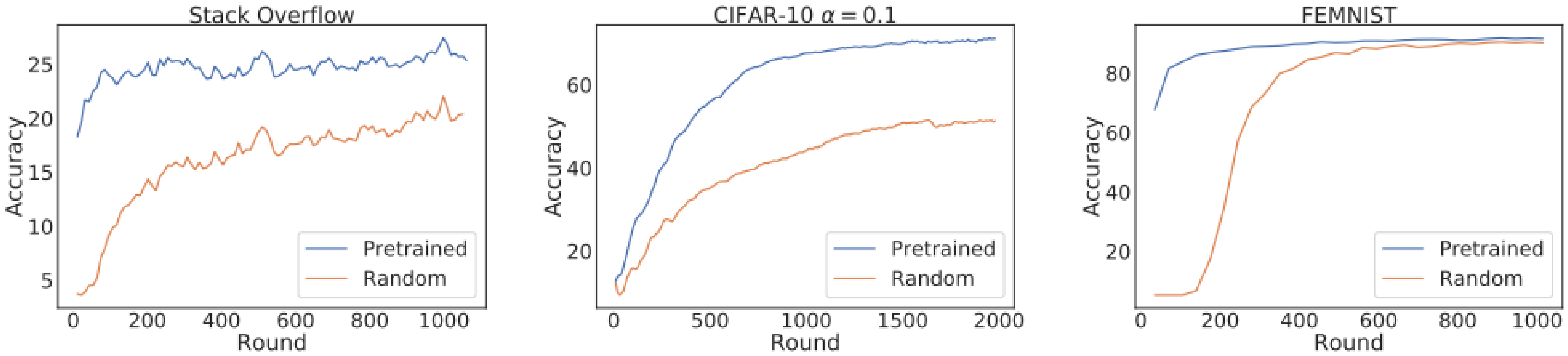
Where to Begin? Exploring the Impact of Pre-Training and Initialization in Federated Learning

John Nguyen Kshitiz Malik Maziar Sanjabi Michael Rabbat
Meta AI {ngjhn,kmalik2,maziars,mikerabbat}@fb.com

Contribution

How does the initialization (random, or pre-trained) impact the behavior of federated optimization methods?



Contribution

- Starting from a pre-trained solution can close the gap between training on IID and non-IID data (Section 5.2). Moreover, the simple SGD at the client outperforms more complex local-update methods in the pre-trained setting. (Section 5.1)

- Towards starting to explain this phenomenon, we observe that inter-device gradient/update diversity is higher for random initialized model at the beginning of training, and inter-device cosine similarity is higher when starting from a pre-trained model. (Section 5.4)

- Surprisingly, full-batch gradient descent without any local step can achieve competitive performance against other SOTA local-update methods in the pre-trained setting.

