DP-ML Proofs

October 7, 2019

1 Privacy Loss when adding $\epsilon_1 + \epsilon_2$ noise

Let:

$A(x) => (\epsilon_1, \delta)$ differentially private mechanism by adding noise from $N(0, \sigma_1^2)$
$B(x) => (\epsilon_2, \delta)$ differentially private mechanism by adding noise from $N(0, \sigma_2^2)$

Theorem 1.1. *If C(x) is a dp-mechanism that adds the sum of noise sampled from $N(0, \sigma_1^2)$ and $N(0, \sigma_2^2)$, then C(x) is $\sqrt(\frac{\epsilon_1^2 * \epsilon_2^2}{\epsilon_1^2 + \epsilon_2^2})$ differentially private.*

Proof. We know that the sum of two normal distributed random variable is also normal $=> N(0, \sigma_1^2) + N(0, \sigma_2^2) = N(0, \sigma_1^2 + \sigma_2^2)$

Therefore, summing up noise from two randomly distributed variables is equivalent to sampling noise from $N(0, \sigma_1^2 + \sigma_2^2 = \sigma_3^2)$

From [2] it follows that, if σ is equivalent to

$$\frac{s}{\epsilon}\sqrt(2ln\frac{1.25}{\delta})$$

then a step is (ϵ, δ) differentially private.

We know:

$$\sigma_3^2 = \sigma_1^2 + \sigma_2^2$$
$$\sigma_3^2 = \frac{s^2}{\epsilon_1^2}(2ln\frac{1.25}{\delta}) + \frac{s^2}{\epsilon_1^2}(2ln\frac{1.25}{\delta})$$
$$\sigma_3^2 = (2ln\frac{1.25}{\delta})(\frac{s^2}{\epsilon_1^2} + \frac{s^2}{\epsilon_1^2})$$
$$\frac{s^2}{\epsilon_3^2}(2ln\frac{1.25}{\delta}) = (2ln\frac{1.25}{\delta})(\frac{s^2}{\epsilon_1^2} + \frac{s^2}{\epsilon_2^2})$$
$$\frac{s^2}{\epsilon_3^2}(2ln\frac{1.25}{\delta}) = (2ln\frac{1.25}{\delta})(\frac{s^2}{\epsilon_1^2} + \frac{s^2}{\epsilon_2^2})$$
$$\frac{s^2}{\epsilon_3^2}(2ln\frac{1.25}{\delta}) = (2ln\frac{1.25}{\delta})(\frac{s^2}{\epsilon_1^2} + \frac{s^2}{\epsilon_2^2})$$

$$\frac{1}{\epsilon_3^2} = \frac{1}{\epsilon_1^2} + \frac{1}{\epsilon_2^2}$$
$$\frac{1}{\epsilon_3^2} = \frac{\epsilon_1^2 + \epsilon_2^2}{\epsilon_1^2 * \epsilon_2^2}$$
$$\epsilon_3^2 = \frac{\epsilon_1^2 * \epsilon_2^2}{\epsilon_1^2 + \epsilon_2^2}$$
$$\epsilon_3 = \sqrt{}(\frac{\epsilon_1^2 * \epsilon_2^2}{\epsilon_1^2 + \epsilon_2^2})$$

■

Here are a few examples for the resulting epsilon value when you add two noise vectors each satisfying (ϵ_1, δ) and (ϵ_2, δ) respectively:

ϵ_1	ϵ_2	ϵ_3
0.5	0.75	0.416
1.0	0.5	0.447
2.0	1.0	0.894
2.0	0.5	0.485

MS ►*Not sure if the above result can be used for $\epsilon > 1$ values since $\frac{s}{\epsilon}\sqrt{}(2ln\frac{1.25}{\delta})$ might only applicable for cases where both $\epsilon_1 < 1$ and $\epsilon_2 < 1$ [2]*◄

2 Privacy guarantee for secure aggregation

Theorem 2.1. *If S(x) is a dp-mechanism that uses a cryptographic protocol to securely-aggregate the output of n (A(x)) mechanisms each satisfying (ϵ, δ)-dp, then C(x) is $\frac{\epsilon}{\sqrt{}(n)}$ private.*

Let $A(x)$ => (ϵ, δ) differentially private mechanism by adding noise from $N(0, \sigma_a^2)$

Let $S(x)$ be (ϵ_s, δ) differentially private by sampling noise from $N(0, \sigma_s)$

Proof. Since $S(x) = \sum_n (A(x))$:

$$\sigma_s^2 = n * \sigma_a^2$$
$$\frac{s^2}{\epsilon_s^2}(2ln\frac{1.25}{\delta}) = \frac{s^2}{\epsilon_a^2}(2ln\frac{1.25}{\delta}) * n$$
$$\frac{1}{\epsilon_s^2} = \frac{n}{\epsilon_a^2}$$
$$\epsilon_s^2 = \frac{\epsilon_a^2}{n}$$
$$\epsilon_s = \frac{\epsilon_a}{sqrt(n)}$$

■

3 Privacy guarantee when sampling from with/without secure aggregation

Let:

$A(x) => (\epsilon_1, \delta)$ differentially private mechanism by adding noise from $N(0, \sigma_1^2)$
$B(x) => (\epsilon_2, \delta)$ differentially private mechanism via secure aggregation by adding noise from $N(0, \sigma_2^2 * \sqrt{n})$ where $\epsilon_2 = \epsilon_1/sqrt(n)$

The goal is to find the (ϵ_3, δ) guarantee of a mechanism $C(x)$ that is (ϵ_1, δ) differentially private with probability p and (ϵ_3, δ) differentially private with probability (1-p).

