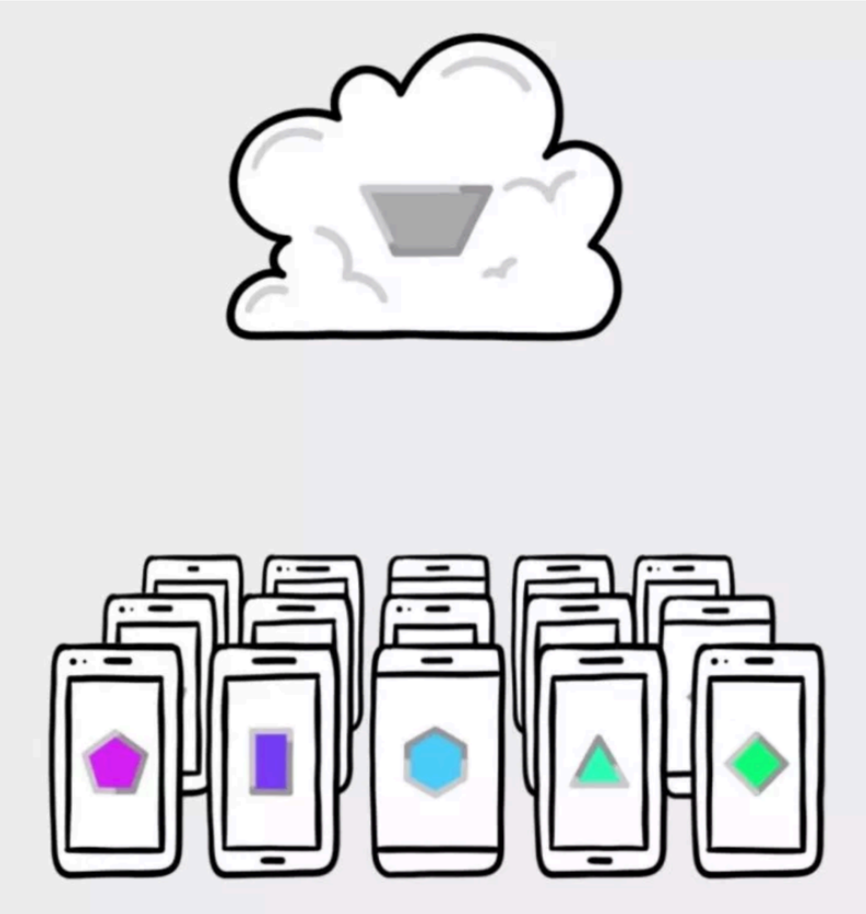
Federated Learning

Shusen Wang

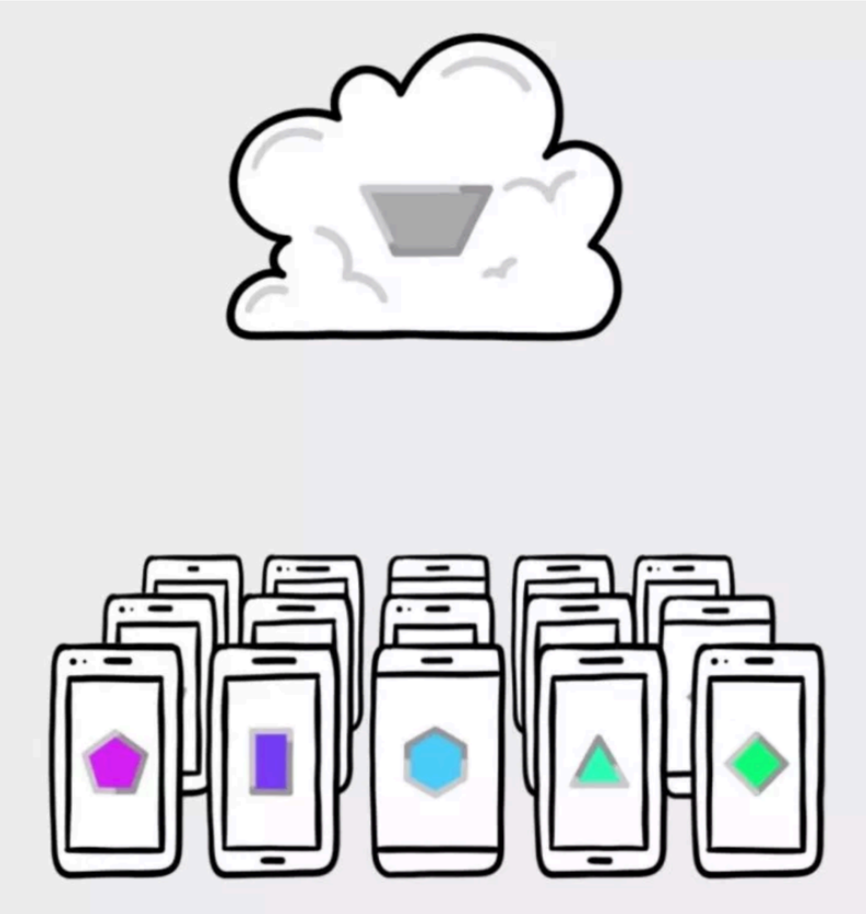
Motivating Examples



Problem: Google wants to train a model using users' mobile data.

Possible solution: Centralized learning
- Collect users' data.
- Train a model on the cluster.

