Private Adaptive Optimization with Side Information

Tian Li
(CMU)

Manzil Zaheer
(DeepMind)

Sashank Reddi
(Google Research)

Virginia Smith
(CMU)

Motivation

Motivation

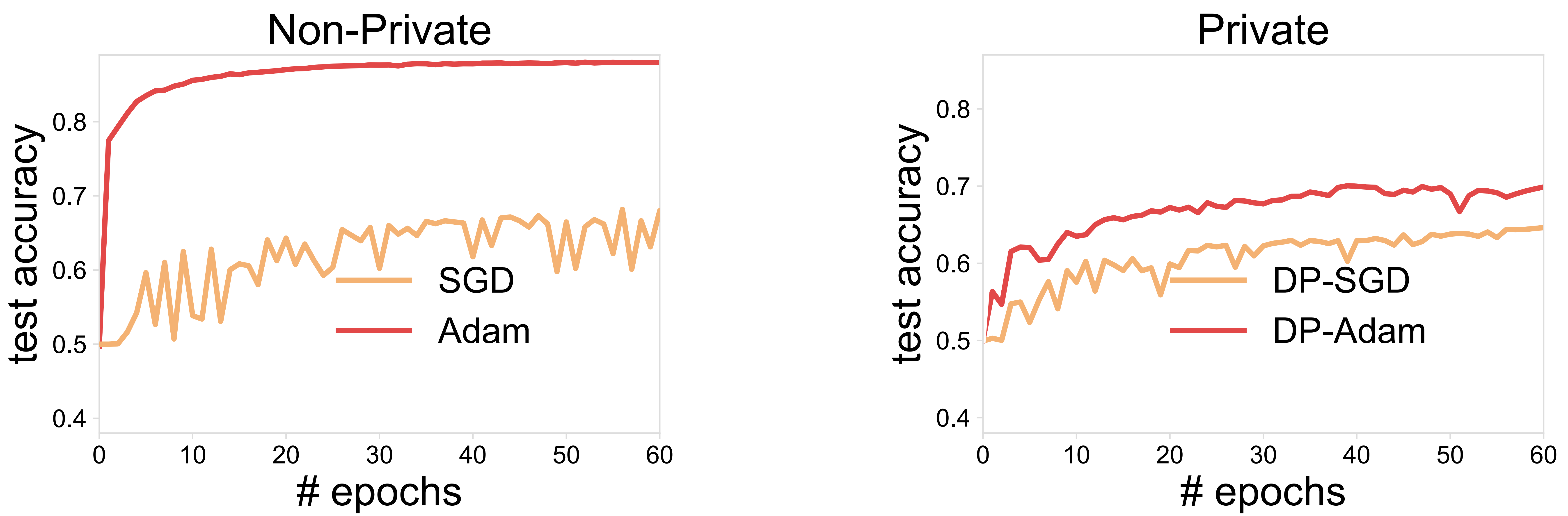
- Adaptive optimizers (e.g., Adam, AdaGrad, RMSProp) are useful for a variety of ML tasks

Motivation

- Adaptive optimizers (e.g., Adam, AdaGrad, RMSProp) are useful for a variety of ML tasks
- However, performance may degrade significantly when trained with differential privacy (DP), especially when the model dimension is large

Motivation

- Adaptive optimizers (e.g., Adam, AdaGrad, RMSProp) are useful for a variety of ML tasks
- However, performance may degrade significantly when trained with differential privacy (DP), especially when the model dimension is large



How to effectively adapt to the geometry of gradients under DP?

How to effectively adapt to the geometry of gradients under DP?

directly plug in private gradients to estimate the statistics ?