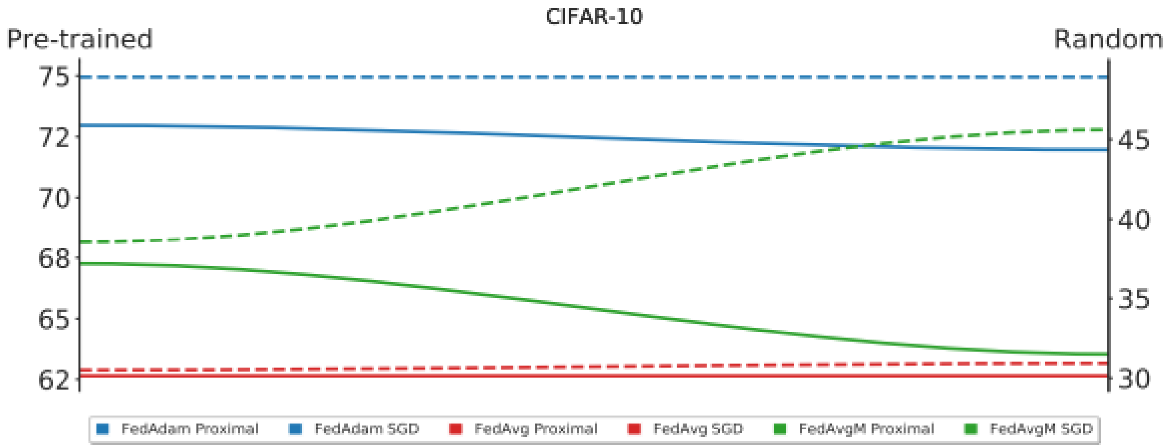
Related Work

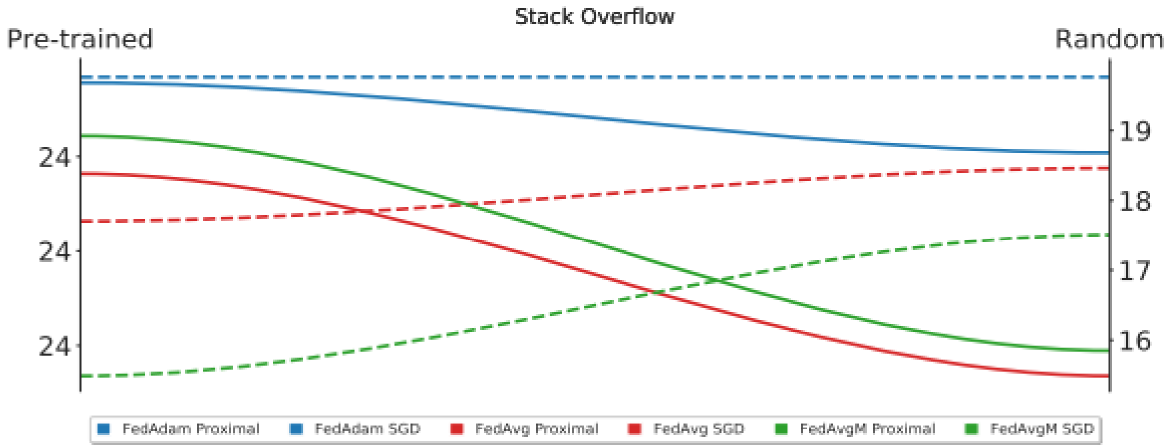
	NA	LS	GM	AS
FedAvg Nova	✓	✓	✗	✗
FedAvg Proximal	✗	✓	✗	✗
FedAvg SGD	✗	✓	✗	✗
FedAvg GD	✗	✗	✗	✗
FedAvgM Nova	✓	✓	✓	✗
FedAvgM Proximal	✗	✓	✓	✗
FedAvgM SGD	✗	✓	✓	✗
FedAvgM GD	✗	✗	✓	✗
FedAdam Nova	✓	✓	✓	✓
FedAdam Proximal	✗	✓	✓	✓
FedAdam SGD	✗	✓	✓	✓
FedAdam GD	✗	✗	✓	✓

Algorithm 1 FedOpt framework

```
Input: initial global model x^0, server and client step sizes η_s, η_c, local epochs E, rounds T
for each round t = 1, ..., T do
    Server sends x^{t-1} to all clients i ∈ S^t.
    for each client i ∈ S^t in parallel do
        Initialize local model y_i^0 ← x^{t-1}.
        Each client performs E epochs of local updates via y_i^{k+1} = ClientOpt(y_i^k, F_i, η_c). Let
        y_i^E denote the result after performing E epochs of local updates.
        After local training, client i sends Δ_i^t = x^{t-1} − y_i^E to the server.
    end for
    Server computes aggregate update Δ^t = (1/|S^t|) Σ_{i∈S^t} p_i Δ_i^t.
    Server updates global model x^t = ServerOpt(x^{t-1}, −Δ^t, η_s, t).
end for
```