Motivating Examples



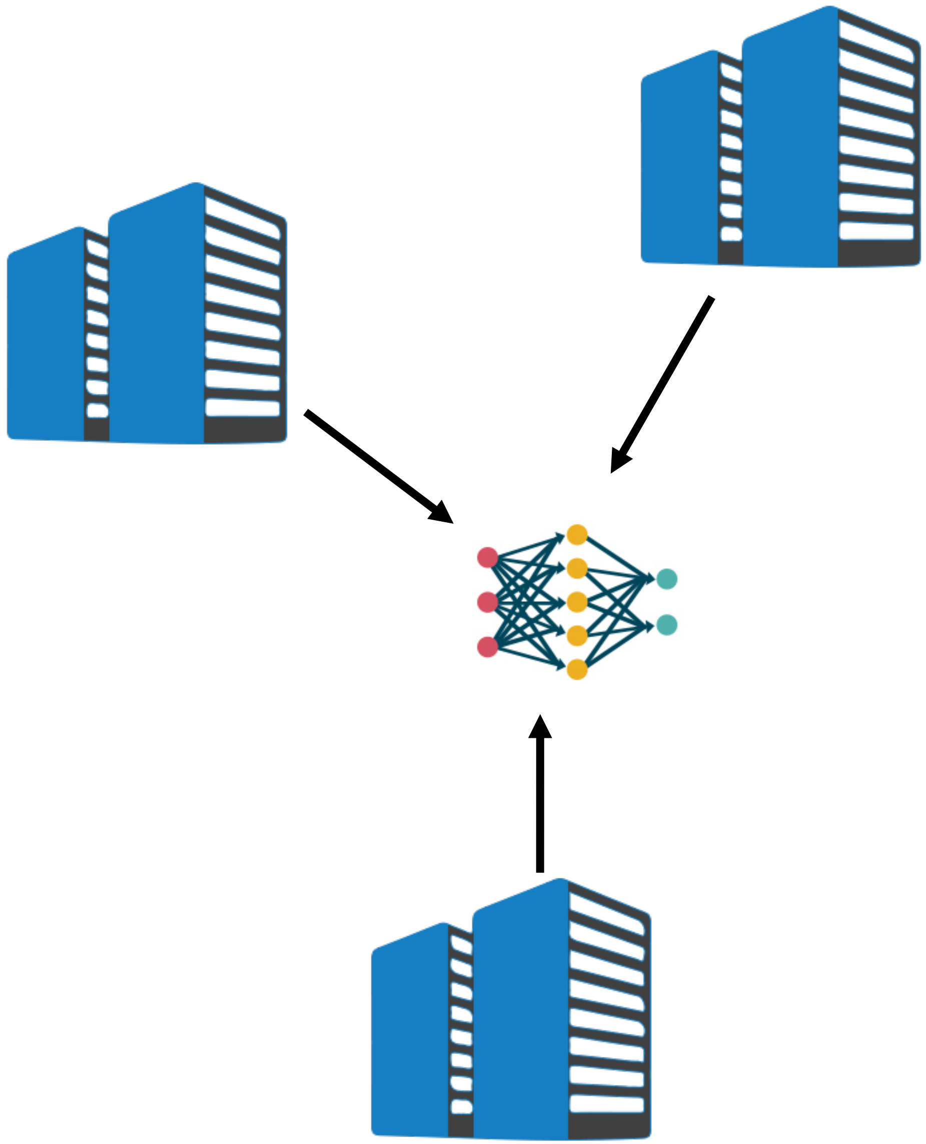
Problem: Google wants to train a model using users' mobile data.

Possible solution: Centralized learning

- Collect users' data.
- Train a model on the cluster.

Challenge: Users may refuse to upload their data, especially sensitive data, to Google's server.

Motivating Examples



Problem: Hospitals want to jointly train a model using medical data.

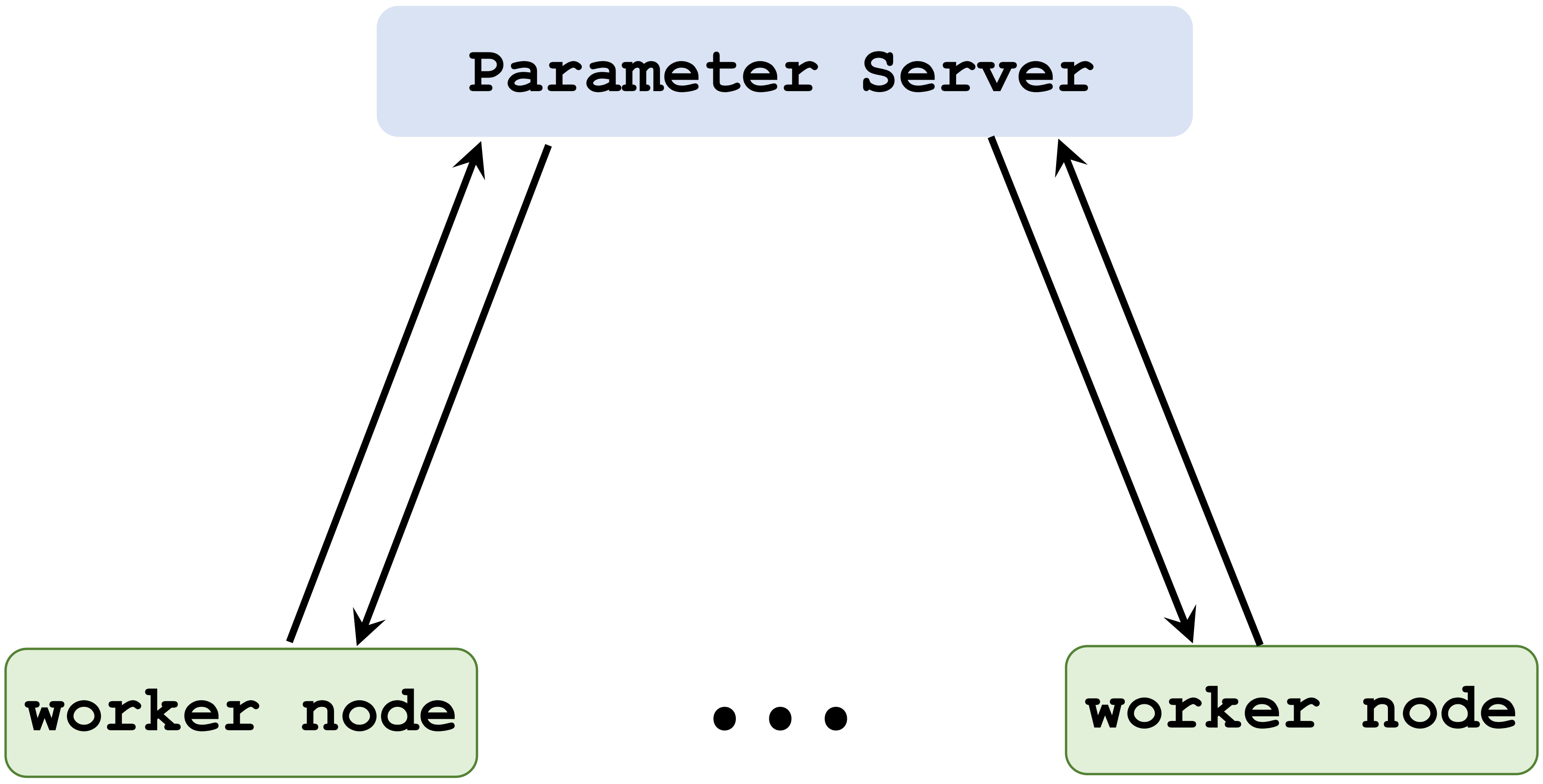
Possible solution: Centralized learning

- Aggregate the data.
- Train a model on the server.

