CS 325: Private Data Analysis **Fall 2022**

Lecture 8: Sparse Vector Technique

Lecturer: Di Wang *Scribes: Di Wang*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

Suppose we receive a sequence of k sensitivity-1 queries in an online setting: at timestep t, we receive a query $f : \mathcal{X}^n \mapsto \mathbb{R}$, and we have to answer $f_t(D)$ as best we can under the constraint of differential privacy. In this setting, all we can really do is run the Laplace mechanism on the queries as they come. If there are a total of k queries, then using basic composition, we would be able to answer all queries with accuracy roughly $O(\frac{k}{\epsilon})$. If we use advanced composition, we can do a bit better: $O(\frac{\sqrt{k}}{\epsilon})$. But the point is, both of these are polynomial in k, and this is unavoidable. Thus, when we have a very large number of questions to answer, even with the Advanced composition theorem, it is difficulty to yield a reasonable privacy guarantee.

Fortunately, in some situations we will consider a slightly easier question: we didnt necessarily want to answer all the queries, but only identify which ones were large. In this case, we can hope a gain over the naive analysis by discarding the numeric answer to queries that lie significantly below the threshold.

Why we may save privacy budget? Consider the Noisy-max mechanism, we add the same Laplacian noise to each of the query, but we only return the larges perturbed value. If we release all the answers, then the algorithm will be $k\epsilon$-DP. However, if we just return the largest one then it will be ϵ-DP! This is the same for the Exponential mechanism. However, those two mechanisms can be only used in the offline and non-interactive setting where f_i are given in advance. In this section, we will consider an online setting, where f_i arrives sequentially. You can also think it as selection problem in the online setting. Since the queries will be coming in an online manner, we cant hope to say whether the first query is going to be the largest one until we see the later queries. Instead, we try to answer which queries are greater than some pre-specified (and publicly known) threshold T. Thus, our problem is

We have a sequence of k sensitivity-1 queries $\{f_1, \cdots, f_k\}$, and the goal is to identify the first c queries (where $c \leq k$) which have value greater than some (public) threshold T.

In order to do this in the online setting, we will introduce the **sparse vector technique**.

The technique itself is simple, we just add noise and report only whether the noisy value exceeds the threshold, and we will show that the privacy degrades only with the number of queries with is above the threshold. In details, we will just run the Laplace mechanism for each query: however, rather than outputting the value of this query, we only return the bit corresponding to whether it is greater than or less than some threshold. **The tricky part is that we must not only add Laplace noise to the query result but also to the threshold T, and each (noisy) query result is compared to the resulting (noisy) threshold** $\hat{T}$. As we will see in the proof, this has the miraculous effect of privatizing the result of many queries simultaneously, thus saving significant amounts in our privacy budget.

It is notable that there are other versions of the Sparse Vector Technique. However, some of them are incorrect, see [1] for details.

We start from a simpler case where $c = 1$, that is we want to identify the first query which is above the threshold. Based on our previous idea we propose the Algorithm 1.

Algorithm 1 Input is a private database D, an adaptively chosen stream of sensitivity 1 queries $f_1, \ldots$, and a threshold T. Output is a stream of responses $a_1, \ldots$

AboveThreshold$(D, \{f_i\}, T, \epsilon)$
  Let $\hat{T} = T + \text{Lap}\left(\frac{2}{\epsilon}\right)$.
  for Each query i **do**
    Let $\nu_i = \text{Lap}(\frac{4}{\epsilon})$
    if $f_i(D) + \nu_i \geq \hat{T}$ **then**
      Output $a_i = \top$.
      Halt.
    else
      Output $a_i = \perp$.
    end if
  end for

Figure 8.1: The AboveThreshold method for finding the first large query in a stream.

Theorem 8.1 *Algorithm 1 is ϵ-DP.*

A very interesting part of this proof is how the two randomizations serve to preserve privacy in different parts of the algorithm. Roughly speaking, randomizing the threshold privatizes all but the last result, whereas the last one is privatized by the noise addition to the query.

Proof: Fix any two neighboring datasets $D \sim D'$. Let A denote the random variable representing the output of Algorithm 1 on D and A' for Algorithm 1 on D'. The output is some realization of these random variables $a \in \{\perp, \top\}^S$ with $S \leq k$ and has the form that for all $i < S$, $a_i = \perp$ and $a_S = \top$. There are two types of randomness, the noisy threshold $\hat{T}$ and the perturbation to each of the k queries $\{v_i\}_{i=1}^S$. Defining the maximum noisy value of any query $f_1, \cdots, f_{S-1}$ on D:

$$g(D) = \max_{i<T}(f_i(D) + v_i) \tag{8.1}$$

Note that fixing $v_1, \cdots, v_{S-1}$we have

$$\begin{aligned}
\mathbb{P}_{\hat{T}, v_S}[A = a] &= \mathbb{P}_{\hat{T}, v_S}[\hat{T} > g(D) \text{ and } f_S(D) + v_S \geq \hat{T}] \\
&= \mathbb{P}_{\hat{T}, v_S}(\hat{T} \in (g(D), f_S(D) + v_S]) \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathbb{P}[v_S = v]\mathbb{P}[\hat{T} = t]\mathbb{I}[t \in (g(D), f_S(D) + v]] dv dt
\end{aligned}$$

