A CODE FOR REPRODUCIBILITY

See `https://anonymous.4open.science/r/shuffgauss-6D7C` for the code of our implementation.

B ADDITIONAL PROOFS

B.1 PROOF OF THEOREM 1

Here, we give the full proof of Theorem 1.

Proof. As explained in the main text, the neighboring databases of interest are

$$\mathcal{M}(D) \sim \mathcal{N}(0, \sigma^2 I_n),$$
$$\mathcal{M}(D') \sim \frac{1}{n}\left(\mathcal{N}(e_1, \sigma^2 I_n) + \ldots + \mathcal{N}(e_n, \sigma^2 I_n)\right).$$

We are concerned with calculating the following quantity, $\mathbb{E}_{x\sim\mathcal{M}(D)}\left[\left(\frac{\mathcal{M}(D')(x)}{\mathcal{M}(D)(x)}\right)^\lambda\right]$ for RDP:

$$\begin{aligned}
&\mathbb{E}_{x\sim\mathcal{M}(\mathcal{D})}\left[\left(\frac{\mathcal{M}(D')(x)}{\mathcal{M}(D)(x)}\right)^\lambda\right] \\
&= \int \left(\frac{\exp\left[-(x_1-1)^2/2\sigma^2 - \sum_{i=2}^n x_i^2/2\sigma^2\right] + \ldots}{n\exp[-\sum_{i=1}^n x_i^2/2\sigma^2]}\right)^\lambda \exp[-\sum_{i=1}^n x_i^2/2\sigma^2]\frac{d^n x}{(2\pi\sigma^2)^{n/2}} \quad (14)\\
&= \int \left(\frac{\exp\left[-(x_1-1)^2/2\sigma^2 + x_1^2/2\sigma^2\right] + \ldots}{n}\right)^\lambda \exp[-\sum_{i=1}^n x_i^2/2\sigma^2]\frac{d^n x}{(2\pi\sigma^2)^{n/2}} \quad (15)\\
&= \int \left(\exp\left[(2x_1-1)/2\sigma^2\right] + \ldots\right)^\lambda \exp[-\sum_{i=1}^n x_i^2/2\sigma^2]\frac{d^n x}{n^\lambda(2\pi\sigma^2)^{n/2}} \quad (16)\\
&= \int \left(\sum_{i=1}^n \exp\left[(2x_i-1)/2\sigma^2\right]\right)^\lambda \exp[-\sum_{i=1}^n x_i^2/2\sigma^2]\frac{d^n x}{n^\lambda(2\pi\sigma^2)^{n/2}}. \quad (17)
\end{aligned}$$

Let us explain the above calculation in detail. We first notice that $\mathbb{E}_{x\sim\mathcal{M}(\mathcal{D})}\left[\left(\frac{\mathcal{M}(D')(x)}{\mathcal{M}(D)(x)}\right)^\lambda\right]$ is an n-th dimensional integral of x_i ($i \in [n]$), as shown in Equation 14. For each term $j \in [n]$ in the nominator of $(\ldots)^\lambda$, we divide it by the denominator $\exp[-\sum_{i=1}^n x_i^2/2\sigma^2]$, yielding $\exp[-(x_j-1)^2/2\sigma^2 - \sum_{i\neq j}^n x_i^2/2\sigma^2 + \sum_{i=1}^n x_i^2/2\sigma^2] = \exp[-(x_j-1)^2/2\sigma^2 + x_j^2/2\sigma^2]$, with all x_i terms in $i \in [n]$ except j canceled out (Equation 15). The term can be further simplified to $\exp[(2x_j-1)/2\sigma^2]$ as in Equation 17.

Then, we expand the expression $(\ldots)^\lambda$ using the multinomial theorem:

$$\left(\sum_{i=1}^n \exp\left[(2x_i-1)/2\sigma^2\right]\right)^\lambda = \sum_{\substack{k_1+\ldots+k_n=\lambda;\\ k_1,\ldots,k_n\geq 0}} \binom{\lambda}{k_1,\ldots,k_n} \prod_{i=1}^n \exp\left[k_i(2x_i-1)/2\sigma^2\right], \quad (18)$$

where $\binom{\lambda}{k_1,\ldots,k_n} = \frac{\lambda!}{k_1!k_2!\ldots k_n!}$ is the multinomial coefficient, and $k_i \in \mathbb{Z}^+$ for $i \in [n]$.