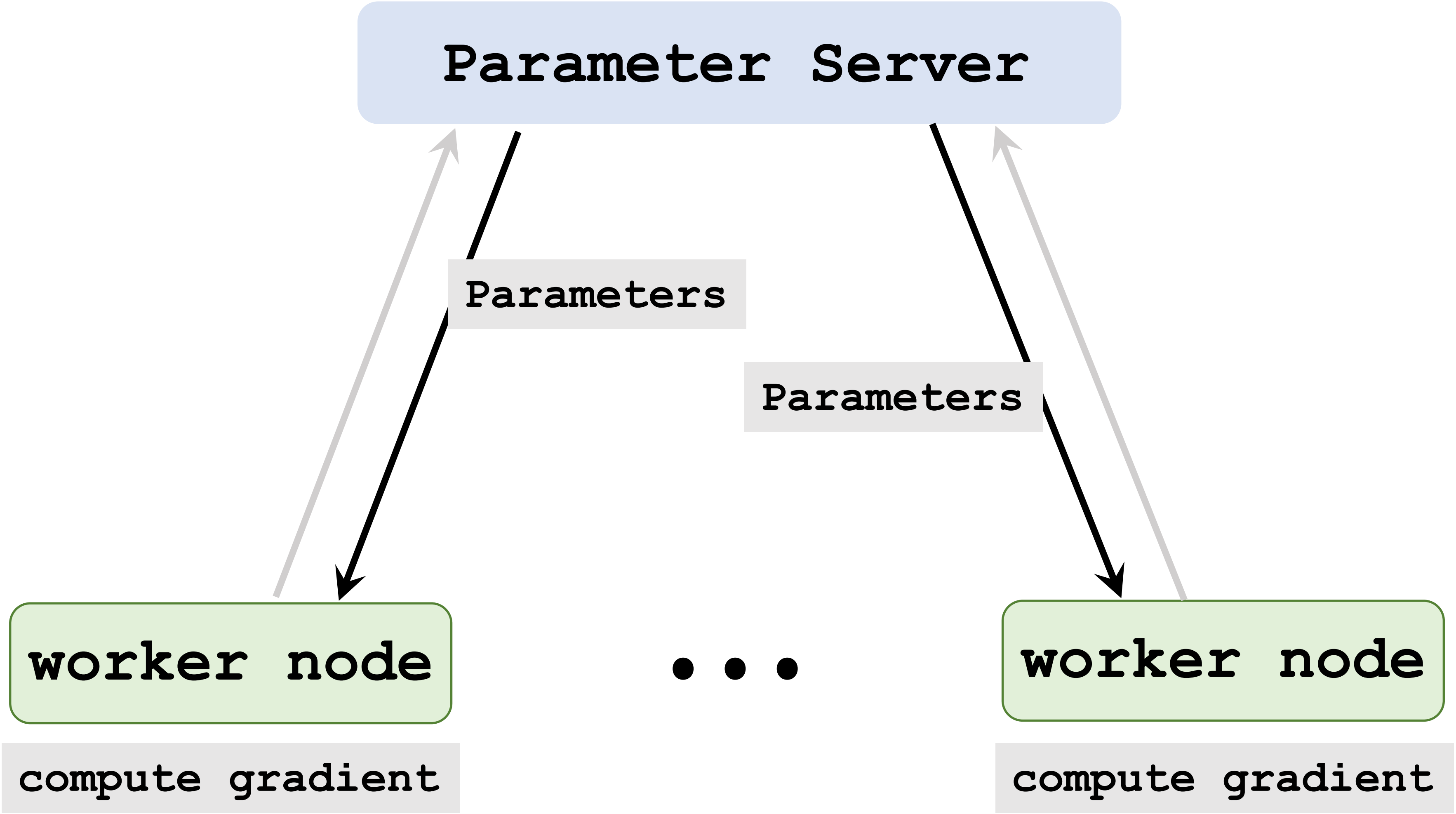
Challenge: Laws or policies may forbid giving patients' data to others.

