$;
  else
    $r_i = \top$;
    count = count + 1 ;
    $z_1 = \text{Lap}(\frac{\Delta}{\epsilon_1})$; $z_2 = \text{Lap}(\frac{\Delta}{\epsilon_1})$;
  end
  if *count* $\geq c$ **then**
    Halt
  end
end

Case study 2: PrivateSQL

- Title: PrivateSQL: A Differentially Private SQL Query Engine
  - Authors: Ios Kotsogiannis, Yuchao Tao, Xi He, Maryam Fanaeepour, Ashwin Machanavajjhala , Michael Hay, Gerome Miklau
  - In VLDB 2019

- Highlights
  - System work — an end-to-end differentially private relational database system is proposed, which supports a rich class of SQL queries.
  - Automatically calculating sensitivity and adding noise.
  - Answering complex SQL counting queries under a fixed privacy budget by generating <u>private synopses</u>.

- Kotsogiannis, I., Tao, Y., He, X., Fanaeepour, M., Machanavajjhala, A., Hay, M., & Miklau, G. (2019). PrivateSQL: a differentially private SQL query engine. Proceedings of the VLDB Endowment, 12(11), 1371-1384.

Case study 2: PrivateSQL

- Design Goals:
  - Workloads:
    - The system should answer a workload of queries with bounded privacy loss.
  - Complex Queries:
    - Each query in the workload can be a complex SQL expression over multiple relations.
  - Multi-resolution Privacy:
    - The system should allow the data owner to specify which entities in the database require protection.

Private Synopses

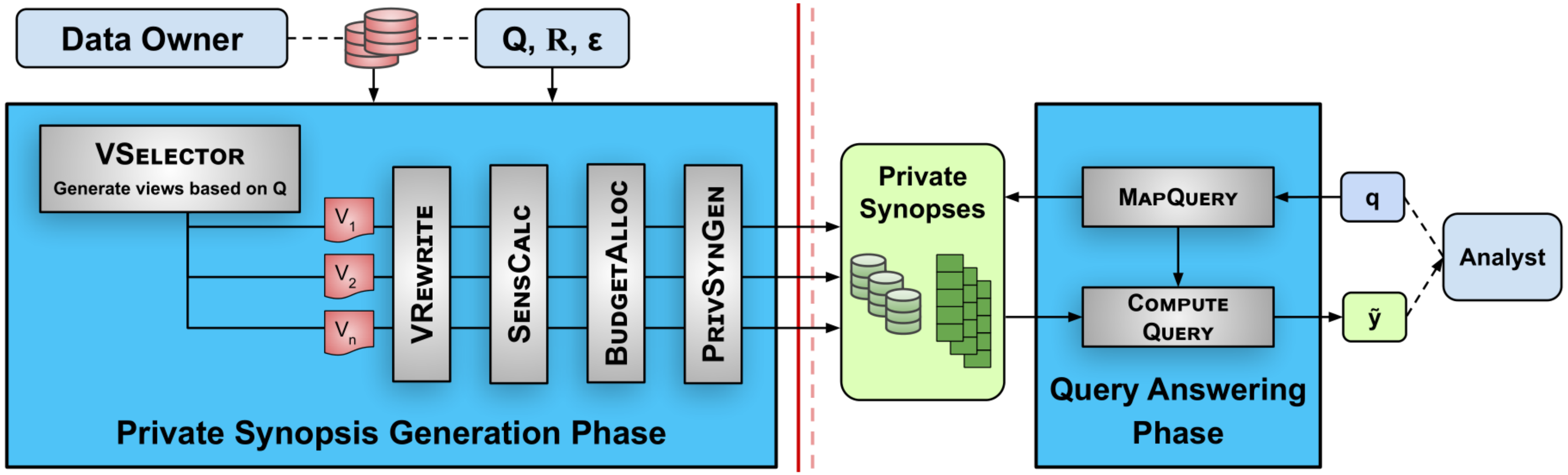
Privacy Policies

Case study 2: PrivateSQL

- Architecture
  - Two main phases
    - Phase 1. Synopsis Generation.
    - Phase 2. Query Answering.

A **synopsis** captures important statistical information about the database.

A **view** is interpreted as a relational algebra expression.