How to effectively adapt to the geometry of gradients under DP?

directly plug in private gradients to estimate the statistics ?

first privatize the gradients

$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(g^{i,t}, C\right) + \mathcal{N}\left(0,\sigma^2 C^2\right)\right)$$

How to effectively adapt to the geometry of gradients under DP?

directly plug in private gradients to estimate the statistics ?

first privatize the gradients

$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(g^{i,t}, C\right) + \mathcal{N}\left(0,\sigma^2C^2\right)\right)$$

then plug in private gradients to any adaptive optimization methods

$$m^t \leftarrow \beta_1 m^t + (1-\beta_1)\tilde{g}^t,\ v^t \leftarrow \beta_2 v^t + (1-\beta_2)\left(\tilde{g}^t\right)^2$$

$$w^{t+1} \leftarrow w^t - \alpha\frac{m^t}{\sqrt{v^t}+\epsilon}$$

How to effectively adapt to the geometry of gradients under DP?

directly plug in private gradients to estimate the statistics ?

first privatize the gradients

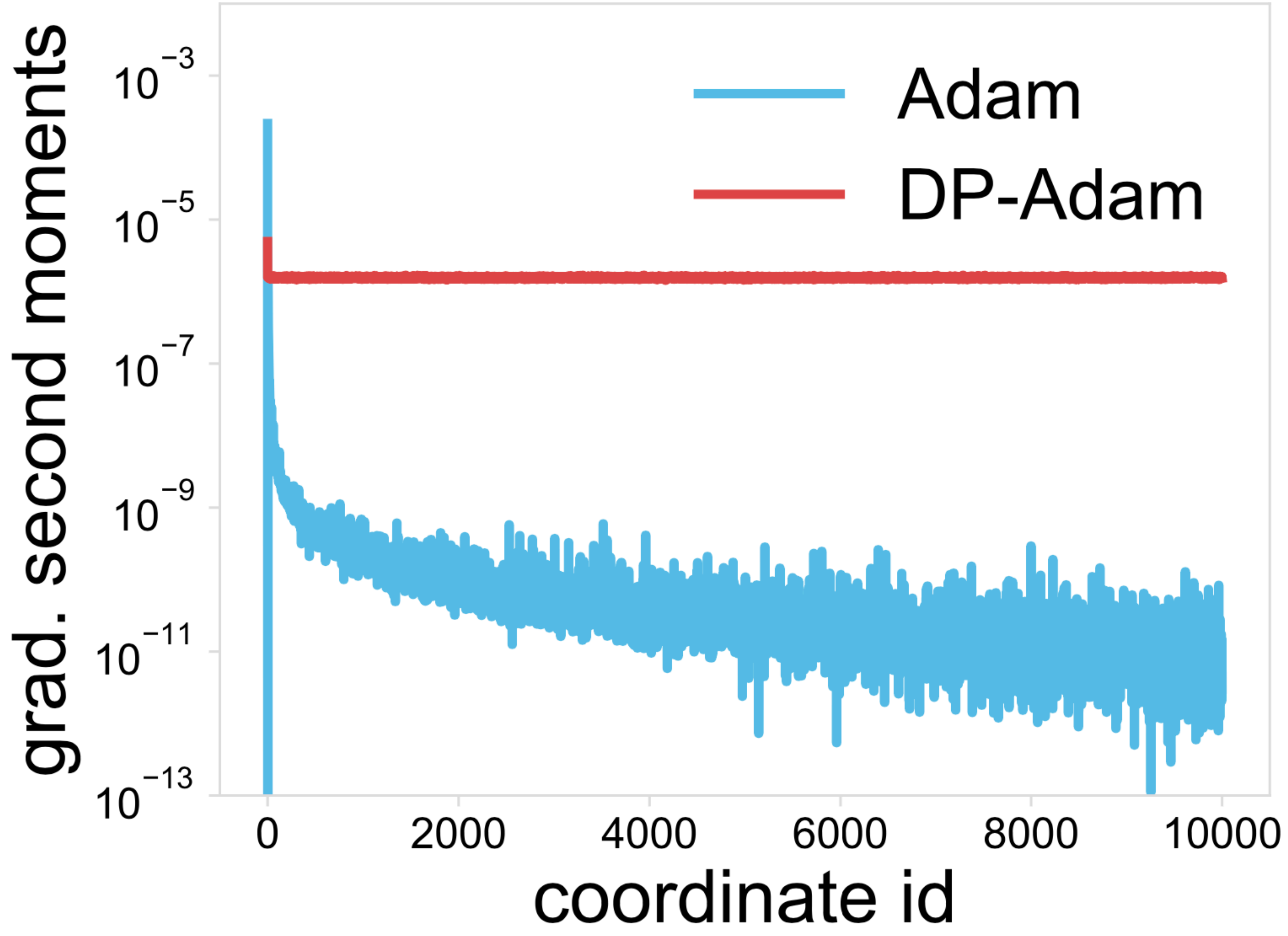
$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(g^{i,t}, C\right) + \mathcal{N}\left(0,\sigma^2 C^2\right)\right)$$