Before proceeding, we make a detour to calculate the following integral (with $k \in \mathbb{Z}^+$):

$$\begin{aligned}
&\int \exp[k(2x-1)/2\sigma^2]\exp[-x^2/2\sigma^2]\frac{dx}{\sqrt{2\pi\sigma^2}} \\
&= \int \exp[-(x-k)^2/2\sigma^2 + (k^2-k)/2\sigma^2]\frac{dx}{\sqrt{2\pi\sigma^2}}
\end{aligned}$$

$$= \exp[(k^2 - k)/2\sigma^2].$$

Using the above expression and Equation 18, we can write Equation 17 as

$$\begin{aligned}
&\int \left(\sum_{i=1}^{n} \exp\left[(2x_i - 1)/2\sigma^2\right] \right)^{\lambda} \exp[-\sum_{i=1}^{n} x_i^2/2\sigma^2] \frac{d^n x}{n^\lambda (2\pi\sigma^2)^{n/2}} \\
&= \int \sum_{\substack{k_1+\ldots+k_n=\lambda; \\ k_1,\ldots,k_n \geq 0}} \binom{\lambda}{k_1, \ldots, k_n} \prod_{i=1}^{n} \exp\left[k_i(2x_i - 1)/2\sigma^2\right] \frac{\exp[-x_i^2/2\sigma^2] d^n x}{n^\lambda (2\pi\sigma^2)^{n/2}} \\
&= \frac{1}{n^\lambda} \sum_{\substack{k_1+\ldots+k_n=\lambda; \\ k_1,\ldots,k_n \geq 0}} \binom{\lambda}{k_1, \ldots, k_n} \prod_{i=1}^{n} \exp[(k_i^2 - k_i)/2\sigma^2], \qquad (19)
\end{aligned}$$

where we have moved the $1/n^\lambda$ factor to the front. Noticing that $\prod_{i=1}^{n} \exp[-k_i] = \exp[-\sum_i^n k_i]$ and that $\sum_{i=1}^{n} k_i = \lambda$ under the multinomial constraint, we have

$$\begin{aligned}
&\frac{1}{n^\lambda} \sum_{\substack{k_1+\ldots+k_n=\lambda; \\ k_1,\ldots,k_n \geq 0}} \binom{\lambda}{k_1, \ldots, k_n} \prod_{i=1}^{n} \exp[(k_i^2 - k_i)/2\sigma^2] \\
&= \frac{e^{-\lambda/2\sigma^2}}{n^\lambda} \sum_{\substack{k_1+\ldots+k_n=\lambda; \\ k_1,\ldots,k_n \geq 0}} \binom{\lambda}{k_1, \ldots, k_n} e^{\sum_{i=1}^{n} k_i^2/2\sigma^2}.
\end{aligned}$$

Combining the above expression with Equation 2, we obtain the expression given in Equation 4 as desired. ■

B.2 Proof of Theorem 2

Proof. Theorem 9 of Wang et al. (2019) states that for subsampling rate γ, a mechanism $\mathcal{M}$ satisfying $(\lambda, \epsilon(\lambda))$-RDP applied to a subsampled set of data, satisfy $(\lambda, \epsilon'(\lambda))$, where

$$\begin{aligned}
\epsilon'(\lambda) \leq \frac{1}{\lambda - 1} \log[1 + \gamma^2 \binom{\lambda}{2} \min\left\{4(e^{\epsilon_m(2)} - 1), e^{\epsilon_m(2)} \min\{2, (e^{\epsilon(\infty)-1})^2\}\right\} \\
+ \sum_{j=3}^{\lambda} \gamma^j \binom{\lambda}{j} e^{(j-1)\epsilon_m(j)} \min\{2, (e^{\epsilon(\infty)-1})^j\}]
\end{aligned}$$

Since $\epsilon(\infty)$ is unbounded, we can simplify it to the expression given in Theorem 2 in a straightforward way. ■

B.3 Proof of Theorem 3

The following lemma is useful:

Lemma 3. *Let f(k) be any monotonically decreasing function with respect to k for $k \in \mathbb{N}$, $1 \leq k \leq n$ and $n \in \mathbb{N}$. Also let $0 \leq \gamma \leq 1$. Then, the following inequality holds:*

$$\sum_{k=1}^{n} \binom{n}{k} f(k) \leq f(1) e^{-\Delta^2 \mu/2} + f((1 - \Delta)\mu + 1).$$

where $\mu = \gamma n$, Δ is an arbitrary number conditioned on $(1 - \Delta)\mu$ being integer and $0 \leq \Delta \leq 1$.