We need to bound the ratio:

$$\frac{P[C(x) \in S] - \delta}{P[C'(x') \in S]}$$
$$\frac{p * P[A(x) \in S] + (1-p) * P[B(x) \in S] - \delta}{p * P[A(x') \in S] + (1-p) * P[B(x') \in S]}$$

Let:

$A = P[A(x) \in S]$
$B = P[B(x) \in S]$
$A' = P[A(x') \in S]$
$B' = P[B(x') \in S]$

$$\begin{aligned}
&\frac{p*A+(1-p)*B-\delta}{p*A'+(1-p)*B'} \\
&= \frac{p*A+(1-p)*B-(p*\delta+(1-p)*\delta)}{p*A'+(1-p)*B'} \\
&= \frac{p*(A-\delta)+(1-p)*(B-\delta)}{p*A'+(1-p)*B'} \\
&= \frac{p*(A-\delta)+(1-p)*(B-\delta)}{p*A'+(1-p)*B'} \\
&= \frac{p*(A-\delta)}{p*A'+(1-p)*B'}+\frac{(1-p)*(B-\delta)}{p*A'+(1-p)*B'} \\
&= \frac{p*(A-\delta)}{p*A'}*\frac{1}{1+\frac{(1-p)B'}{p(A')}}+\frac{(1-p)*(B-\delta)}{(1-p)*B'}*\frac{1}{1+\frac{p*A'}{(1-p)(B')}} \\
&= e^{\epsilon_1}*\frac{p(A')}{(1-p)B'+p(A')}+e^{\frac{\epsilon_1}{sqrt(n)}}*\frac{(1-p)B'}{p(A')+(1-p)(B')} \\
&= e^{\epsilon_1}*\frac{p(A')}{(1-p)B'+p(A')}+e^{\frac{\epsilon_1}{sqrt(n)}}*\frac{(1-p)B'}{p(A')+(1-p)(B')}
\end{aligned}$$

Need to express numerator in the following form:

$$e^x(p(A')+(1-p)(B'))$$

where x is the epsilon-guarantee of combining two epsilons.

3.1 No assumption about A' and B'

$$<= \frac{e^{\epsilon_1}(p(A')+(1-p)(B'))}{p(A')+(1-p)(B')}$$

Guarantee = (ϵ_1, δ)

3.2 A' = B'

$$
\begin{aligned}
&<= \frac{e^{\epsilon_1} * p * A' + e^{\frac{\epsilon_1}{\sqrt(n)}} * (1-p)(A')}{p(A') + (1-p)(A')} \\
&<= \frac{A' * (e^{\epsilon_1} * p + e^{\frac{\epsilon_1}{\sqrt(n)}} * (1-p))}{A'} \\
&<= (e^{\epsilon_1} * p + e^{\frac{\epsilon_1}{\sqrt(n)}} * (1-p)) \\
&<= e^{\epsilon_1 + ln(p)} + e^{\frac{\epsilon_1}{\sqrt(n)} + ln(1-p)} \\
&<= e^{\epsilon_1 + ln(p) + \frac{\epsilon_1}{\sqrt(n)} + ln(1-p)}
\end{aligned}
$$

bcoz $e^x + e^y <= e^{x+y}$ where $x >= 0$ and $y >= 0$

Hence above bound valid only if $\frac{\epsilon_1}{\sqrt(n)} + ln(1-p) > 0$ and $\epsilon_1 + ln(p) > 0$

4 Next Steps/TODOs

1. I am thinking of coming up with a design that is close to the Encode/Shuffle/Analyze [1] architecture here. It seems to strike a balance between the local/global dp utility tradeoff that we are trying to optimize. The idea is to have a shuffler service as an intermediary, that minimizes the data points exposed to the coordinator/analyzer. The hard part would is that the shuffler shouldn't be able to look at the data points but should be able to group them together into say malicious/non malicious using some metadata. There is prior work on similarity preserving hashes that could come in handy here to allow the shuffler to do that.

2. I also want to figure out a theoretical upper bound for the std dev of the noise that can be added to KRUM without breaking the guarantee. I will have to revisit proofs for Multi-KRUM.

3. Keep thinking if there is a tighter bound for the above mechanism.

References

[1] Bittau, A., Úlfar Erlingsson, Maniatis, P., Mironov, I., Raghunathan, A., Lie, D., Rudominer, M., Kode, U., Tinnes, J., and Seefeld, B. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the Symposium on Operating Systems Principles (SOSP)* (2017), pp. 441–459.

[2] Dwork, C., and Roth, A. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science 9*, 3-4 (2014).