Efficient Adaptive Federated Optimization

Su Hyeong Lee [1] Sidharth Sharma [2] Manzil Zaheer [3] Tian Li [1]

[1]University of Chicago [2]Columbia University [3]Work done while at Google DeepMind

Overview

Goal: Adaptivity has been shown to accelerate convergence, and be critical to training transformer-based models such as LLMs. However, adaptive optimization imposes additional constraints on client memory and communication during distributed training. Can we develop a strategy to overcome these bottlenecks in federated learning?

Contributions:

- ✓ Develop a class of efficient jointly adaptive distributed training algorithms (FedAda^2/FedAda^2++) to mitigate the restrictions above while retaining full benefits of adaptivity
- ✓ Ensure that FedAda^2-class algorithms maintain an identical communication complexity as the vanilla FedAvg algorithm
- ✓ Provide robust convergence guarantees for the general, non-convex setting, achieving the same best known convergence rate as prior federated adaptive optimizers despite deploying joint adaptivity

Why is Adaptivity Desirable?

Motivating example: Training-time gradient distribution

Definition: Learning algorithm $\mathcal{A}$ is *deeply remorseful* if it incurs infinite regret in expectation. If $\mathcal{A}$ is guaranteed to instantly incur such regret due to sampling even a single client with a heavy-tailed gradient distribution, then $\mathcal{A}$ is *resentful* of heavy-tailed noise.

Theorem 1

For μ-strongly convex online global objectives, FedAvg becomes a deeply remorseful algorithm and is resentful of heavy-tailed noise.

- **Corollary 1:** Introducing client-side adaptivity via AdaGrad for the setting in Theorem 1 produces a non-remorseful and a non-resentful algorithm! Analogous result holds for joint adaptivity.
- **Corollary 2:** Even a single client with heavy-tailed gradient noise is able to instantaneously propagate their volatility to the global model, severely destabilizing distributed learning in expectation.

Moral of story: The advantage of federated learning is the large supply of clients, which enable the trainer to draw from an abundant stream of computational power. However, the downside is that the global model may become strongly impacted by the various gradient distributions induced by local data shards, which must be dealt with carefully to ensure stable training (e.g. using adaptive optimizers to mitigate regret).

FedAda^2: Efficient Joint Adaptivity

FedAda^2: Efficient Adaptive Federated Optimization

for $t = 1, \ldots, T$ **do**
  Sample participating clients $\mathcal{S}^t \subset [N]$
  for each client $i \in \mathcal{S}^t$ (in parallel) **do**
   (Main Idea 1:) Zero Precond. Initialization
   for $k = 1, \ldots, K$ **do**
    Draw $g^t_{i,k} \sim \mathcal{D}(x^t_{i,k-1})$, let $m_k \leftarrow MOM(g^t_{i,k})$
    (Main Idea 2:) Any Efficient Optimizer
   end for
   $\Delta^t_i = x^t_{i,K} - x_{t-1}$
  end for
  Server Update
end for

SM3-ADAGRAD VARIANT:

$$m_k \leftarrow g^t_{i,k}, \quad \mu_k(b) \leftarrow 0 \quad \text{for} \quad \forall b \in \{1, \ldots, q\},$$

$$\text{Loop } j: \begin{cases} v_k(j) \leftarrow \min_{b: S_b \ni j} \mu_{k-1}(b) + \left(g^t_{i,k}(j)\right)^2 \\ \mu_k(b) \leftarrow \max\{\mu_k(b), v_k(j)\}, \quad \forall b : S_b \ni j \end{cases}$$

Non-convex Convergence Analysis

Theorem 2

Under some assumptions, FedAda^2-class algorithms deterministically satisfy

$$\min_{t \in [T]} \|\nabla f(x_{t-1})\|^2 \leq \frac{\Psi_1 + \Psi_2 + \Psi_3 + \Psi_4 + \Psi_5}{\Psi_6}$$

where asymptotically,

$$\Psi_1 = \Theta(1),\ \Psi_2 = \eta^2\eta_\ell^2 T,\ \Psi_3 = \eta\eta_\ell^2 T,\ \Psi_4 = \eta\eta_\ell \log(1 + T\eta_\ell^2)$$

and

$$\Psi_5 = \begin{cases} \eta^3\eta_\ell^3 T \text{ if } \mathcal{O}(\eta_\ell) \leq \mathcal{O}(1) \\ \eta^3\eta_\ell T \text{ if } \Theta(\eta_\ell) > \Omega(1) \end{cases}, \quad \Psi_6 = \begin{cases} \eta\eta_\ell T \text{ if } \mathcal{O}(T\eta_\ell^2) \leq \mathcal{O}(1) \\ \eta\sqrt{T} \text{ if } \Theta(T\eta_\ell^2) > \Omega(1) \end{cases}.$$