Case study 2: PrivateSQL

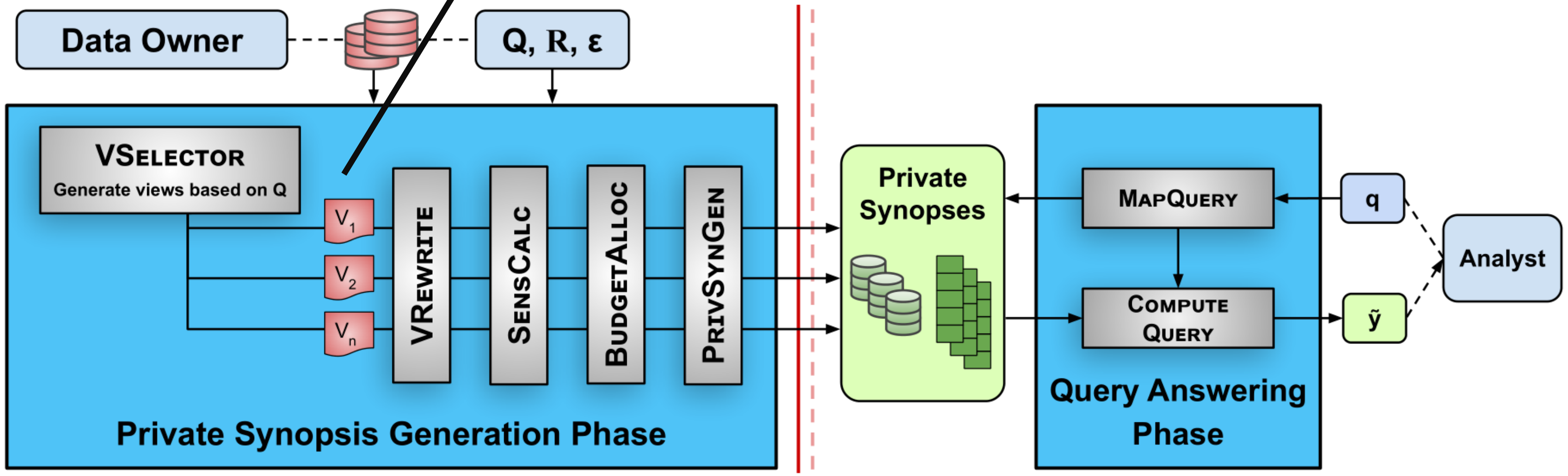
- Architecture
  - Two main phases
    - Phase 1. Synopsis Generation.
    - Phase 2. Query Answering.

<u>Challenge.</u> 1. hard to compute the global sensitivity of a SQL view; 2. some operation may yield unbounded numbers of tuples.

<u>Solution.</u> 1. learn a threshold from data; 2. adopt Truncation operator to bound the join size by throwing away join keys above the threshold.

A **synopsis** captures important statistical information about the database.

A **view** is interpreted as a relational algebra expression.



* SVT is used as a sub-routine to calculate the threshold from the data.