1. Pre-training affects how federated optimization algorithms behave.

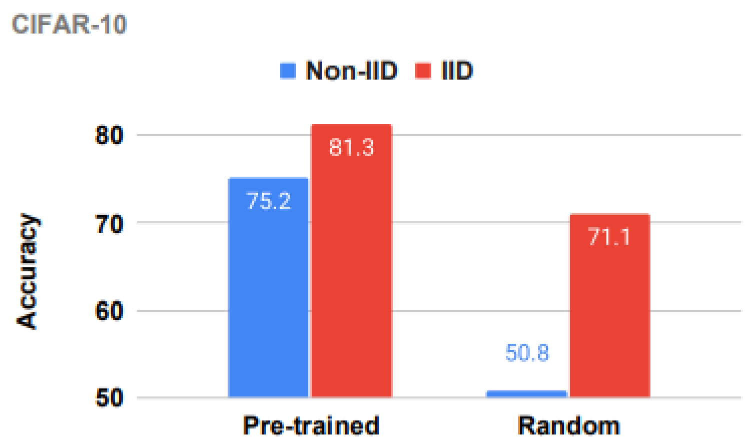


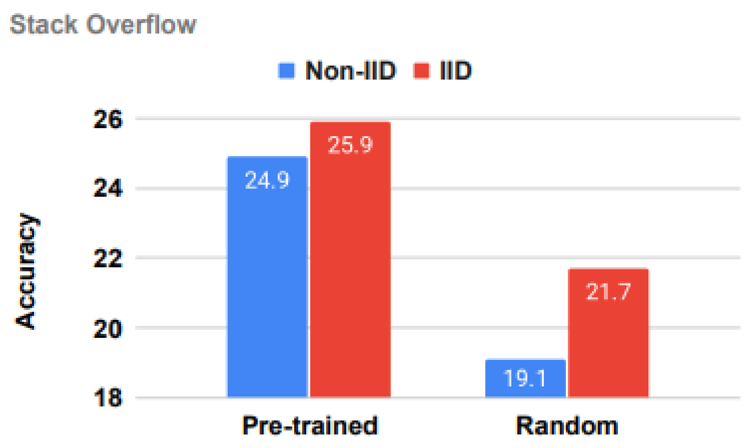


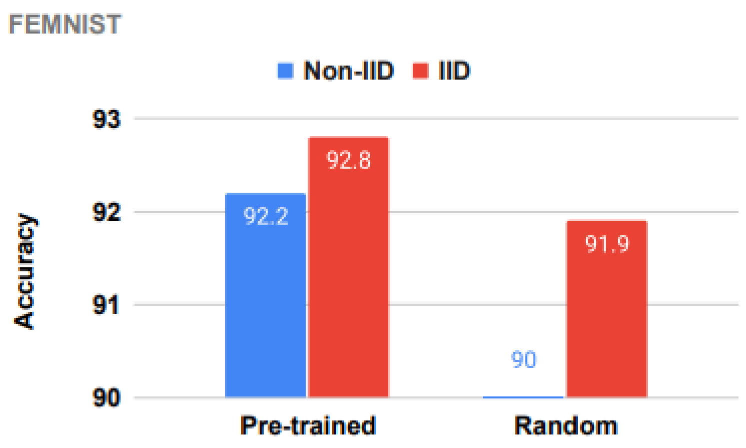
Contribution

- Starting from a pre-trained solution can close the gap between training on IID and non-IID data (Section 5.2). Moreover, the simple SGD at the client outperforms more complex local-update methods in the pre-trained setting. (Section 5.1)
- Towards starting to explain this phenomenon, we observe that inter-device gradient/update diversity is higher for random initialized model at the beginning of training, and inter-device cosine similarity is higher when starting from a pre-trained model. (Section 5.4)
- Surprisingly, full-batch gradient descent without any local step can achieve competitive performance against other SOTA local-update methods in the pre-trained setting.