Distributed Learning vs. Federated Learning

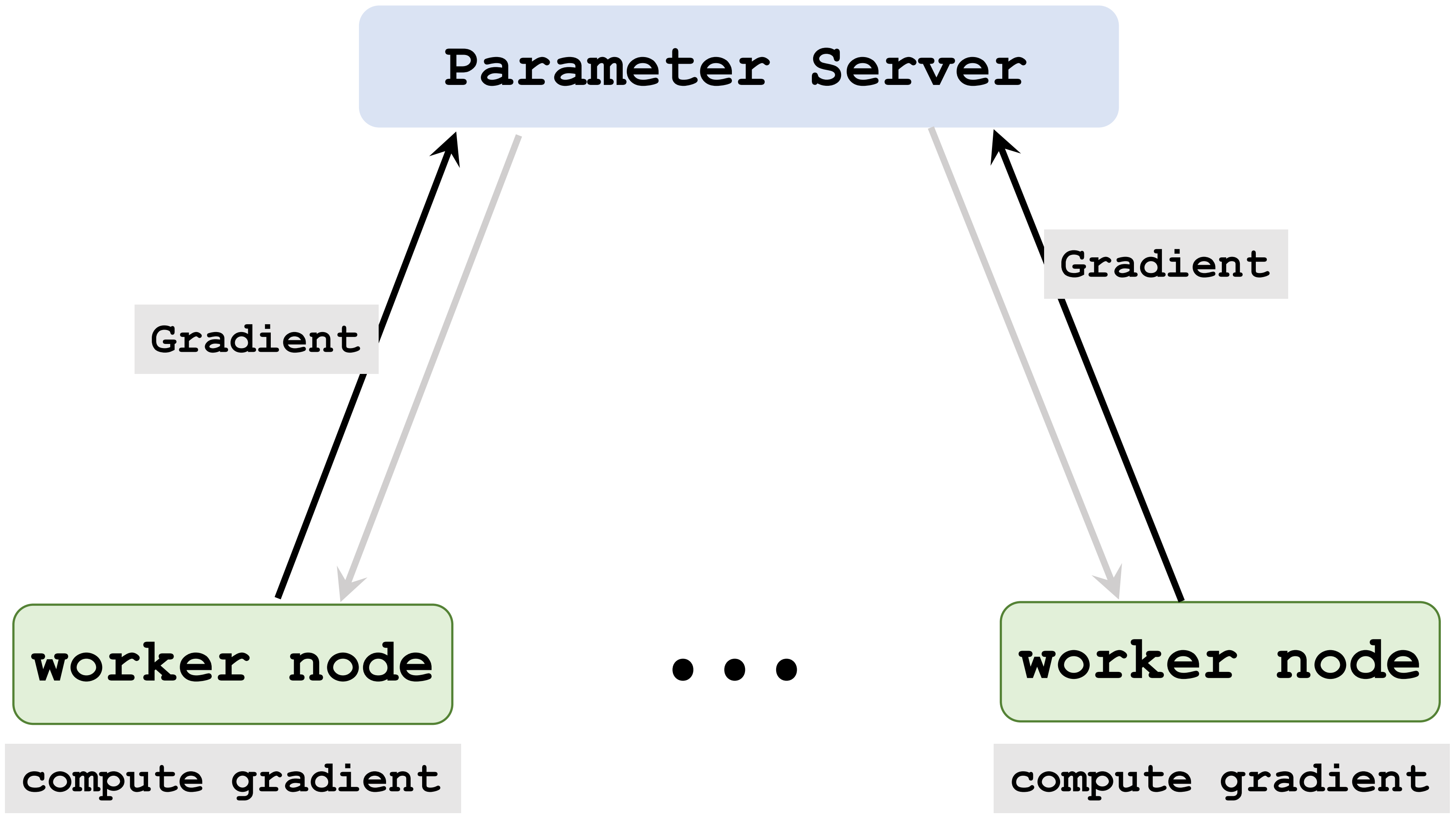
Distributed Learning



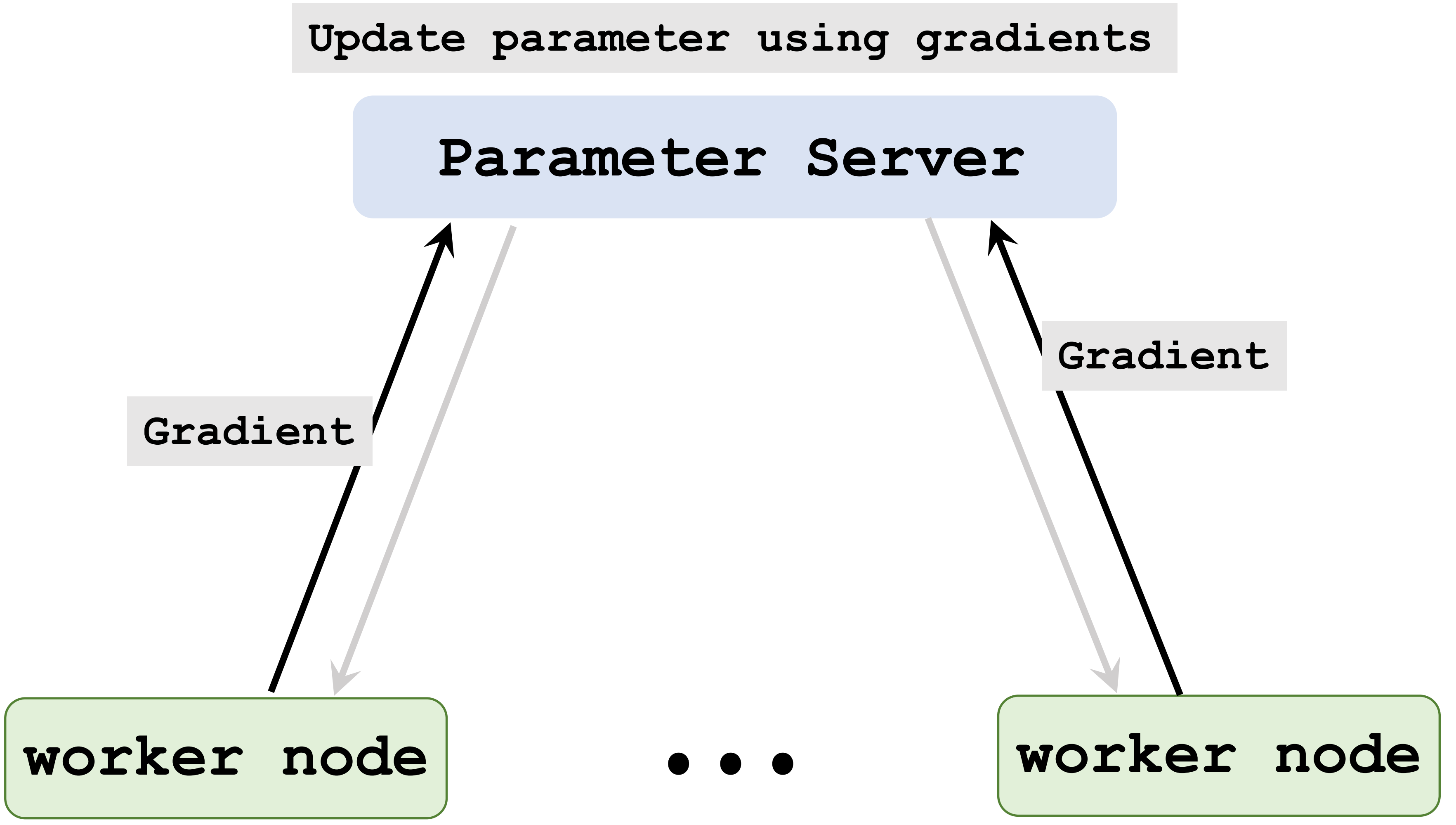
Distributed Learning



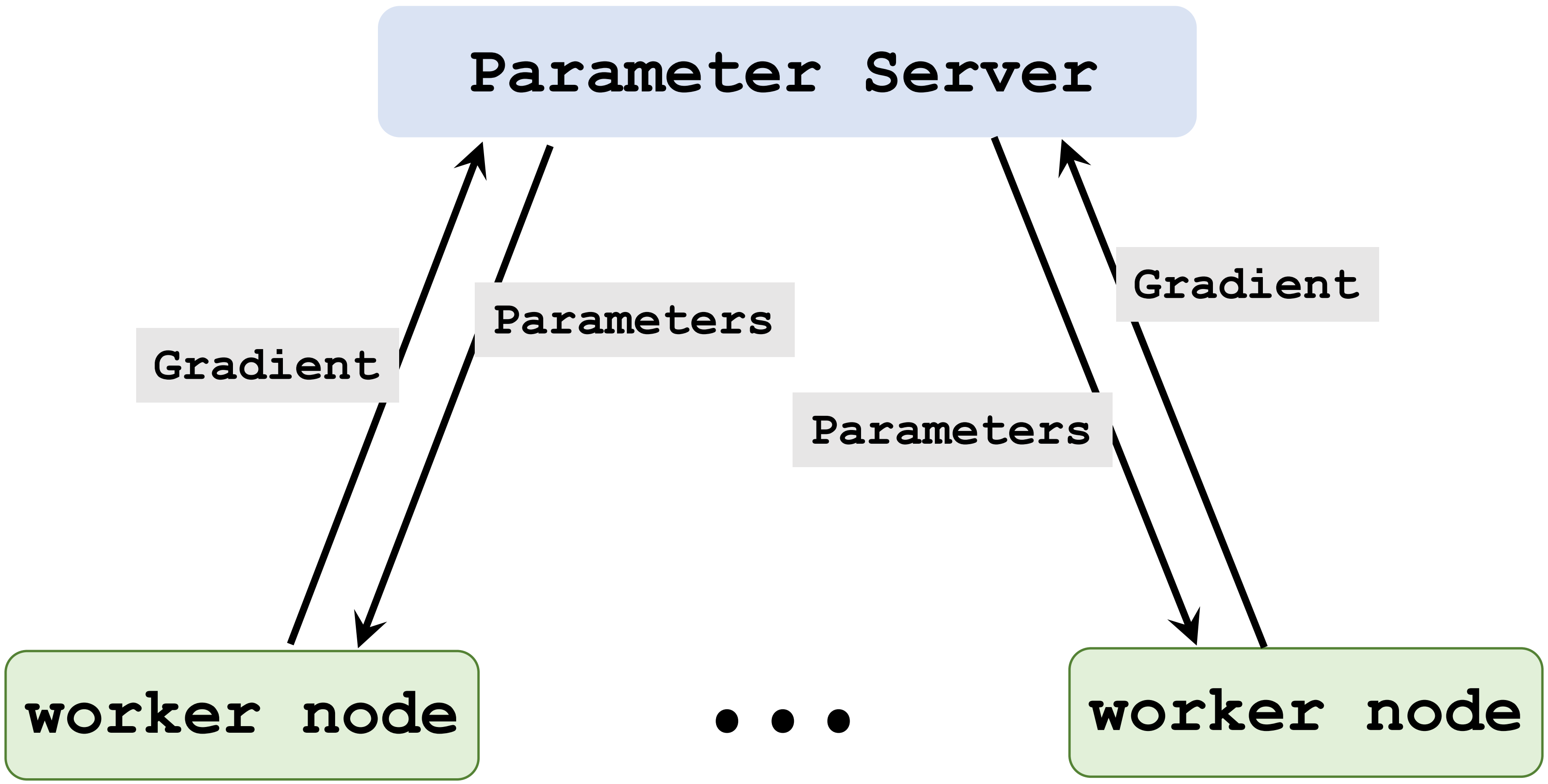
Distributed Learning



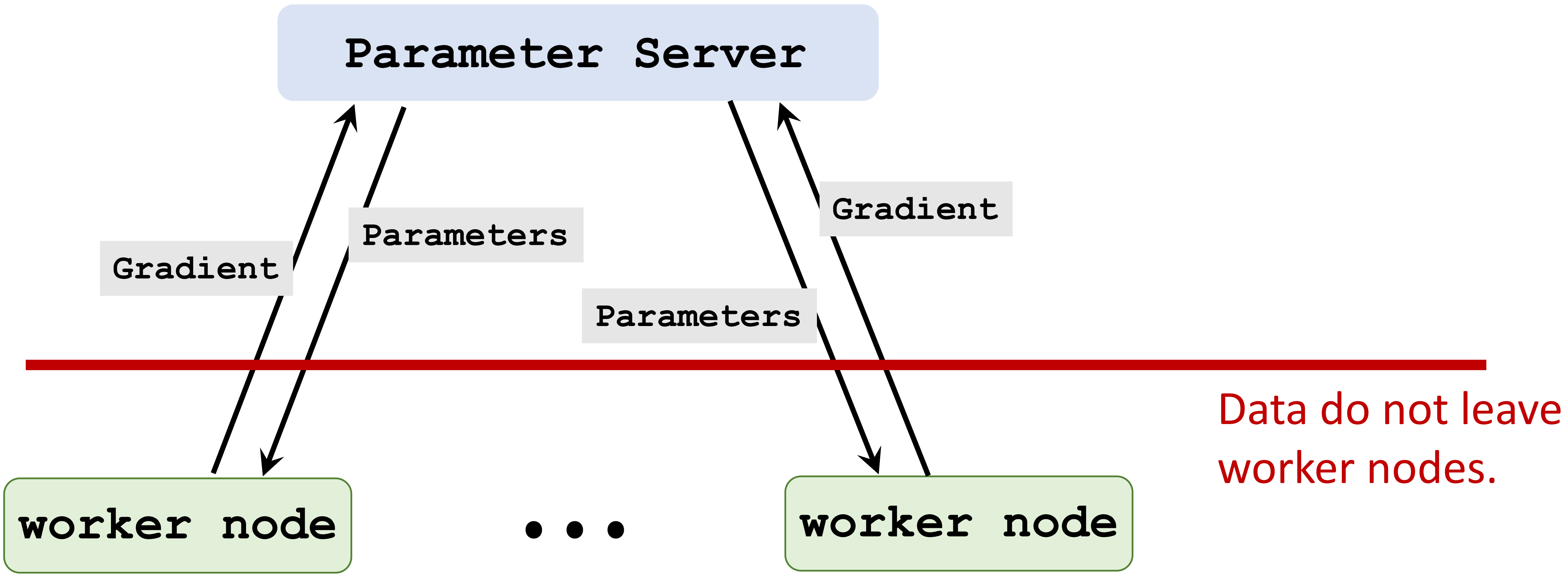
Distributed Learning



Distributed Learning



Distributed Learning



What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

1. Users have control over their device and data.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

1. Users have control over their device and data.
2. Worker nodes are unstable.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

