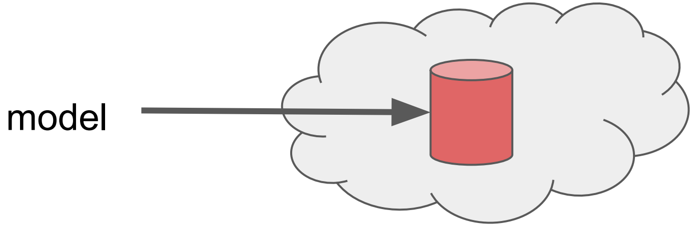
Breaking the centralized barrier in cross-device federated learning

Sai Praneeth Karimireddy, **Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, Ananda Theertha Suresh**

How to adapt normal deep learning algorithms to federated learning in a principled way?

Federated Learning: Setting

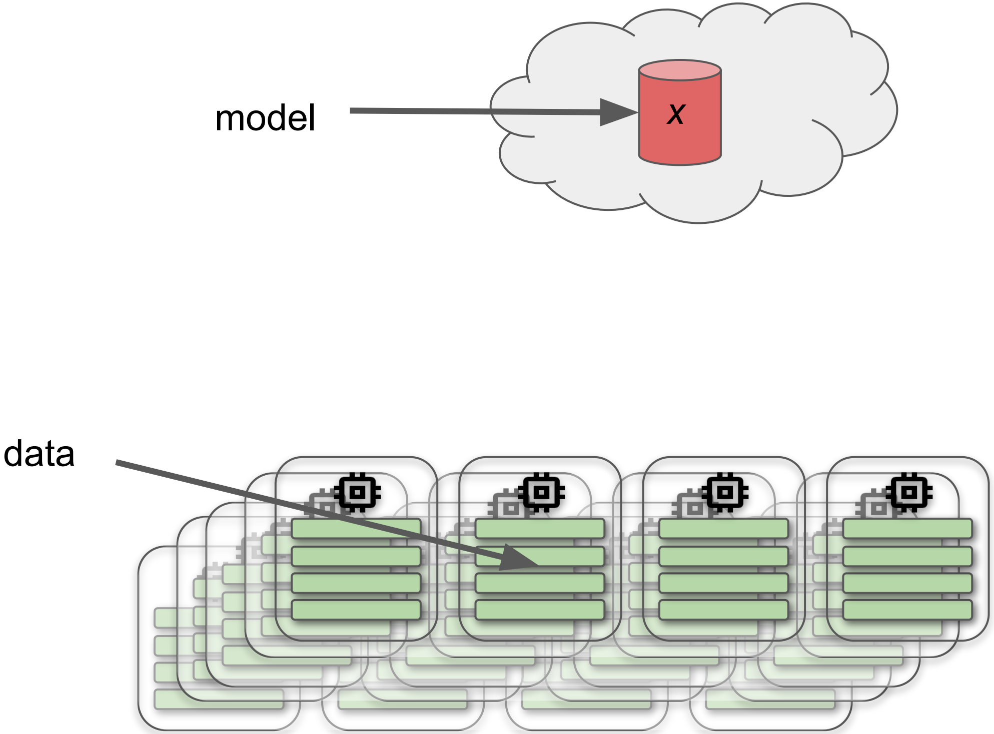
[McMahan et al. 2016]



Server
(e.g. Google)

Clients
(e.g. mobile phones)

Cross-device federated learning



- Huge (possibly infinite) number of clients => each client is seen *at most* once.
- High overhead per round
- Each client has a small amount of *heterogeneous* data

Cross silo vs. Cross device

Client data

$$\min_{x \in R^n} \frac{1}{N} \sum_{i=1}^{N} E_\zeta [f_i(x; \zeta)]$$

clients

- Cross-silo is closer to "finite-sum" optimization
- Can use variance reduction (SCAFFOLD, etc.)

Client data

$$\min_x \mathbb{E}_{i \sim \mathcal{D}} \left[f_i(x) := \frac{1}{n_i} \sum_{\nu=1}^{n_i} f_i(x; \xi_\nu) \right]$$

clients

- Cross-device is closer to "stochastic" optimization. Essentially, N is ∞.
- Use algorithms like SGD, momentum, Adam etc.