2. Pre-training closes the accuracy gap between non-IID and IID







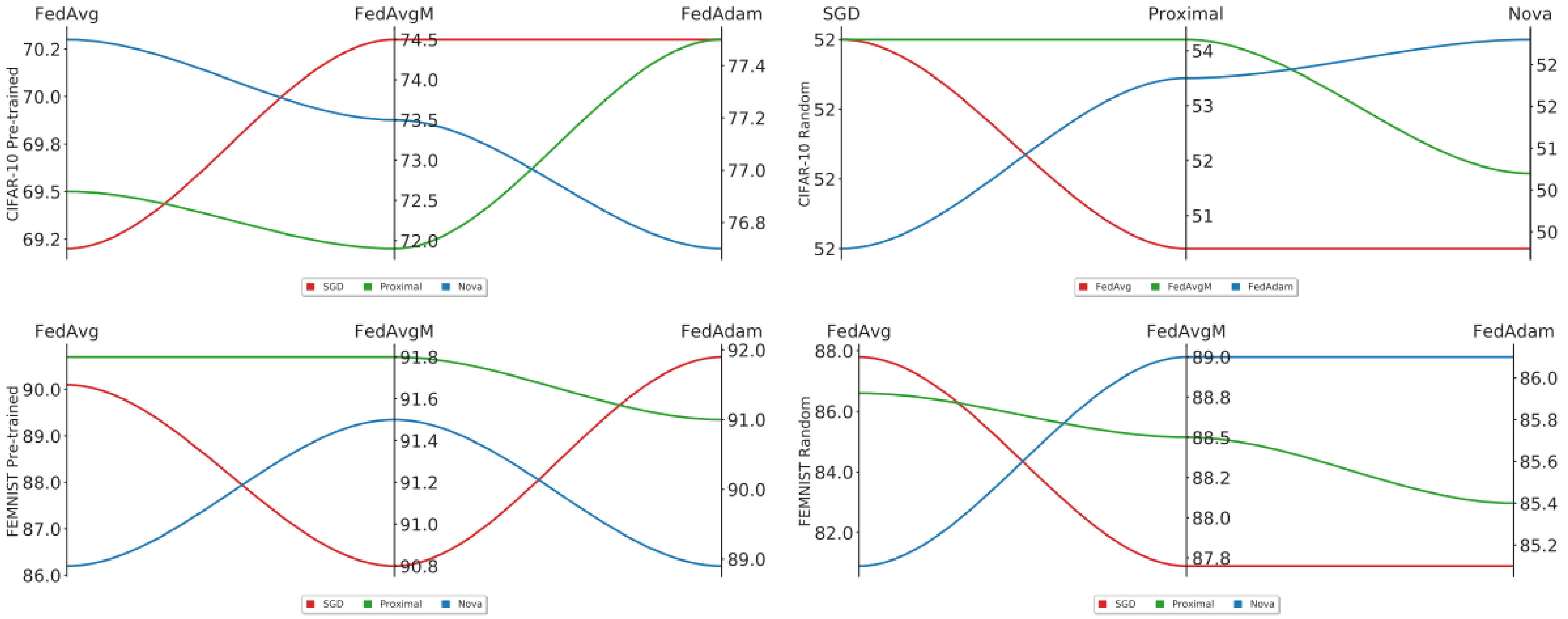
Contribution

- Starting from a pre-trained solution can close the gap between training on IID and non-IID data (Section 5.2). Moreover, the simple SGD at the client outperforms more complex local-update methods in the pre-trained setting. (Section 5.1)

