Carnegie Mellon University Computer Science Department
DeepMind
Google Research
MACHINE LEARNING Carnegie Mellon University

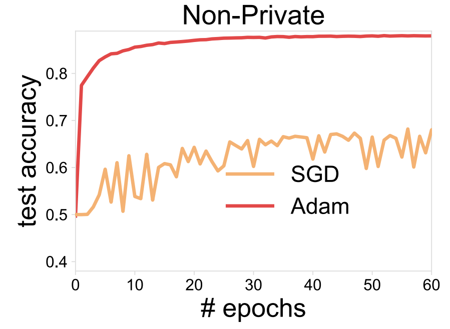
Private Adaptive Optimization with Side Information

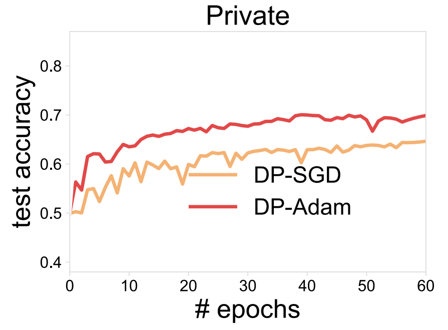
Tian Li (CMU), Manzil Zaheer (DeepMind), Sashank Reddi (Google Research), Virginia Smith (CMU)

Motivation

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

✘ Performance may degrade significantly when trained with differential privacy (DP)





A baseline: directly plug in private gradients to estimate the statistics?

For example (vanilla DP-Adam)

1. first privatize the gradients

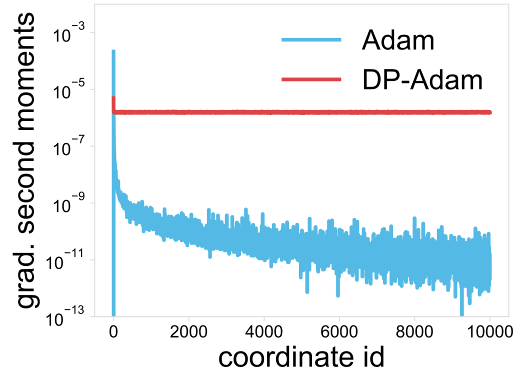
$$\tilde{g}^t \leftarrow \frac{1}{|B|}\left(\sum_{i\in B} \text{clip}\left(g^{i,t}, C\right) + \mathcal{N}\left(0, \sigma^2 C^2\right)\right)$$

2. then plug in private gradients to any adaptive optimization methods (e.g., Adam)

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

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

Estimates can be very noisy!



Insights

Use side Information to approximate the preconditioner

Estimate gradient statistics on small public data at each iteration

- obtained via 'opt-out' users or proxy data
- can be in-distribution or out-of-distribution

Non-sensitive common knowledge about the training data

- easy to obtain before training
- e.g., token frequencies in NLP

Useful for both private and non-private training

AdaDPS: Private Adaptive Optimization with Side Information

Option 1: With public data x_{pub}

1. get gradients on public data: $\hat{g}^t \leftarrow \frac{1}{|B|}\sum_{j\in B}\nabla f(x^j; w^t), x^j \in x_{\text{pub}}$
2. update preconditioner A^t with recurrence ϕ: $A^t \leftarrow \phi(A^{t-1}, \hat{g}^t)$

Option 2: Without public data

A^t estimated via heuristics (e.g., TF-IDF values or feature frequencies)

(optional) maintain a momentum buffer using $\tilde{g}^t$

Privatize preconditioned gradients (in the simplest form)

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

A encodes how predictive each coordinate is / estimates of gradient geometry

- preconditioning *before* privatizing the gradients (instead of the other order)
- can cover a range of adaptive methods

Convergence Analysis

1. With public data, convex case, RMSProp updates

$$\underbrace{\frac{G}{\sqrt{T}}\sum_{j=1}^{d}\mathbb{E}\left[A_j^T\right]}_{\text{standard RMSProp rate}} + \underbrace{\frac{\alpha}{\sqrt{T}}\max_{t\in[T]}\mathbb{E}\left[\|\mathcal{N}\|^2_{A^t}\right]}_{\text{DP noise (subsampled Gaussian mechanism)}}$$