1. Users have control over their device and data.
2. Worker nodes are unstable.
3. Communication cost is higher than computation cost.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

1. Users have control over their device and data.
2. Worker nodes are unstable.
3. Communication cost is higher than computation cost.
4. Data stored on worker nodes are not IID.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

1. Users have control over their device and data.
2. Worker nodes are unstable.
3. Communication cost is higher than computation cost.
4. Data stored on worker nodes are not IID.
5. The amount of data is severely imbalanced.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

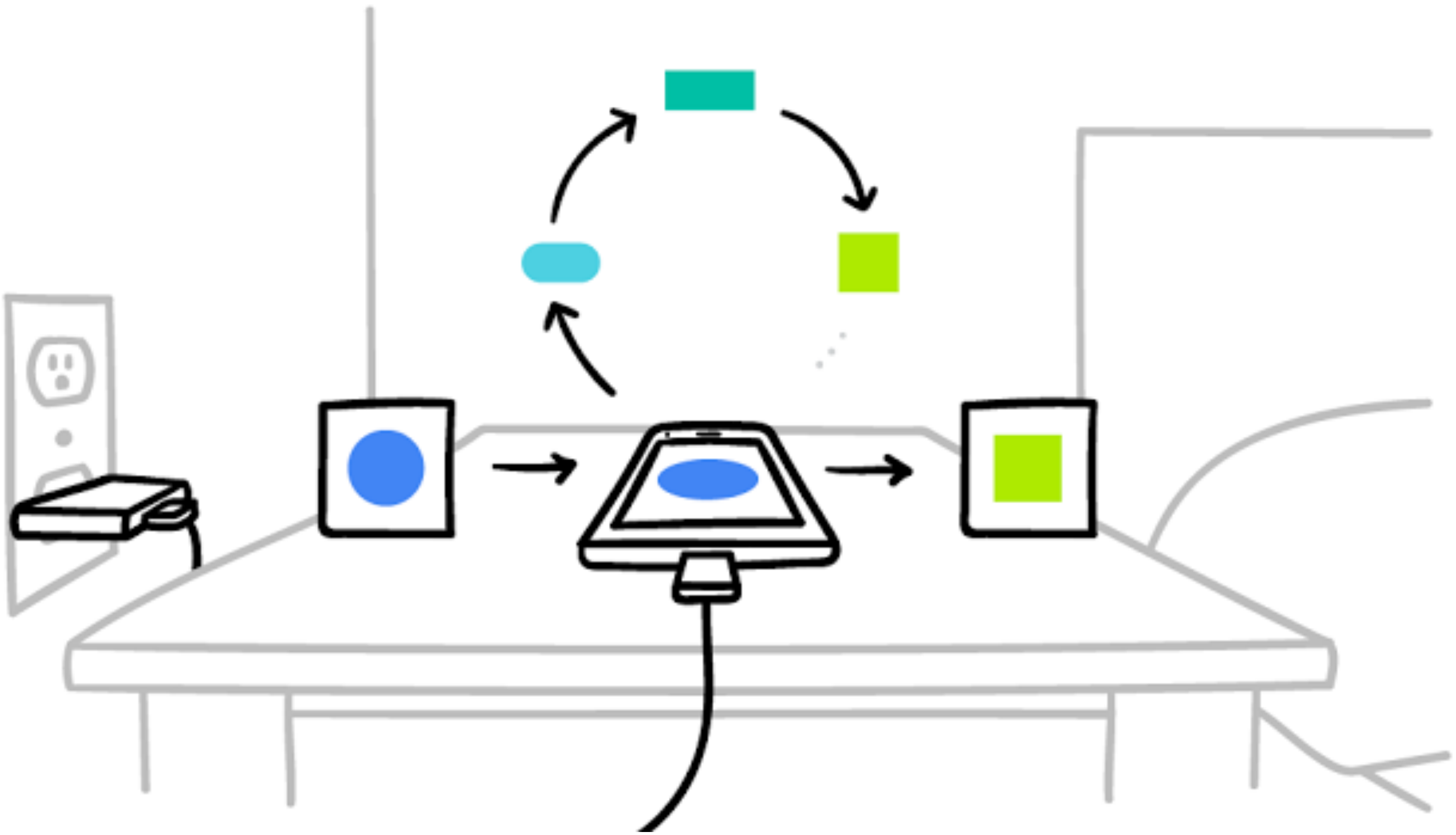
1. Users have control over their device and data.
2. Worker nodes are unstable.
3. Communication cost is higher than computation cost.
4. Data stored on worker nodes are not IID.
5. The amount of data is severely imbalanced.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