Cross device FL: In this talk

$$\min_x \mathbb{E}_{i \sim \mathcal{D}} \left[f_i(x) \right]$$

Model parameters

Expectation over (possibly infinite) clients

Full local gradients

And we only sample 1 client per round.

Also, we focus on momentum.

See the paper for full details.

Part 1: Algorithms
Mime Framework

Solving FL: Server only momentum

[Assume only 1 client per round]

$$x = x - \eta\left((1-\beta)\nabla f_i(x) + \beta m\right)$$

Update server parameters

$$m = (1-\beta)\nabla f_i(x) + \beta m$$

Update server momentum

+ Convergence guaranteed

- Communicates every update

Solving FL: FedAvg with momentum

[McMahan et al. 2016,
Hsu et al. 2019,
Reddi et al. 2020]

- Starting from x, run K local updates

$$\underbrace{y_i = y_i - \eta \nabla f_i(y_i)}_{\text{Repeat K times}}$$

+ Communicates only every K updates

- Use $(x - y_i)$ as a pseudo-gradient.

$$x = x - ((1-\beta)(x - y_i) + \beta m)$$

Update server parameters

$$m = (1-\beta)(x - y_i) + \beta m$$

\- bad convergence due to client drift because each client overfits to itself

[SCAFFOLD, Karimireddy et al. 2019]

But momentum helps!

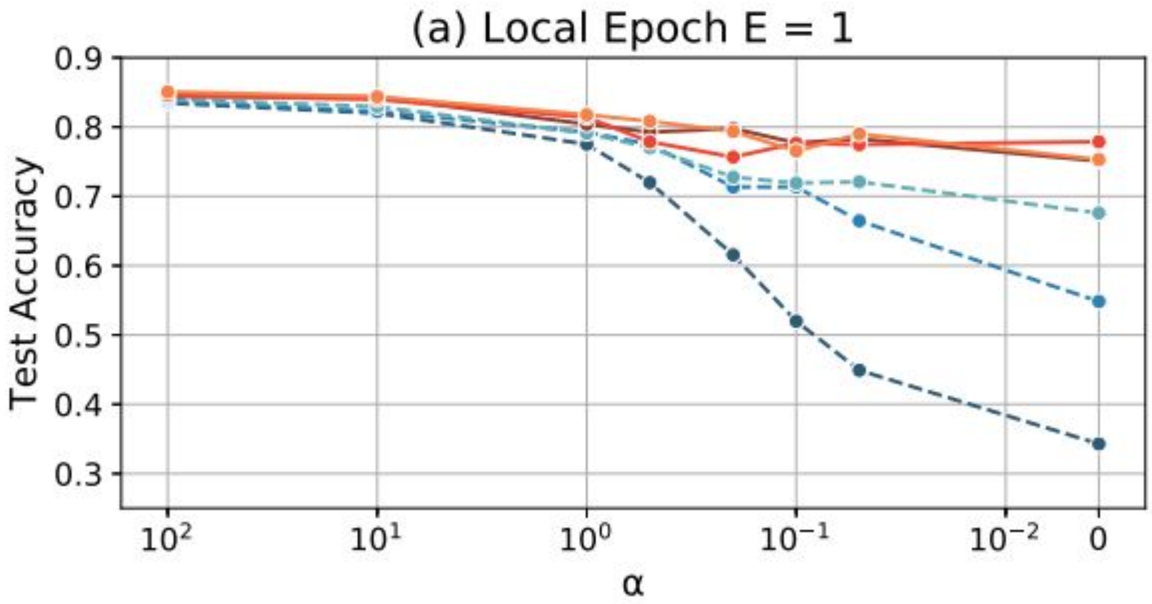


Figure from [Hsu et al. 2019]

FedAvg with momentum (in red) outperforms FedAvg SGD (in blue).

α quantifies non-iidness.