then plug in private gradients to any adaptive optimization methods

$$m^t \leftarrow \beta_1 m^t + (1-\beta_1)\tilde{g}^t,\ v^t \leftarrow \beta_2 v^t + (1-\beta_2)\left(\tilde{g}^t\right)^2$$

$$w^{t+1} \leftarrow w^t - \alpha\frac{m^t}{\sqrt{v^t}+\epsilon}$$

estimates can be very noisy!



AdaDPS: Private Adaptive Optimization with Side Information

AdaDPS: Private Adaptive Optimization with Side Information

With public data

Estimate gradient statistics on public data at each iteration

- obtained via 'opt-out' users or proxy data

AdaDPS: Private Adaptive Optimization with Side Information

With public data

Estimate gradient statistics on public data at each iteration

- obtained via ‘opt-out’ users or proxy data

Without public data

Non-sensitive common knowledge about the training data

- e.g., token frequencies in NLP

AdaDPS: Private Adaptive Optimization with Side Information

With public data

Estimate gradient statistics on public data at each iteration

- obtained via 'opt-out' users or proxy data

Without public data

Non-sensitive common knowledge about the training data

- e.g., token frequencies in NLP

$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(\frac{g^{i,t}}{A}, C\right) + \mathcal{N}\left(0, \sigma^2 C^2\right)\right), \; A \approx \sqrt{\mathbb{E}\left[g^2\right]} + \epsilon$$

A encodes how predictive each coordinate is

preconditioning *before* privatizing the gradients

AdaDPS: Private Adaptive Optimization with Side Information

$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(\frac{g^{i,t}}{A}, C\right) + \mathcal{N}\left(0, \sigma^2 C^2\right)\right), \; A \approx \sqrt{\mathbb{E}\left[g^2\right]} + \epsilon$$

A encodes how predictive each coordinate is

preconditioning *before* privatizing the gradients

AdaDPS: Private Adaptive Optimization with Side Information

Convergence:

(informal) rate: $O\left(\frac{1}{\sqrt{T}}\right) + O\left(\frac{1}{\sqrt{T}}\mathbb{E}\left[\|\mathcal{N}\|_A^2\right]\right)$

reduced DP noise when the gradients are sparse

$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(\frac{g^{i,t}}{A}, C\right) + \mathcal{N}\left(0, \sigma^2 C^2\right)\right),\ A \approx \sqrt{\mathbb{E}\left[g^2\right]} + \epsilon$$