Proof. We split the summation over k to two parts: from $k = 1$ to k equal or less than $(1 - \Delta)n\gamma$, and those equal or larger than $(1 - \Delta)n\gamma + 1$. Note that the Chernoff bound states that: $\mathbb{P}[X \leq (1 - \Delta)\mu] \leq e^{-\Delta^2 \mu/2}$ for all $0 < \Delta < 1$. Then,

$$\sum_{k=1}^{n} \binom{n}{k} f(k) = \sum_{k_1=1}^{(1-\Delta)\mu} \binom{n}{k_1} f(k_1) + \sum_{k_2=(1-\Delta)\mu+1}^{n} \binom{n}{k_2} f(k_2)$$

$$
\begin{aligned}
&\leq \sum_{k_1=1}^{(1-\Delta)\mu} \binom{n}{k_1} f(1) + \sum_{k_2=(1-\Delta)\mu+1}^{n} \binom{n}{k_2} f(k_2) \\
&\leq \mathbb{P}[k \leq (1-\Delta)\mu] f(1) + \sum_{k_2=(1-\Delta)\mu+1}^{n} \binom{n}{k_2} f(k_2) \\
&\leq f(1) e^{-\Delta^2 \mu/2} + \sum_{k_2=(1-\Delta)\mu+1}^{n} \binom{n}{k_2} f(k_2) \\
&\leq f(1) e^{-\Delta^2 \mu/2} + \sum_{k_2=(1-\Delta)\mu+1}^{n} \binom{n}{k_2} f((1-\Delta)\mu+1) \\
&\leq f(1) e^{-\Delta^2 \mu/2} + f((1-\Delta)\mu+1)
\end{aligned}
$$

■

Proof of Theorem 3. Note that $f(k) = e^{(\lambda-1)\epsilon_{\gamma,k}^{SSG}}$ satisfies the condition of $f(k)$ in Lemma 3. Then, by applying Lemma 3, the first part of the summation

$$\sum_{k=1}^{n} \binom{n}{k} \gamma^k (1-\gamma)^{n-k} e^{(\lambda-1)\epsilon_{\gamma,k}^{\mathrm{SSG}}(\lambda)}$$

can be bounded by

$$e^{-\Delta^2 n\gamma/2 + (\lambda-1)\epsilon_{\gamma,1}^{\mathrm{SSG}}(\lambda)},$$

while the second part of the summation can bounded by

$$e^{(\lambda-1)\epsilon_{\gamma,(1-\Delta)n\gamma+1}^{\mathrm{SSG}}(\lambda)}$$

Combining the above two terms gives the desired expression. ■

The main advantage of using Equation 13 is that the calculation of the RDP with this theorem is $O(n)$ times more efficient computationally compared to Equation 11. Note that only two terms are calculated in the logarithm of Equation 13. One can potentially calculate more terms (but still less than $O(n)$) to make the approximation better. We leave it for future work as from Figure 1b, Equation 13 is sufficiently tight compared to existing techniques.

C Algorithms

C.1 GetUniqueCount: Note on the permutation invariance of Equation 9

It is best to describe the counting factor in our numerical evaluation of Equation 9 via an example.

Consider again $(x_1 + x_2 + x_3)^3$. Expanding the multinomials, the terms with exponents of the form $x_i^2 x_j$ are

$$3(x_1^2 x_2 + x_1^2 x_3 + x_2^2 x_1 + x_2^2 x_3 + x_3^2 x_1 + x_3^2 x_2).$$

Here, the factor 3 comes from the multinomial coefficient $\binom{3}{2,1,0}$. These terms $x_1^2 x_2, x_1^2 x_3, x_2^2 x_1, x_2^2 x_3, x_3^2 x_1, x_3^2 x_2$ belong to the same subset of the form $x_i^2 x_j$, which do not have repetition of exponent with same degrees. It has $3! = 6$ elements in total. This subset contributes effectively a factor of $3 \times 6 = 18$ to the expansion in Equation 9,

$$18 e^{(2^2+1^2)/2\sigma^2},$$

ignoring factor unrelated to the multinomial coefficients.

Consider the same expansion, but the terms with exponents of the form $x_i x_j x_k$:

$$6 x_1 x_2 x_3.$$

Table 2: Neural network architecture for MNIST.

Layer	Parameters
Convolution	16 filters of 8×8 , strides 2
Max-pooling	2×2
Convolution	32 filters of 4×4, strides 2
Max-pooling	2×2
Linear	32 units
Softmax	10 units

Here, the multinomial coefficient is $\binom{3}{1,1,1} = 6$. Since the number of repetition of exponent with the same degree is 3 (x_i, x_j, x_k has the same degree 1), the subset has $3!/(3!) = 1$ element. This subset contributes effectively a factor of $6 \times 1 = 6$ to the expansion in Equation 9,

$$6e^{(1^1+1^2)/2\sigma^2},$$

ignoring factor unrelated to the multinomial coefficients.