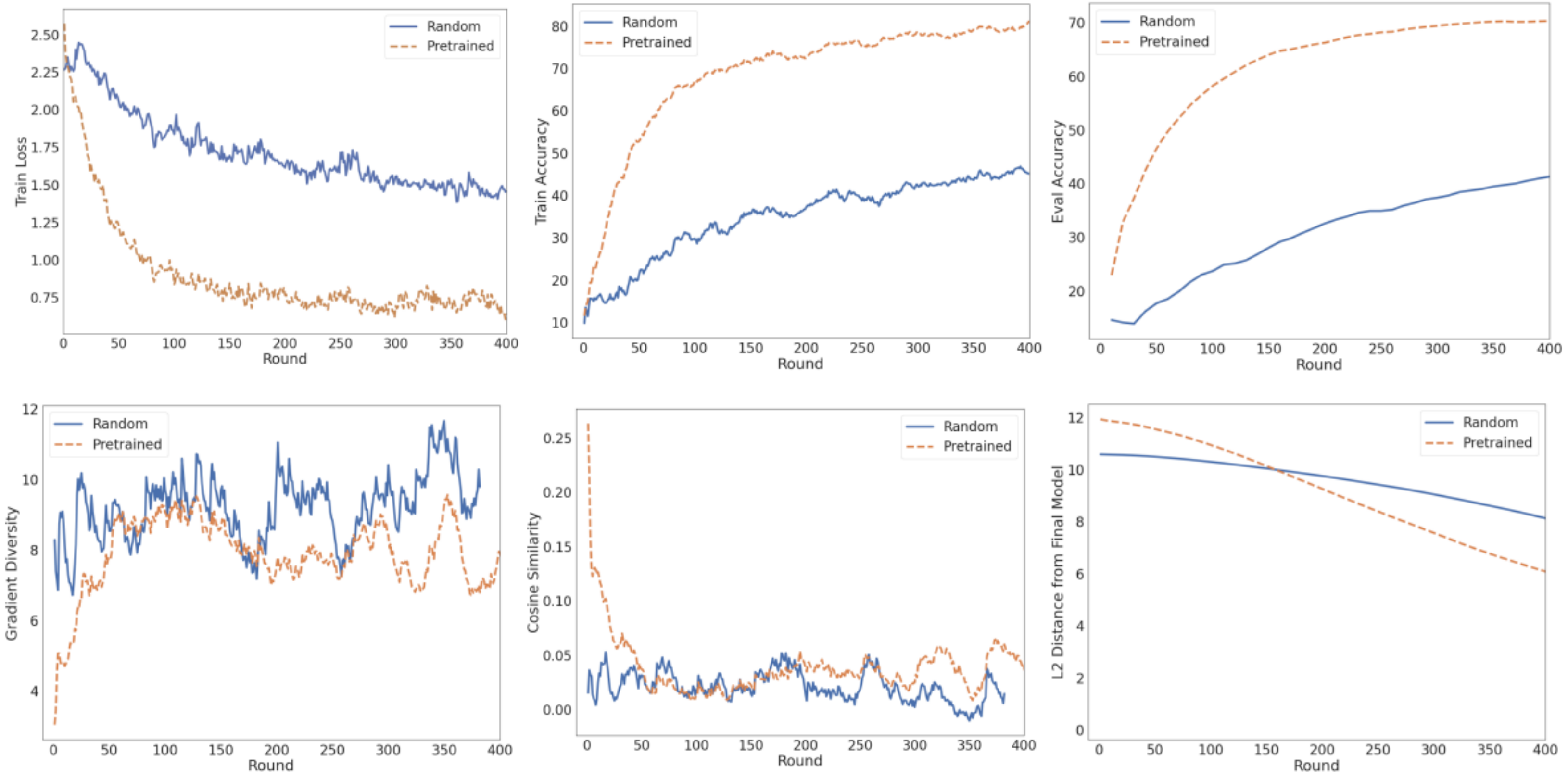
- Towards starting to explain this phenomenon, we observe that inter-device gradient/update diversity is higher for random initialized model at the beginning of training, and inter-device cosine similarity is higher when starting from a pre-trained model. (Section 5.4)