A encodes how predictive each coordinate is

preconditioning *before* privatizing the gradients

Empirical Results

Empirical Results

centralized training

sample-level DP

Empirical Results

centralized training

sample-level DP

federated learning

client-level DP

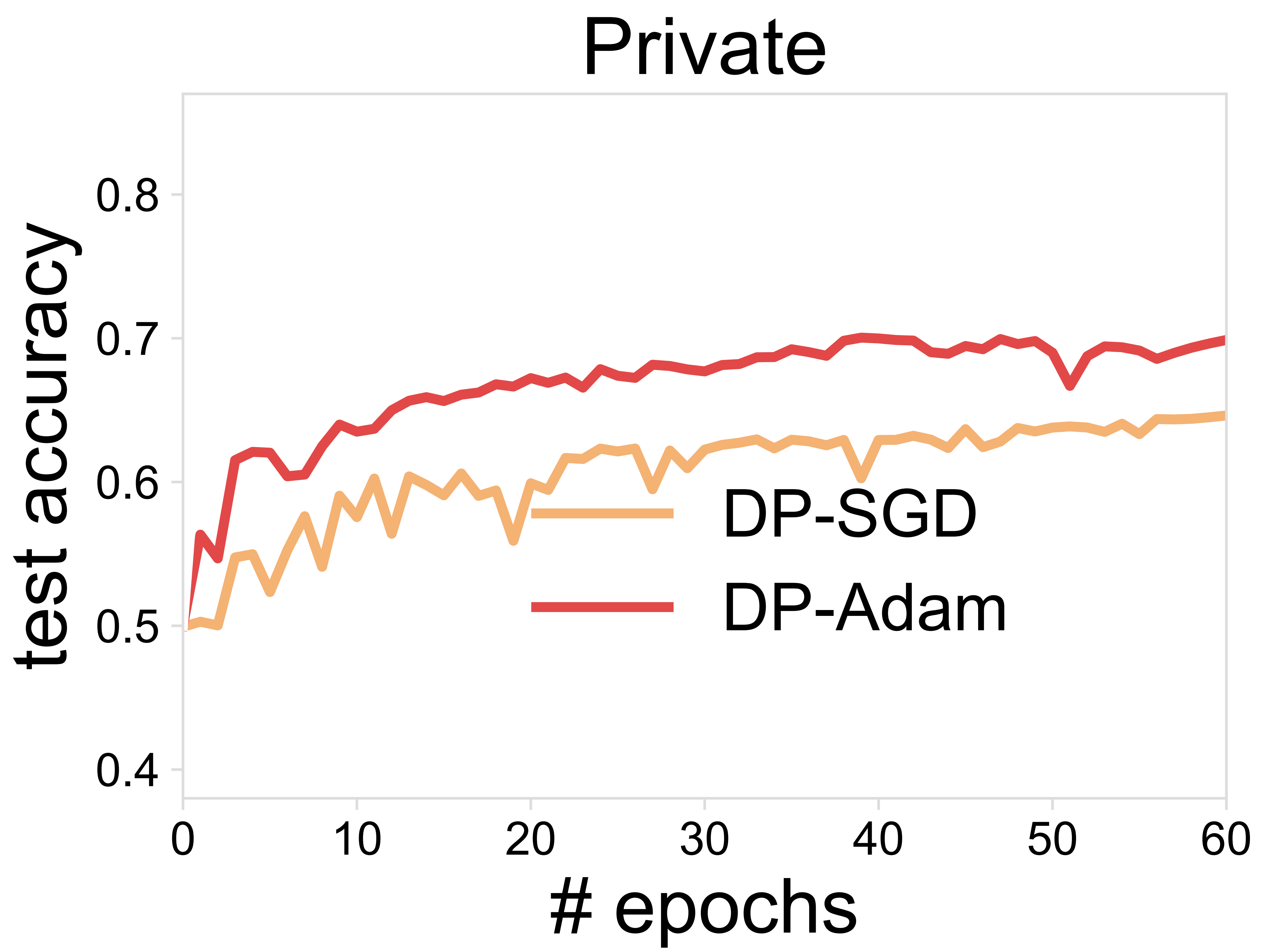
Empirical Results

centralized training

sample-level DP

federated learning

client-level DP



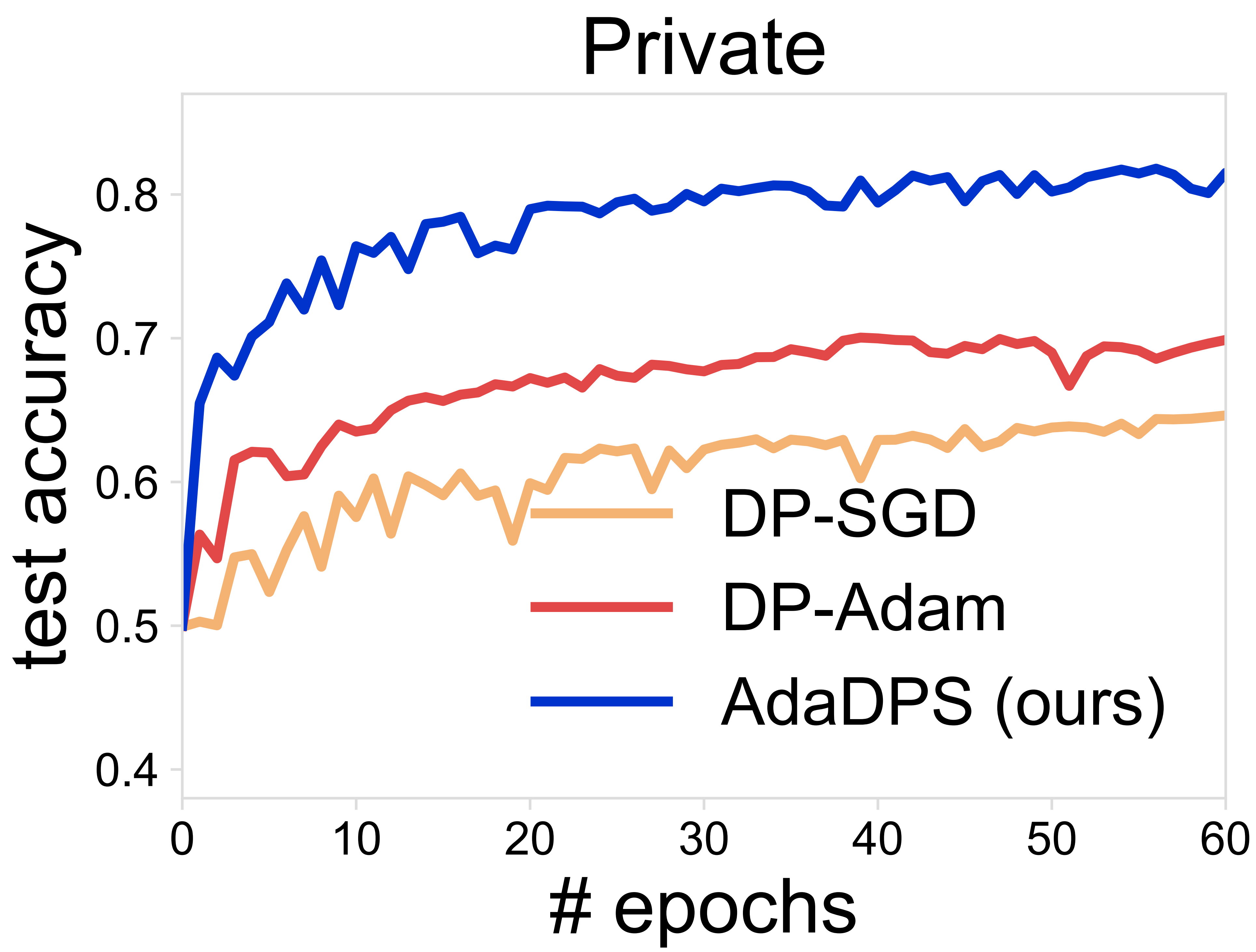
Empirical Results

centralized training

sample-level DP

federated learning

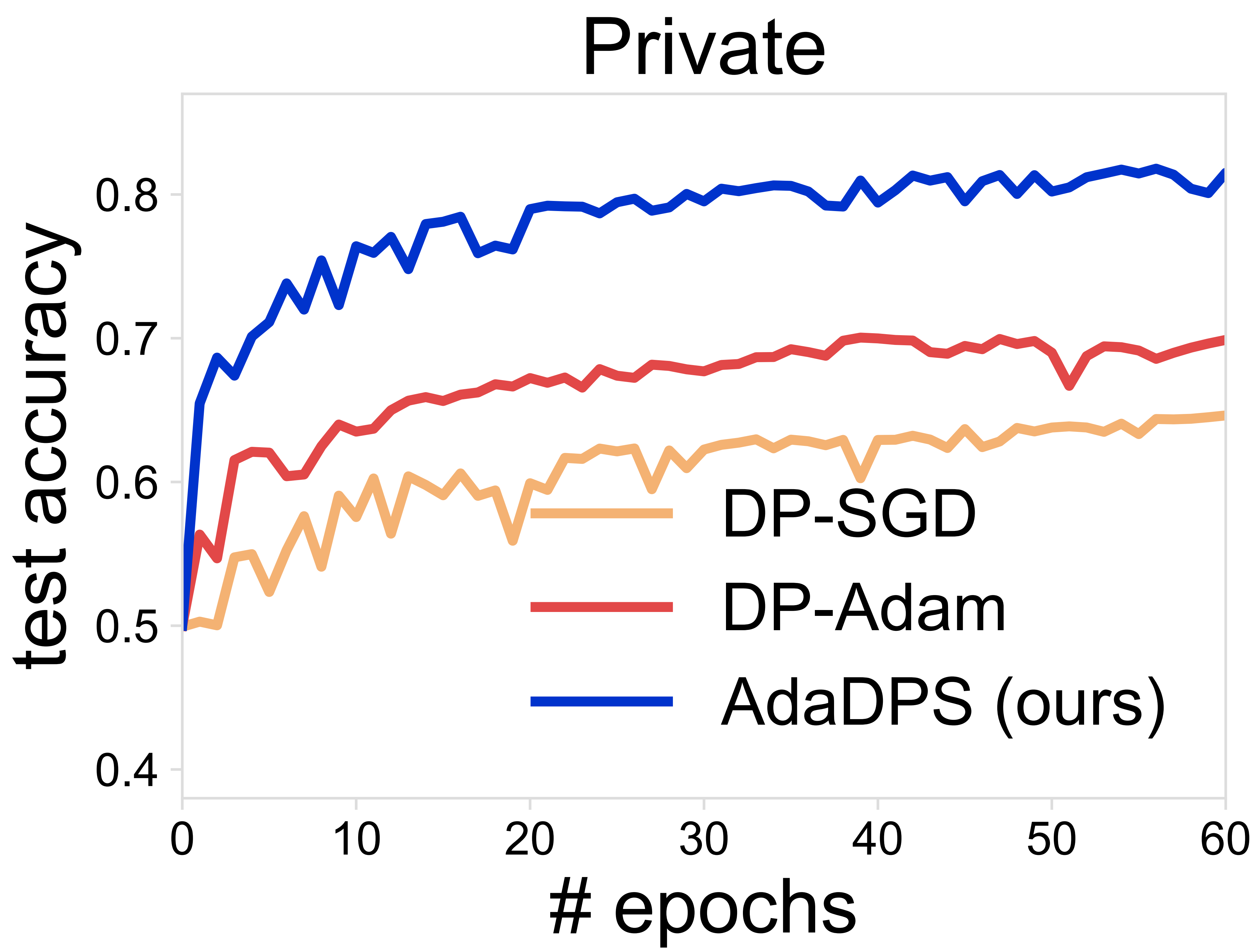