Case study 3: Privacy-preserving Deep Learning

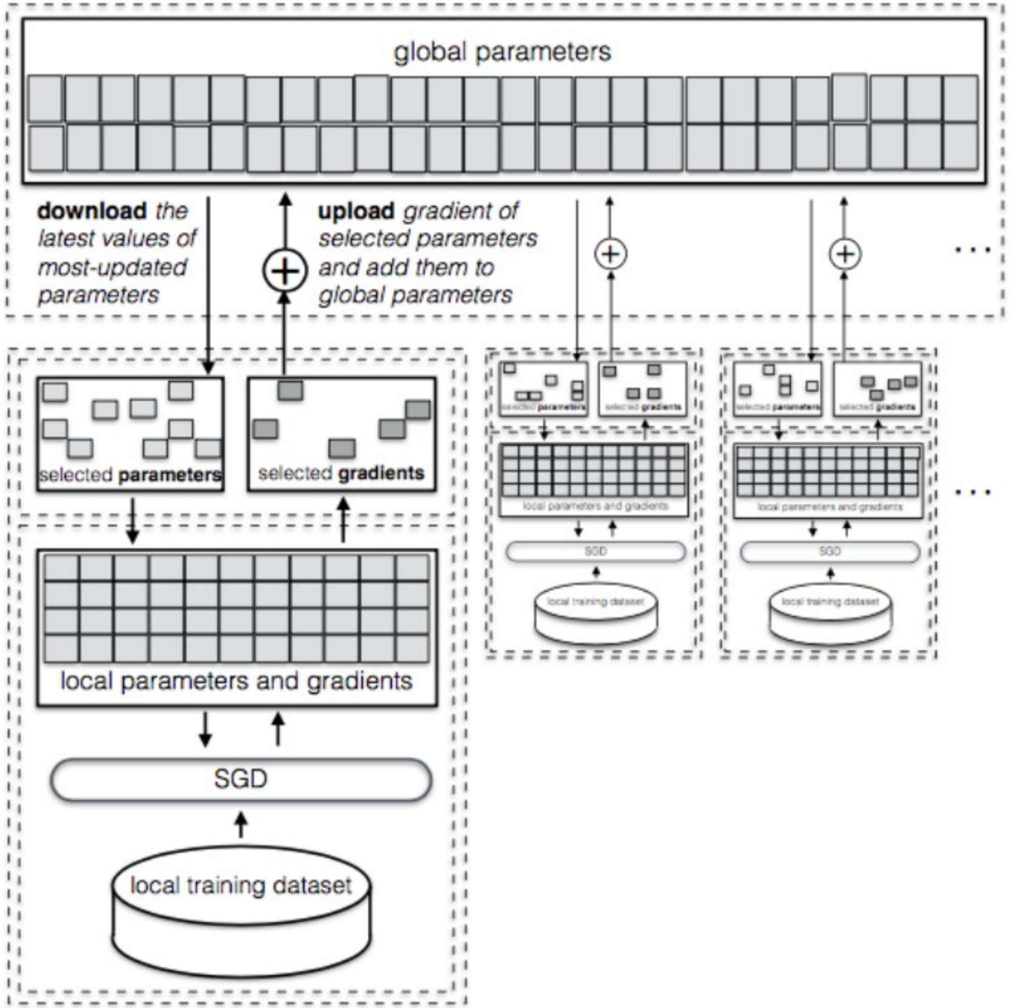
- Title: Privacy-preserving Deep Learning
  - Authors: Reza Shokri, Vitaly Shmatikov
  - In CCS 2015

- Highlights
  - Early system work in considering user data privacy for deep learning.
  - A mechanism called distributed selective SGD (DSSGD) is proposed.
  - Efforts in analysis and mitigation of privacy leakage, using differential privacy for privacy-preserving deep learning.

◆ Shokri, R., & Shmatikov, V. (2015, October). Privacy-preserving deep learning. In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (pp. 1310-1321).