Solving FL: Mime with momentum

- Apply server momentum locally in the clients

$$\underbrace{y_i = y_i - \eta\left((1-\beta)\nabla f_i(y_i) \boxed{+\beta m}\right)}_{\text{Repeat K times}}$$

Fixed server momentum

- Momentum is computed globally (at server) and applied locally (at clients)

$$m = (1-\beta)\nabla f_i(x) + \beta m$$

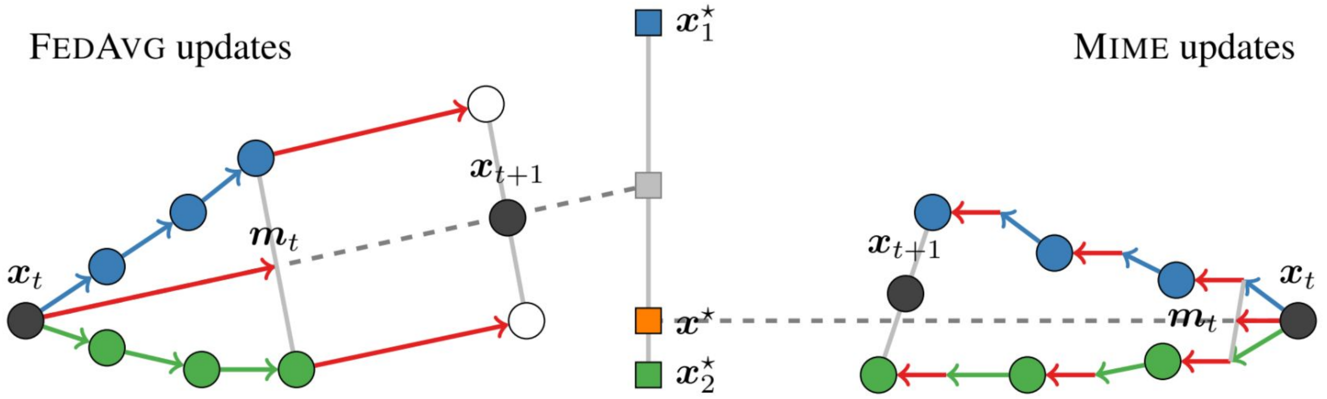
Update server momentum

- Observation 1: Momentum helps

- Observation 2: Momentum helps *more* for non-iid

  It must be “mixing” updates from different clients, preventing overfitting.

FedAvg momentum vs. Mime momentum



Mime Framework: Principles

1. Use “optimizer state” (momentum) every client update

2. Update “optimizer state” only at server, fixed during client updates.

Mime Framework: Base optimizer

Decompose a base centralized optimizer as $\mathcal{B} = (\mathcal{U}, \mathcal{V})$

1. Parameter update step:

$$\underbrace{x}_{\text{parameters}} = x - \eta\,\mathcal{U}(\underbrace{g}_{\text{gradient}}, \underbrace{s}_{\text{state}})$$

2. State update step:

$$s = \mathcal{V}(g, s)$$

Mime Framework: Base optimizer

For SGD with momentum, ***s* = *m***

1. Parameter update step:

$$\mathcal{U}(g, m) = (1 - \beta)g + \beta m$$

2. State update step:

$$\mathcal{V}(g, m) = (1 - \beta)g + \beta m$$

Momentum algorithm.

$$x = x - \eta\left((1 - \beta)g + \beta m\right)$$
$$m = (1 - \beta)g + \beta m$$

Mime Framework: Full algorithm

- Apply base optimizer locally at the clients

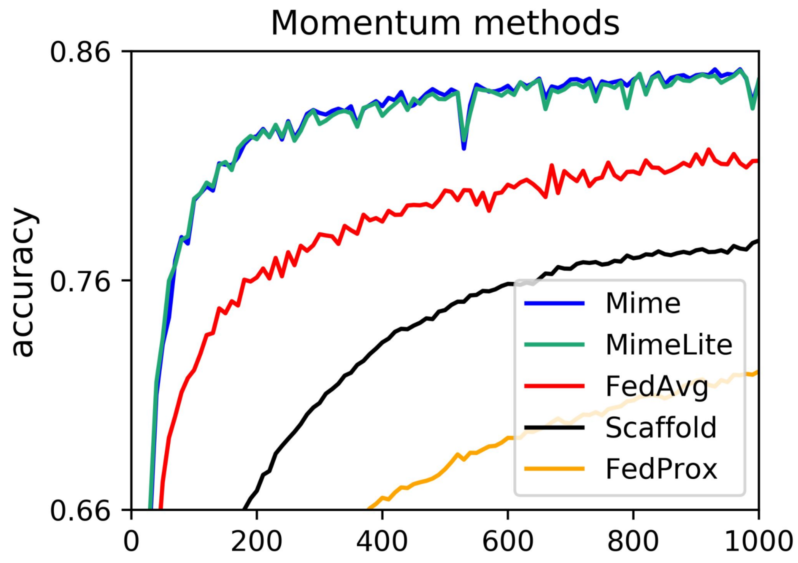
$$\underbrace{y_i = y_i - \eta\mathcal{U}(\nabla f_i(y_i)\boxed{, s)}}_{\text{Repeat K times}}$$