This procedure of calculating the contributing coefficients is called `GetUniqueCount` in Algorithm 1.

C.2 More on Algorithm 1

We describe our algorithm for evaluating Equation 4.

Given λ, we find all partition of integers satisfying $k_1, \ldots, k_n = \lambda$. We denote the operation by `GetPartition`. For each of the partition, we obtain the number of counts for each unique k_i; let them be $\kappa_1, \ldots, \kappa_i$ ($i \leq n$), and denote the operation by `GetUniqueCount`. Subsequently, we calculate the value:

$$\frac{e^{-\lambda/2\sigma^2}}{n^\lambda} \binom{\lambda}{k_1 \ldots k_n} \frac{n!}{\kappa_1! \ldots \kappa_i!} e^{\sum_{j=1}^n k_j^2},$$

and make summation over all the partitions. The algorithm is shown in Algorithm 1.

D Additional Experimental Details

Network architecture. In Table 2, we present the neural network we use for the MNIST experiment presented in Section 5.

Simulation details. We note that some of the experiments of interest can be simulated conveniently by making simple modifications to existing libraries that implement DP-SGD (e.g., `Opacus` Yousefpour et al. (2021)). This is done by noticing that in DP-SGD, given a batch of clipped gradients $\tilde{g}$, Gaussian noise is applied to the sum of the gradients, i.e., $\sum_{i=1}^B \tilde{g}_i \leftarrow \sum_{i=1}^B \tilde{g}_i + \mathcal{N}(0, \sigma^2)$. Here, B is the number of samples. In shuffle Gaussian or local Gaussian, the server sums over the received gradients randomized with Gaussian noises, $\sum_{i=1}^B \{\tilde{g}_i + \mathcal{N}(0, \sigma^2)\} = \sum_{i=1}^B \tilde{g}_i + \mathcal{N}(0, B\sigma^2)$. Therefore, we can simply multiply σ by a factor of $\sqrt{B}$ in existing DP-SGD libraries to simulate the decentralized setting of shuffle/local Gaussian.

D.1 Additional Results

RDP versus approximate DP. In addition to those presented in Section 5.1, let us plot the ϵ dependence at larger number of composition. In Figure 2, we show the RDP and upper bound, again demonstrating significant amplification due to shuffling.

DP-SGD. In Figure 3, we show how varying batch sizes affect the privacy budget for both the shuffle and central-DP models. It is shown that smaller batch size leads to higher accuracy. Note that while smaller batch size reduces the effect of privacy amplification by shuffling, it is partially recovered by privacy amplification due to subsampling. Overall, batch size does not play an important role in privacy accounting, although tuning is essential at attaining optimal accuracy.

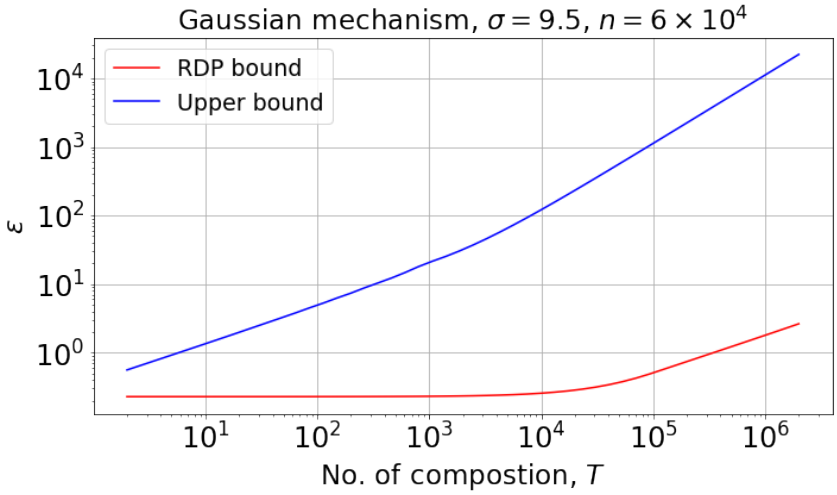

Figure 2: Shuffling effects.

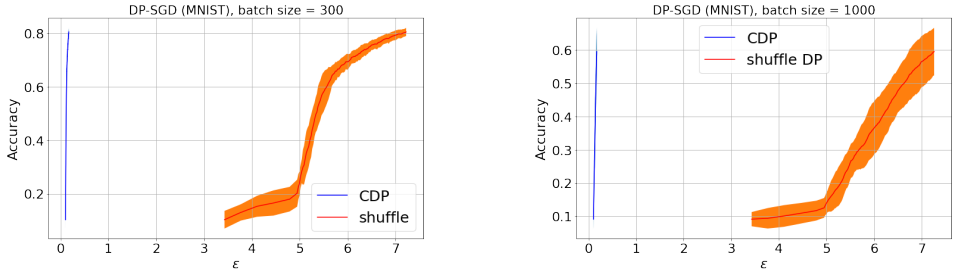

Figure 3: Accuracy-privacy trade-offs for the shuffle and central-DP models when varying batch size.