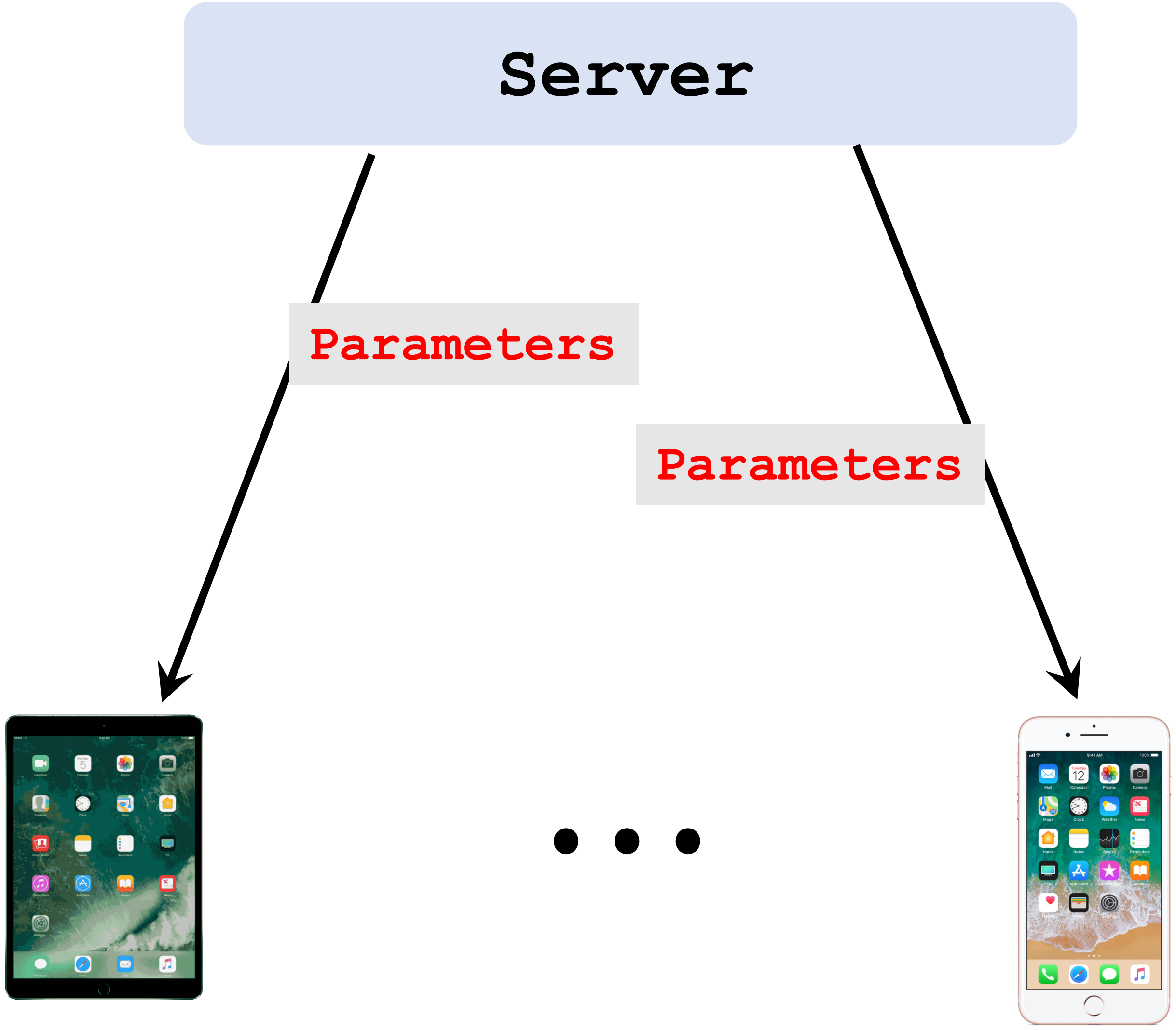
Research Direction 1: Communication-Efficiency