Theorem 3 (Generalization)

Given client $i \in [N]$, strategy $l \in [Op]$, global timestep r, and local timestep p, assume optimizer strategies satisfy update rule

$$x^{r,l}_{i,p} = x^{r,l}_{i,p-1} - \eta_\ell \sum_{\ell=1}^{p} \frac{a^{r,l}_{i,\ell} g^{r,l}_{i,\ell}}{\vartheta^{r,l}_{i,\ell}(g^{r,l}_{i,1}, \ldots, g^{r,l}_{i,\ell})}$$

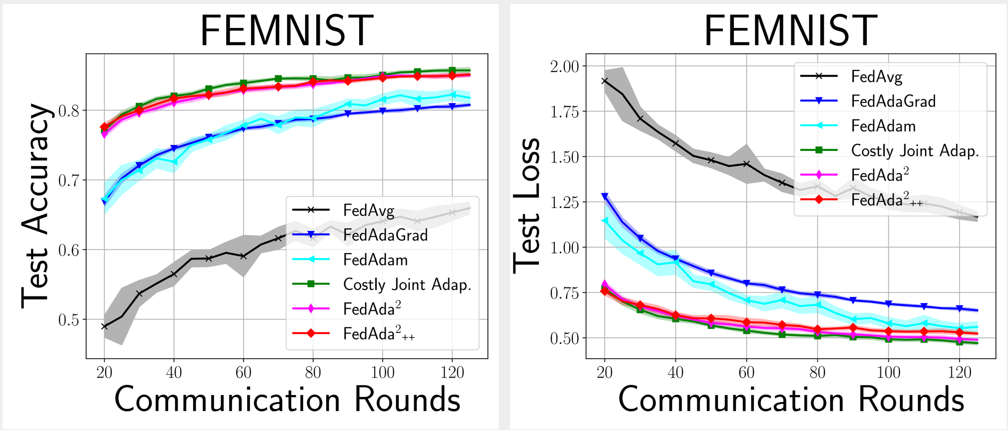
where

$$0 < m_l \leq \vartheta^{r,l}_{i,\ell}(g^{r,l}_{i,1}, \ldots, g^{r,l}_{i,\ell}) \leq M_l \quad \text{and} \quad 0 < a_l \leq a^{r,l}_{i,\ell} \leq A_l$$

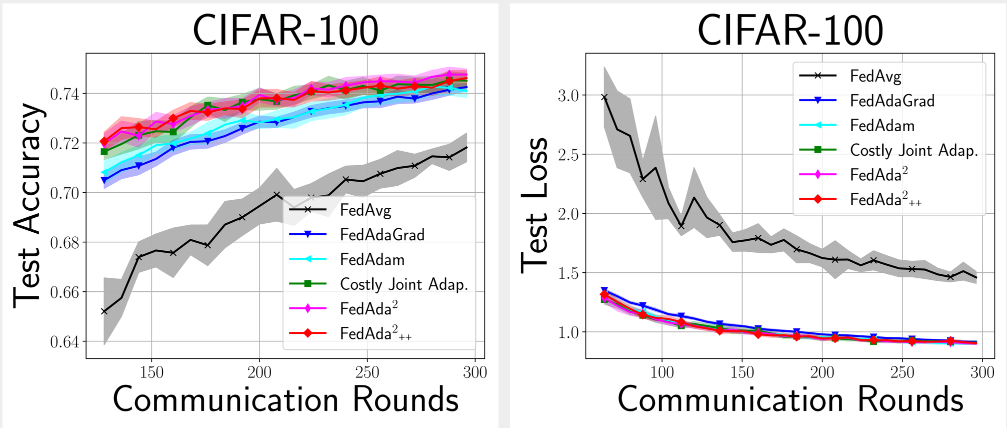
for all possible values of i, ℓ, r, l. If $1 \leq K(O^i_l) \leq K$ and $0 < \Xi^- < w(O^i_l) < \Xi^+$, then the bound in **Theorem 2** holds.