At this point, we perform a change of variables. The goal is to transform from D to D', so we define new

variables. Let $t = \hat{t} - g(D') + g(D)$ and $v = \hat{v} - g(D') + g(D) - f(D) + f(D')$ then we have

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathbb{P}[v_S = v]\mathbb{P}[\hat{T} = t]\mathbb{I}[t \in (g(D), f_S(D) + v]]dvdt \tag{8.2}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathbb{P}[v_S = v]\mathbb{P}[\hat{T} = t]\mathbb{I}[\hat{t} - g(D') + g(D) \in (g(D), f_S(D) + \hat{v} - g(D') + g(D)]]d\hat{v}d\hat{t} \tag{8.3}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathbb{P}[v_S = \hat{v} - g(D') + g(D) - f(D) + f(D')]\mathbb{P}[\hat{T} = \hat{t} - g(D') + g(D)]\mathbb{I}[\hat{t} \in (g(D'), f_S(D') + \hat{v}]]d\hat{v}d\hat{t} \tag{8.4}$$

$$\leq \exp(\epsilon)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathbb{P}[v_S = \hat{v}]\mathbb{P}[\hat{T} = \hat{t}]\mathbb{I}[\hat{t} \in (g(D'), f_S(D') + \hat{v}]]d\hat{v}d\hat{t} \tag{8.5}$$

$$= \mathbb{P}_{\hat{T},v_S}(\hat{T} \in (g(D'), f_S(D') + v_S]) \tag{8.6}$$

$$= \mathbb{P}_{\hat{T},v_S}[\hat{T} > g(D') \text{ and } f_S(D') + v_S \geq \hat{T}] \tag{8.7}$$

$$= \mathbb{P}_{\hat{T},v_S}[A' = a]. \tag{8.8}$$

Where (9.5) is due to that $|g(D') - g(D) + f(D') - f(D)| \leq 2$ (since $|g(D) - g(D')| \leq 1$ by definition), $|f_S(D') - f_S(D)| \leq 1$ and the density function of Laplacian distribution. Thus, for any fixed $\{v_1, \cdots, v_{S-1}\}$, we have $\mathbb{P}_{\hat{T},v_S}[A = a] \leq e^{\epsilon}\mathbb{P}_{\hat{T},v_S}[A' = a]$. Taking the integral w.r.t $\{v_1, \cdots, v_{S-1}\}$ we can proof the theorem. ■

Next, we reason about accuracy. Since we are not outputting the numerical values of the queries, it is not immediately obvious how to define our accuracy notion. We will say that, with high probability, our algorithm makes no mistakes. A mistake will be when the algorithm says a small query is larger than the threshold, or when a large query is smaller than the threshold. But since the answers to the queries are noisy, we must give the algorithm a bit of slack.

We can an algorithm is (α, β)-accurate, if when applied to a sequence of k queries, with probability at least $1 - \beta$, for all $i \in [k]$, if $a_i = \top$, then $f_i(D) \geq T - \alpha$, if $a_i = \perp$ then $f_i(D) \leq T + \alpha$.

Theorem 8.2 *Given a sequence of k queries where all but the last one are significantly smaller than the threshold (i.e., $i < k, f_i(D) \leq T - \alpha$). Then Algorithm 1 is (α, β)-accurate with $\alpha = \frac{8(\log k + \log \frac{2}{\beta})}{\epsilon}$.*

Observe that the error increases only logarithmically in the total number of queries k, as opposed to polynomial as we would have incurred with the naive Laplace mechanism, or if we were trying to output the values of the queries.

Proof: If $a_i = \top$ then we have

$$f_i(D) + v_i \geq \hat{T} \implies f_i(D) \geq T - |\hat{T} - T| - |v_i|.$$

For $|\hat{T} - T| = |\text{Lap}(\frac{2}{\epsilon})|$ we use the tail bound $\mathbb{P}(|\text{Lap}(\lambda)| \geq t\lambda) = \exp(-t)$.

$$\mathbb{P}(|\hat{T} - T| \geq \frac{\alpha}{2}) = \exp(-\frac{\epsilon\alpha}{4}). \tag{8.9}$$

For the term $|v_i|$ we have

$$\mathbb{P}(\max_i |v_i| \geq \frac{\alpha}{2}) \leq k\exp(-\frac{\epsilon\alpha}{8}). \tag{8.10}$$

Setting both these probabilities to be at most $\frac{\beta}{2}$ we have $\alpha \geq \frac{8(\log k + \log \frac{2}{\beta})}{\epsilon}$. ■

Now we focus on the general case c. The idea is to use the (advanced) composition theorem for each large query, that is using c times of the Above Threshold (Algorithm 1). See Algorithm 2 for details.