In Figure 4, we show how varying clipping size/learning rate affects the privacy budget for both the shuffle and central-DP models. These are hyperparameters that have to be fine-tuned to achieve reasonable accuracy. Experiments are repeated 5 times with different random seeds and the mean as well as standard deviation are shown in the figures.

D.2 DP-SGD WITH CIFAR10

We further perform DP-SGD experiments on the CIFAR10 dataset. Here, we first pre-train a convolutional neural network with the CIFAR100 dataset *without* applying DP-SGD, assuming that CIFAR100 is public. Then, the final layer is fine-tuned to train CIFAR10 with DP-SGD.

The parameters are set as follows: batch size varying from 50 to 1,000, $\delta = 10^{-5}$, noise multiplier $\sigma = 0.86$ (corresponding to $(\epsilon_0, \delta_0) = (12, 10^{-5})$), clipping size $C = 0.05$, learning rate 0.05.

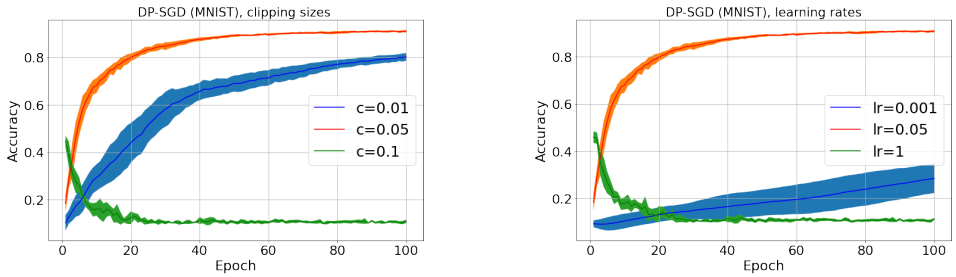

Figure 4: Performance of training MNIST when varying clipping size (left) and learning rate (right).

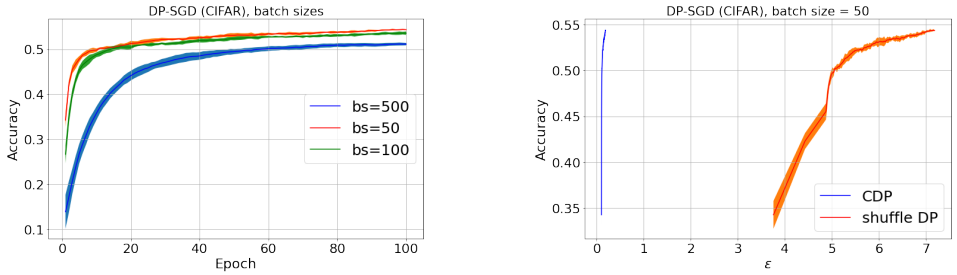

Figure 5: Performance when varying batch size (left) and privacy budget when training with batch-size 50 (right).

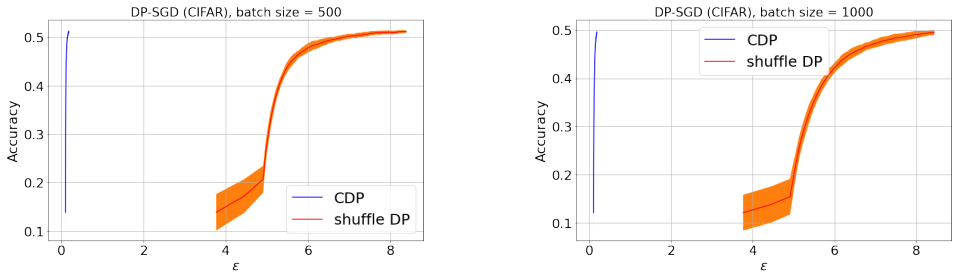

Figure 6: Accuracy-privacy trade-offs for training CIFAR10 when batch size is 500 (left) and 1,000 (right).

In Figures 5 and 6, we show how varying batch sizes affects the privacy budget spent. Again, smaller batch size leads to better performances. Also, optimal accuracy (55%) is attainable with intermediate levels of privacy, $\epsilon \leq 10$. Note again that experiments are repeated 5 times with different random seeds and the mean as well as standard deviation are shown in the figures.