client-level DP



Empirical Results

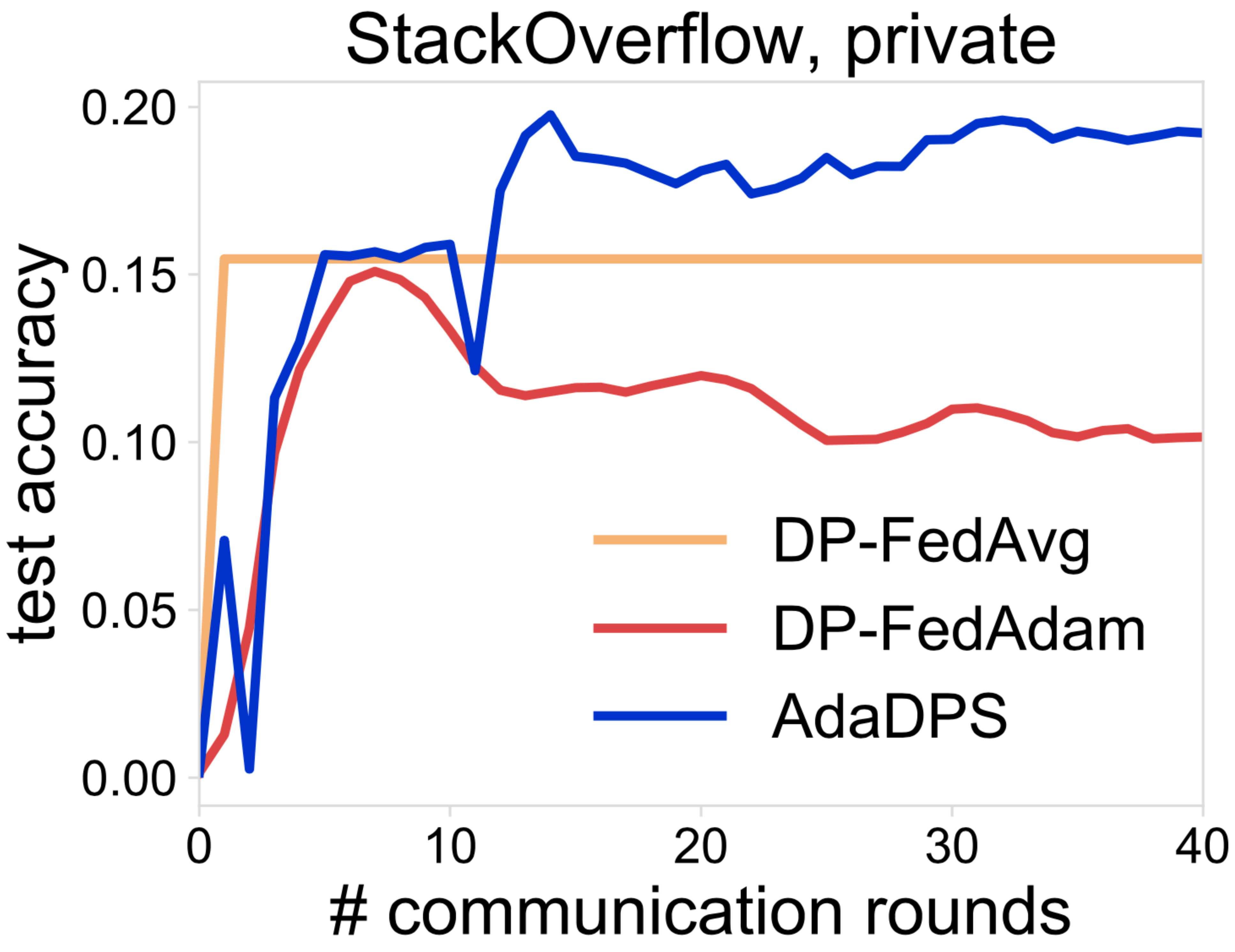
centralized training

sample-level DP



federated learning

client-level DP



Future Works

Future Works

- Exploring other approaches of reducing noise (e.g., with tree aggregation)
- Generalizing our approach without public data to arbitrary neural networks

Future Works

- Exploring other approaches of reducing noise (e.g., with tree aggregation)
- Generalizing our approach without public data to arbitrary neural networks

Full paper: *arxiv.org/abs/2202.05963*

Code: *github.com/litian96/AdaDPS*