reduced DP noise when the gradients are sparse

2. Without public data, fixed preconditioner A, convex case, RMSProp updates

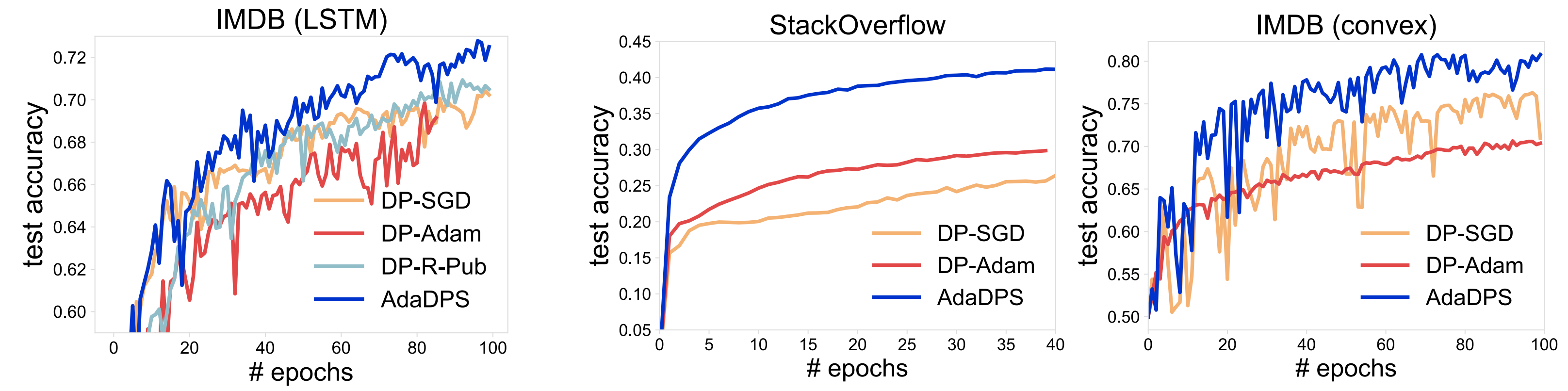
$$\frac{\alpha R+1}{\sqrt{T}}\sum_{j=1}^{d}A_j + \frac{\alpha}{\sqrt{T}}\mathbb{E}\left[\|\mathcal{N}\|^2_A\right], R := \max_{j,t}\frac{\mathbb{E}[(g_j^t)^2]}{A_j^2}$$

One practical choice: feature frequencies for generalized linear models: $A_j = \mathbb{E}[|x_j|] + \epsilon$

Evaluation

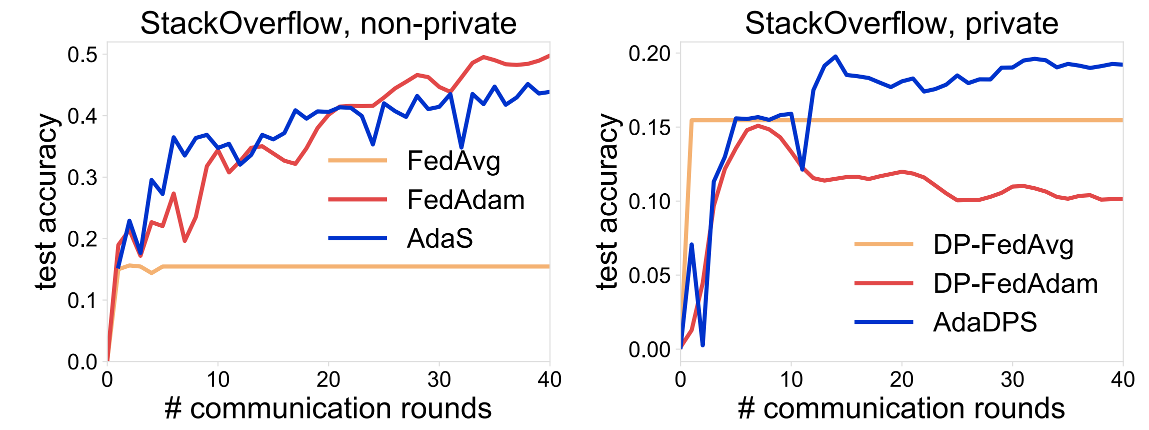
centralized training: sample-level DP

With public data | Without public data



In both convex and non-convex cases, AdaDPS outperforms the baselines significantly.

federated learning: client-level DP



We apply AdaDPS on the client side at each local update in federated optimization. It improves over baselines of vanilla DP-FedAvg and DP-FedAdam.

See paper for all results on all datasets, comparisons with additional recent baselines, results of using OOD data as public data, the effect of public data size, etc.

Future Work

- Exploring other approaches of reducing noise (e.g., with tree aggregation) in the context of adaptive optimization
- Generalizing our approach without public data to arbitrary neural network models

Code: https://github.com/litian96/AdaDPS
ArXiv: https://arxiv.org/abs/2202.05963

ICML | 2022