Experiments

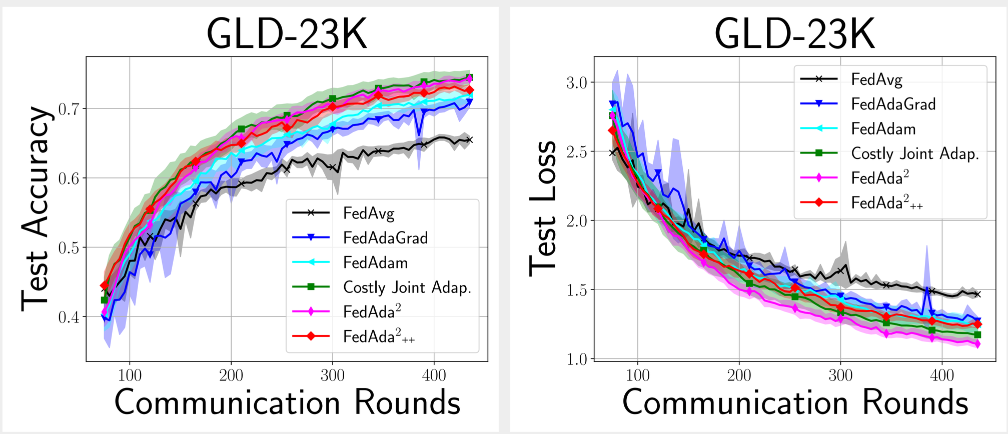
Joint Adaptivity > Server-only Adaptivity > FedAvg



Costly Joint Adaptivity ≈ Joint Adaptivity w/o Preconditioner Transmission (FedAda^2)



Further Efficient FedAda^2++ ≈ Costly Joint Adaptivity



Performance Comparison

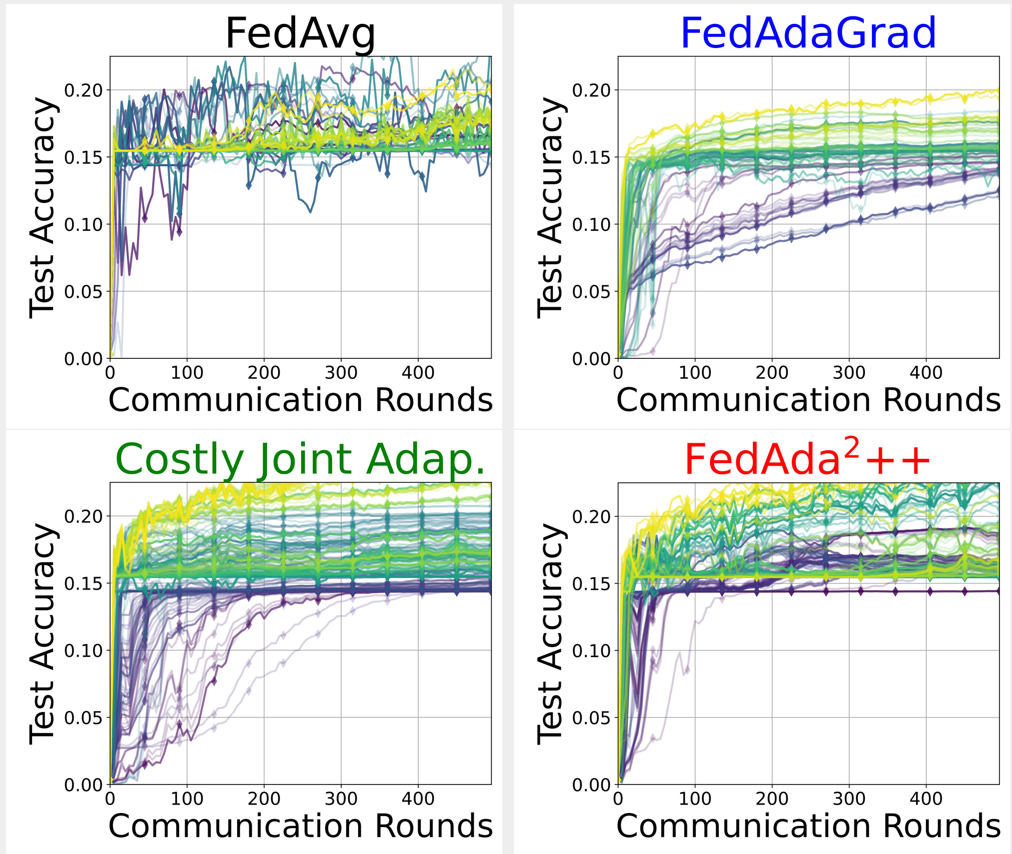
Analytical table summary:

Method	Joint Adaptivity?	Communication	Computation (# grad. calls)	Memory (client)
FedAvg	N	2d	1	d
FedAdaGrad	N	2d	1	d
FedAdam	N	2d	1	d
MIME	N	5d	3	4d
MIMELite	N	4d	2	3d
Costly Joint Adap.	Y	3d	1	2d
FedAda^2	Y	2d	1	2d
FedAda^2++	Y	2d	1	$\sim d$

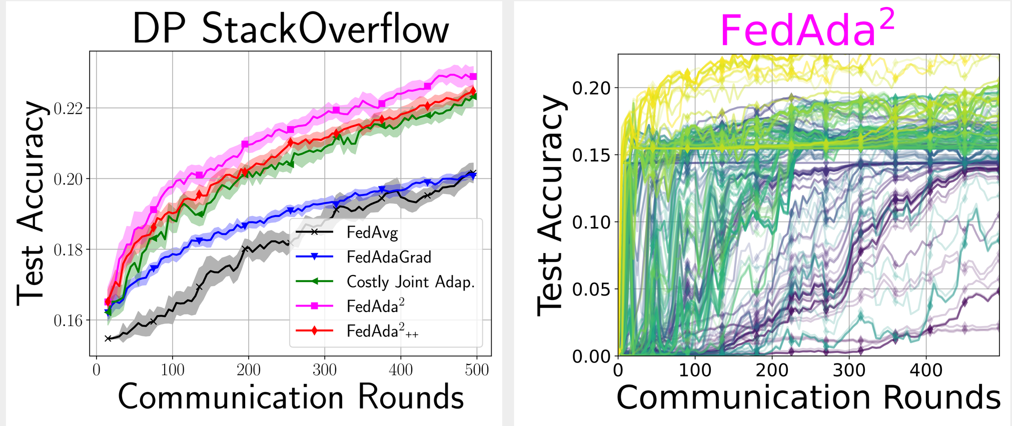
Comparison of algorithms (AdaGrad as the adaptive optimizer, d model dimension). For ViT, second-moment estimates in SM3-FedAda^2++ requires only 0.48% additional memory, with 99% reduction in extra client memory required for client preconditioning.

Further Experiments

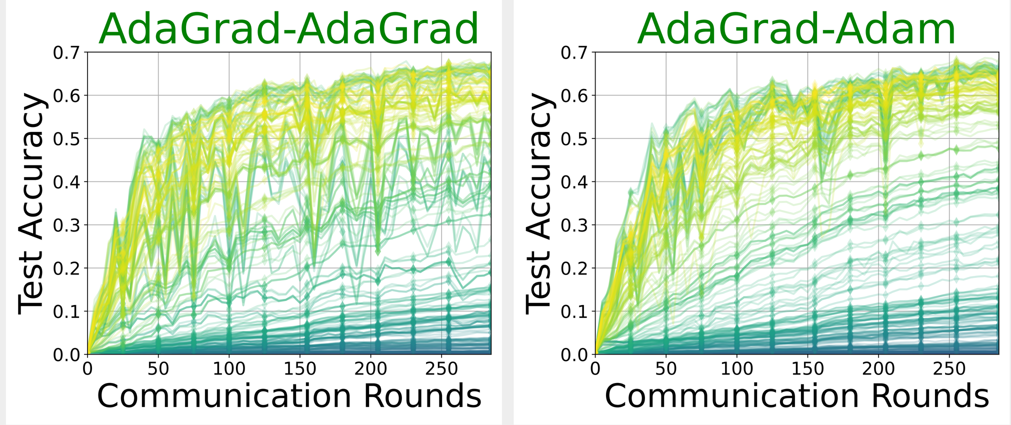
DP StackOverflow (Denoising Effects)



Performance of FedAda^2/FedAda^2++ in DP Setting



Robustness & Asymmetric Preconditioner Transmission



Future Work

1. Generalize full gradient convergence results to stochastic gradients
2. Elucidate the link between attention mechanisms and heavy-tailed gradient noise, and propose additional optimizers
3. Explore empirical performance of blended optimization, identifying settings in which mixing optimizer strategies (e.g. using client-side SGD & Adam in the same round) are advantageous for distributed learning