Fixed state

- State is updated only at server

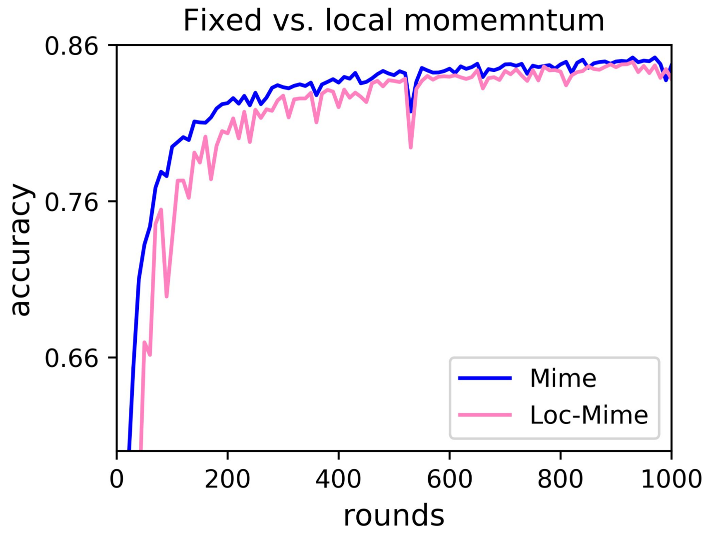
$$s = \mathcal{V}(\nabla f_i(x), s)$$

Experiment results: Comparison



- Extended MNIST 62
- MLP with 2 hidden layers
- 10 local epochs
- 20 of 3400 clients per round
- Momentum = [0, 0.9, 0.99]
- Tuned client lr, server lr =1.
- Regularization for FedProx tuned over [0.1, 0.5, 1] with 0.1 being the best. Using smaller values may improve performance but regularization 0 is same as FedAvg.

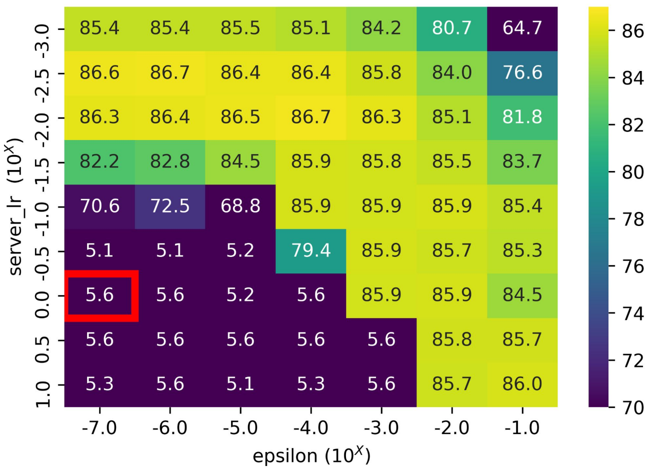
Experiment results: local momentum



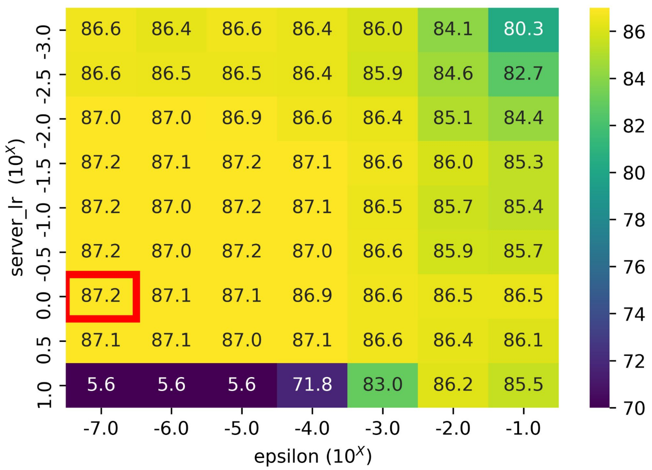
What happens if we also update momentum locally?