- Surprisingly, full-batch gradient descent without any local step can achieve competitive performance against other SOTA local-update methods in the pre-trained setting.

3. Pre-training reduces the impact of system heterogeneity.



4. Pre-training helps align client updates.



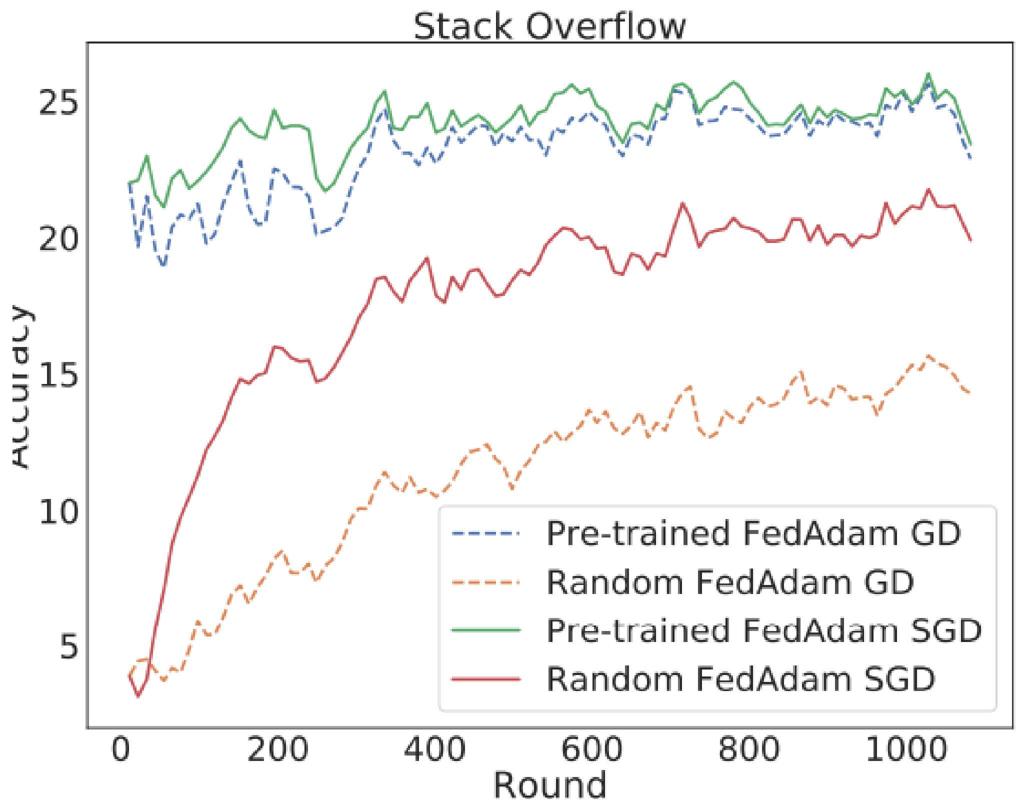
Contribution

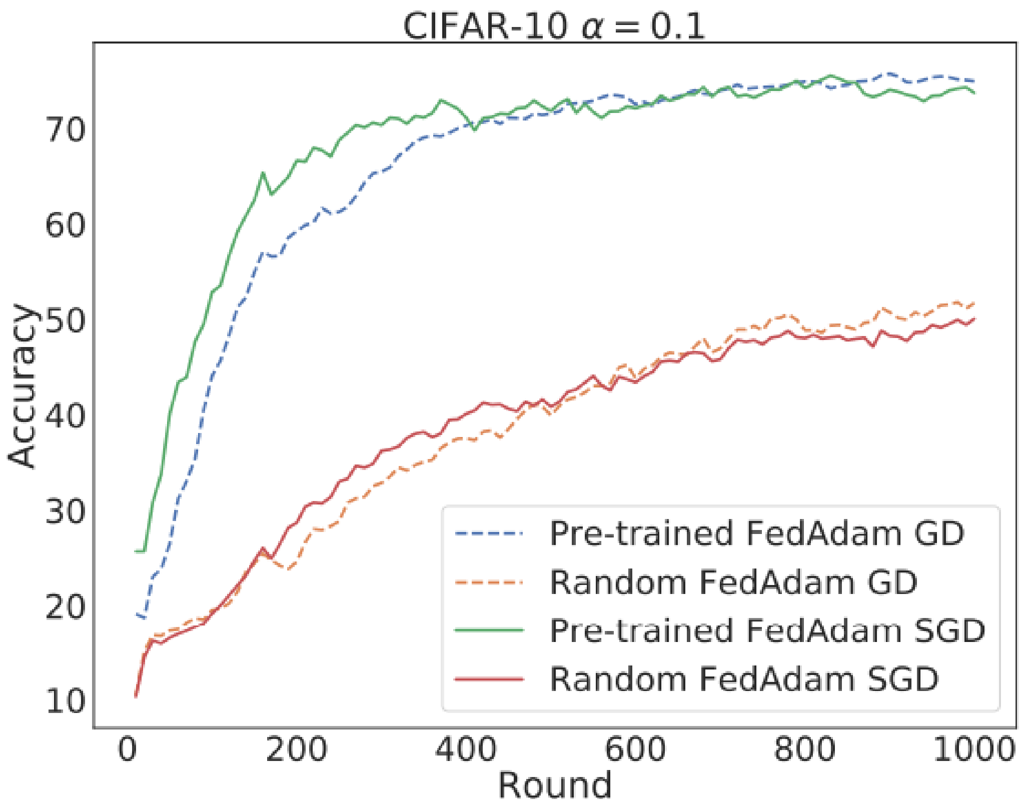
- Starting from a pre-trained solution can close the gap between training on IID and non-IID data (Section 5.2). Moreover, the simple SGD at the client outperforms more complex local-update methods in the pre-trained setting. (Section 5.1)

- Towards starting to explain this phenomenon, we observe that inter-device gradient/update diversity is higher for random initialized model at the beginning of training, and inter-device cosine similarity is higher when starting from a pre-trained model. (Section 5.4)

- Surprisingly, full-batch gradient descent without any local step can achieve competitive performance against other SOTA local-update methods in the pre-trained setting.

5. FEDADAM GD is as effective as FEDADAM SGD with pre-training





Contribution

- Starting from a pre-trained solution can close the gap between training on IID and non-IID data (Section 5.2). Moreover, the simple SGD at the client outperforms more complex local-update methods in the pre-trained setting. (Section 5.1)

- Towards starting to explain this phenomenon, we observe that inter-device gradient/update diversity is higher for random initialized model at the beginning of training, and inter-device cosine similarity is higher when starting from a pre-trained model. (Section 5.4)

- Surprisingly, full-batch gradient descent without any local step can achieve competitive performance against other SOTA local-update methods in the pre-trained setting.

Recommendations

- When evaluate FL algorithms, researchers should experiment with both pre-trained (if available) and random weights as they have different behaviors.
- When deploying FL to production environment, researchers should use adaptive server optimizers such as FedAdam and SGD at client. This setup works well and should be used a baseline before trying out more complex methods.
- Heterogeneity is not as a big of a problem when there is public data to pre-trained a model. We encourage researchers to pay attention other more complex tasks when there is no public data such as recommendation systems or semi-supervised learning.

Conclusion

- We find that pre-training on public data can recover most of the accuracy drop from heterogeneity

- We show that client updates starting from pre-trained weights have higher cosine similarity, which explains why initialized with pre-trained weights can speed up convergence and achieve high accuracy even in heterogeneous settings.