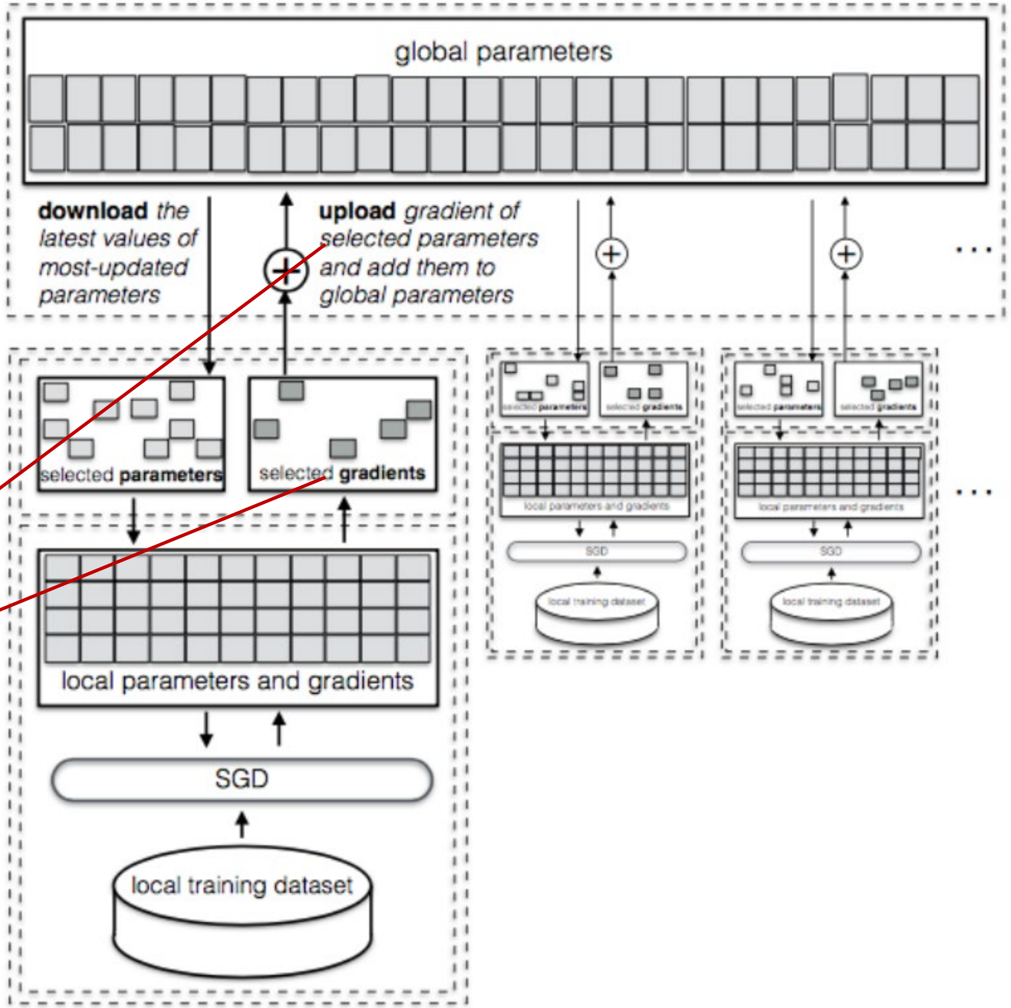
Case study 3: Privacy-preserving Deep Learning

- Private-by-design
  - Preventing direct leakage
    - while training - user do not reveal data to others
    - while using – user can use the model locally
  - Preventing indirect leakage – DP!
    - noise is added to gradients to prevent leakage of information related to local dataset



Case study 3: Privacy-preserving Deep Learning

- Private-by-design
  - Preventing direct leakage
    - while training - user do not reveal data to others
    - while using – user can use the model locally
  - Preventing indirect leakage – DP!
    - noise is added to gradients to prevent leakage of information related to local dataset



Potential privacy leakage:

1. How gradients are selected for sharing
2. The actual values of the shared gradients

→ SVT!

Case study 3: Privacy-preserving Deep Learning

- The algorithm for differentially private DSSGD for user *i*.
  - Sparse vector technique is used to:
    - (i) randomly select a small subset of gradients whose values are above a threshold, and then,
    - (ii) share perturbed values of the selected gradients in a differentially private manner.
  - Note that SVT here can be replaced by EM due to non-interactiveness.

- Let ϵ be the total privacy budget for one epoch of participant i running DSSGD, and let Δf be the sensitivity of each gradient
- Let $c = \theta_u |\Delta \mathbf{w}|$ be the maximum number of gradients that can be uploaded in one epoch
- Let $\epsilon_1 = \frac{8}{9}\epsilon$, $\epsilon_2 = \frac{2}{9}\epsilon$
- Let $\sigma(x) = \frac{2c\Delta f}{x}$

1. Generate fresh random noise $r_\tau \sim \text{Lap}(\sigma(\epsilon_1))$
2. Randomly select a gradient $\Delta w_j^{(i)}$
3. Generate fresh random noise $r_w \sim \text{Lap}(2\sigma(\epsilon_1))$
4. If $\mathsf{abs}(\mathsf{bound}(\Delta w_j^{(i)}, \gamma)) + r_w \geq \tau + r_\tau$, then
   (a) Generate fresh random noise $r'_w \sim \text{Lap}(\sigma(\epsilon_2))$
   (b) Upload $\mathsf{bound}(\Delta w_j^{(i)} + r'_w, \gamma)$ to the parameter server
   (c) Charge $\frac{\epsilon}{c}$ to the privacy budget
   (d) If number of uploaded gradients is equal to c, then Halt Else Goto Step 1
5. Else Goto Step 2