Experiment results: hyperparameters

Accuracy with FedAvg+Adam



Accuracy with Mime+Adam



Which hyperparameters work best?

MimeAdam works with same hyperparameters as centralized Adam.

Part 2: Theory

What is momentum doing?

Cross device FL: Formalism

Client data

$$\min_{x} \mathbb{E}_{i\sim\mathcal{D}} \left[f_i(x) := \frac{1}{n_i} \sum_{\nu=1}^{n_i} f_i(x; \xi_\nu) \right]$$

parameters clients loss function

- **G²** - Bounded Gradient dissimilarity:

$$\mathbb{E}_{i\sim\mathcal{D}} \|\nabla f_i(x) - \nabla f(x)\|^2 \leq G^2$$

- □- Bounded Hessian dissimilarity:

$$\|\nabla^2 f_i(x) - \nabla^2 f(x)\| \leq 2\delta$$

$$\text{Note, } \delta \leq L.$$

Server-only Momentum based variance reduction

Momentum based variance reduction (MVR) adds a small correction [Tran-Dinh et al. 2019, Cutkosky et al. 2020].

$$m^{t+1} = (1-\beta)\nabla f_{i_t}(x^t) + \beta m^t + \beta(\nabla f_{i_t}(x^t) - \nabla f_{i_t}(x^{t-1}))$$

Standard momentum

Correction

Theorem. $\min_{t\in[T]} \mathbb{E}\|\nabla f(x^t_{\mathrm{MVR}})\|^2 \leq \left(\frac{LG}{T}\right)^{2/3}$

Optimal server-only rate!

Convergence: Initial Attempt

$$\text{Theorem.}\quad \min_{t\in[T]} \mathbb{E}\|\nabla f(x^t_{\text{MimeMVR}})\|^2 \leq \left(\frac{LG}{T}\right)^{2/3}$$

Are we done?

Almost, but we can do better.

Prove **advantage** of local steps.

Local steps: Biased gradients

For random client i and gradient at server parameters,

$$\mathbb{E}_{i\sim\mathcal{D}}\left[\nabla f_i(x)\right] = \nabla f(x)$$

Unbiased gradient

But for gradient at client parameters,

$$\mathbb{E}_{i\sim\mathcal{D}}\left[\nabla f_i(y_i)\right] \neq \nabla f(y_i)$$

Causes client drift!

Biased gradient since y_i depends on i

Local steps: Hessian dissimilarity

□- BHD:

$$\|\nabla^2 f_i(x) - \nabla^2 f(x)\| \leq 2\delta$$

Amount of bias controlled by

$$\|\mathbb{E}_{i\sim\mathcal{D}}[\nabla f_i(y_i)] - \nabla f(y_i)\|$$

$$\approx \delta \mathbb{E}_i \|y_i - x\|$$

Diff. in Hessians

Distance moved in a round

- Num. steps should be inversely proportional to □

Convergence: With MVR

□- BHD:

$$\|\nabla^2 f_i(x) - \nabla^2 f(x)\| \leq 2\delta$$

Optimal Server-only momentum based variance reduction (MVR):

$$\text{Theorem.} \quad \min_{t\in[T]} \mathbb{E}\|\nabla f(x^t_{\text{MVR}})\|^2 \leq \left(\frac{LG}{T}\right)^{2/3}$$

MimeMVR is **faster** than any server-only method!

Mime with momentum based variance reduction (MimeMVR):

$$\text{Theorem.} \quad \min_{t\in[T]} \mathbb{E}\|\nabla f(x^t_{\text{MimeMVR}})\|^2 \leq \left(\frac{\delta G}{T}\right)^{2/3}$$

Takeaways

- Momentum injects global information and helps reduce client drift.
- Compute momentum globally at server, apply it during each client update.
- Usefulness of local steps depends on Hessian variance

Thank You.

See you at the poster!