Algorithm 2 Input is a private database D, an adaptively chosen stream of sensitivity 1 queries $f_1, \ldots$, a threshold T, and a cutoff point c. Output is a stream of answers $a_1, \ldots$

Sparse$(D, \{f_i\}, T, c, \epsilon, \delta)$

If $\delta = 0$ **Let** $\sigma = \frac{2c}{\epsilon}$. **Else Let** $\sigma = \frac{\sqrt{32c \ln \frac{1}{\delta}}}{\epsilon}$
Let $\hat{T}_0 = T + \text{Lap}(\sigma)$
Let count $= 0$
for Each query i **do**
  Let $\nu_i = \text{Lap}(2\sigma)$
  if $f_i(D) + \nu_i \geq \hat{T}_{\text{count}}$ **then**
    Output $a_i = \top$.
    Let count $=$ count $+1$.
    Let $\hat{T}_{\text{count}} = T + \text{Lap}(\sigma)$
  else
    Output $a_i = \perp$.
  end if
  if count $\geq c$ **then**
    Halt.
  end if
end for

Figure 8.2: The AboveThreshold method for finding the first c large query in a stream.

Theorem 8.3 *ALgorithm 2 is (ϵ, δ) or ϵ-DP. Moreover, suppose we are given a sequence of k queries where only c are large (i.e., the number of $i \in [n]$ such that $f_i(D) \geq T - \alpha$ is at most c). Then if $\delta = 0$, Algorithm 2 is (α, β)-accurate for $\alpha = \frac{8c(\log k + \log \frac{2c}{\beta})}{\epsilon}$; if $\delta > 0$, Algorithm 2 is (α, β)-accurate for $\alpha = \frac{16\sqrt{2c}(\log k + \log \frac{2c}{\beta})}{\epsilon}$.*

The proof is just followed by the (Advanced) Composition Theorem and Theorem 9.2.

We see that the dependence on the number of large values c is polynomial, and on total number of queries k is logarithmic, allowing us to detect large queries from a big set with good accuracy, as long as not too many of them are large. We note that there is also a variant of sparse vector, which we refer to as NumericSparse (Algorithm 3), which can output the (approximate) value of everything that is above the threshold at the cost of a small multiplicative factor in α. Specifically, if the algorithm is going to output $\top$ for a query, it instead adds **fresh Laplace noise** to the value of the query and outputs the result. The output of the algorithm will now be a stream of values in $\{\perp\} \cap \mathbb{R}$ with the following guarantees

- If $a_i = \perp$ then $f_i(D) \leq T + \alpha$
- If $a_i \in \mathbb{R}$, then $|f_i(D) - a_i| \leq \alpha$

The value of α is the same as in Theorem 9.3, up to a constant factor.

The intuition behind this modification: those whole idea behind the sparse vector technique is to pay logarithmically in the total number of queries, but polynomially in the number which are above the threshold. Since we already pay for these queries anyway, we can afford to do another private operation for each of them without changing the overall privacy cost by more than a constant factor.

References

[1] Min Lyu, Dong Su, and Ninghui Li. Understanding the sparse vector technique for differential privacy. *Proceedings of the VLDB Endowment*, 10(6), 2017.

Algorithm 3 Input is a private database D, an adaptively chosen stream of sensitivity 1 queries $f_1, \ldots$, a threshold T, and a cutoff point c. Output is a stream of answers $a_1, \ldots$

NumericSparse$(D, \{f_i\}, T, c, \epsilon, \delta)$

If $\delta = 0$ **Let** $\epsilon_1 \leftarrow \frac{8}{9}\epsilon$, $\epsilon_2 \leftarrow \frac{2}{9}\epsilon$. **Else Let** $\epsilon_1 = \frac{\sqrt{512}}{\sqrt{512}+1}\epsilon, \epsilon_2 = \frac{2}{\sqrt{512}+1}$

If $\delta = 0$ **Let** $\sigma(\epsilon) = \frac{2c}{\epsilon}$. **Else Let** $\sigma(\epsilon) = \frac{\sqrt{32c \ln \frac{2}{\delta}}}{\epsilon}$

Let $\hat{T}_0 = T + \text{Lap}(\sigma(\epsilon_1))$

Let count $= 0$

for Each query i **do**

  Let $\nu_i = \text{Lap}(2\sigma(\epsilon_1))$

  if $f_i(D) + \nu_i \geq \hat{T}_{\text{count}}$ **then**

    Let $v_i \leftarrow \text{Lap}(\sigma(\epsilon_2))$

    Output $a_i = f_i(D) + v_i$.

    Let count $=$ count $+1$.

    Let $\hat{T}_{\text{count}} = T + \text{Lap}(\sigma(\epsilon_1))$

  else

    Output $a_i = \perp$.

  end if

  if count $\geq c$ **then**

    Halt.

  end if

end for

Figure 8.3: The AboveThreshold method for finding the first c large query in a stream.