Trade computation for communication

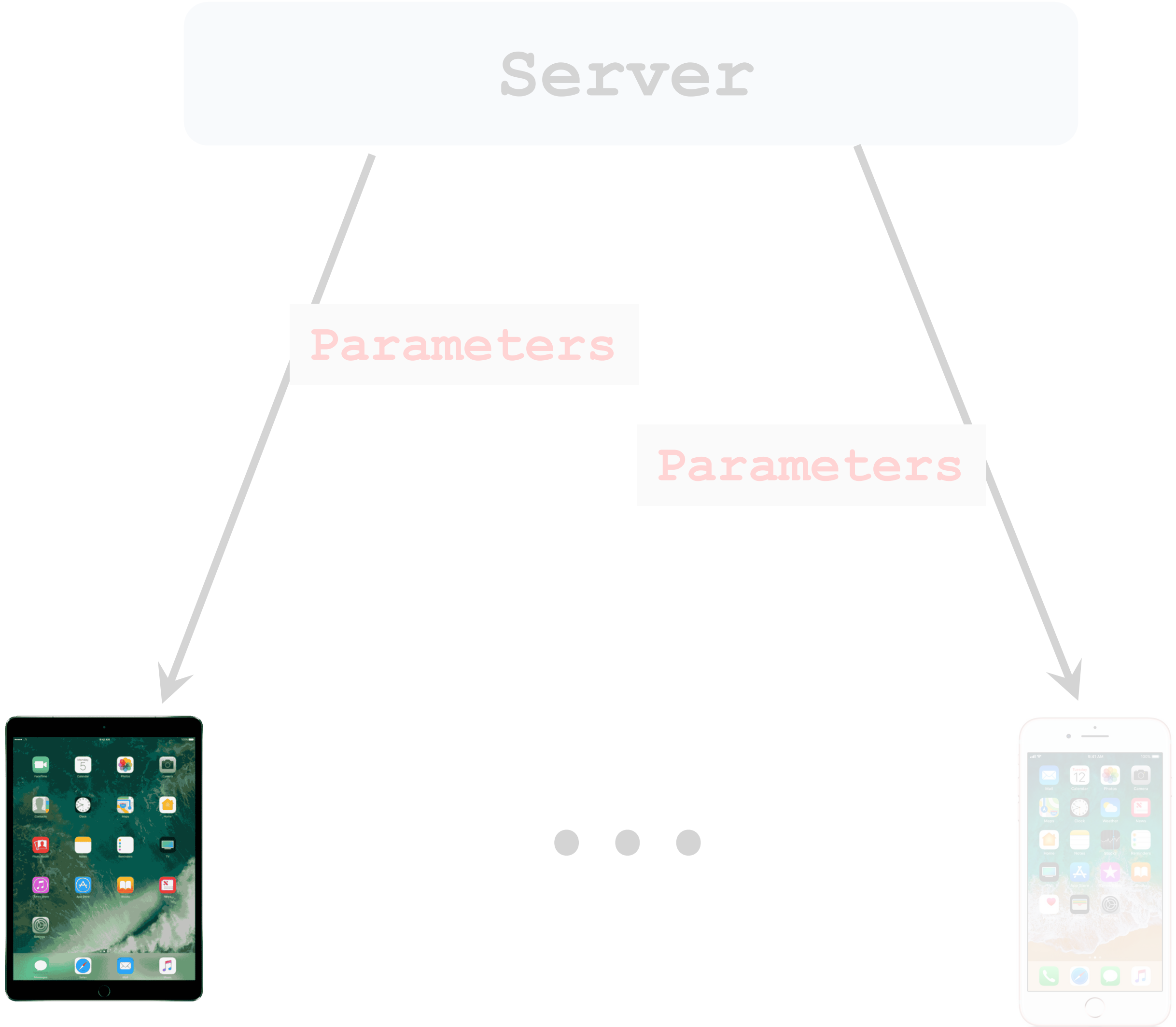


The image is From Google Research Blog

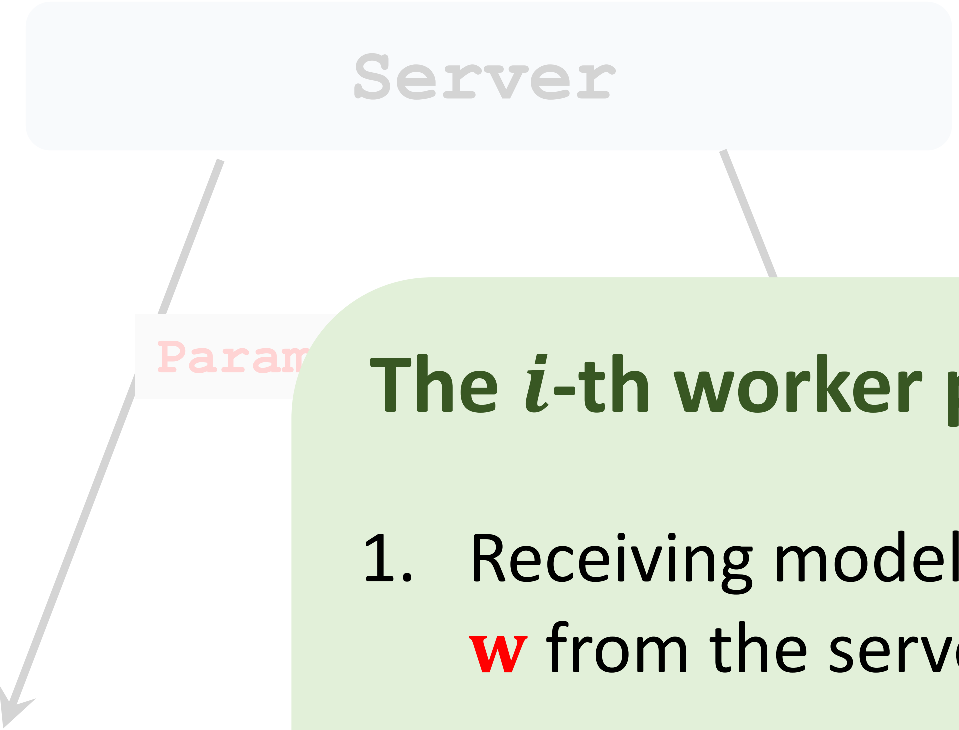
Let us recall parallel gradient descent



Let us recall parallel gradient descent



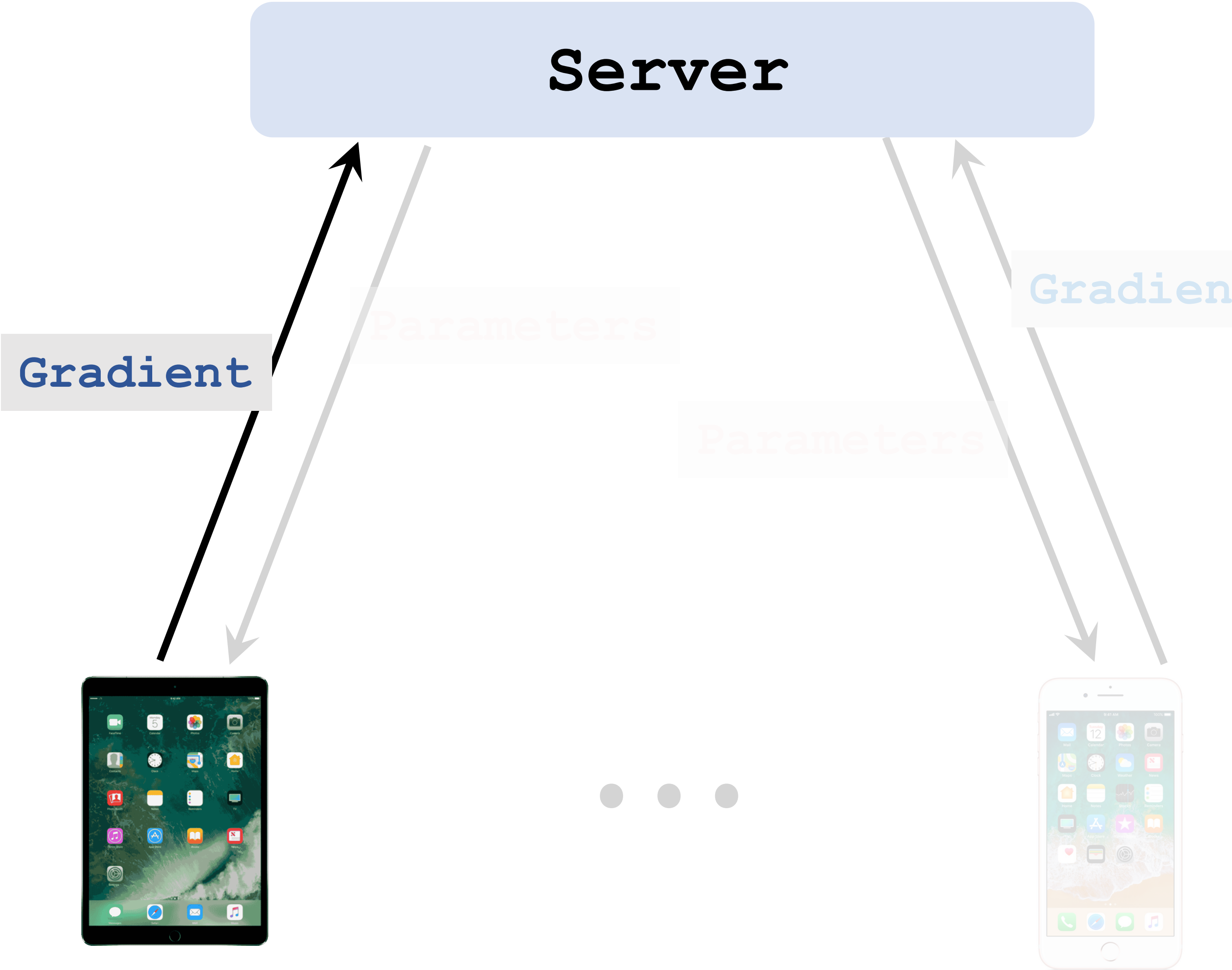
Let us recall parallel gradient descent



The i-th worker performs:

1. Receiving model parameters $\mathbf{w}$ from the server.
2. Using $\mathbf{w}$ and its local data to compute gradient $\mathbf{g}_i$.
3. Sending $\mathbf{g}_i$ to the server.

Let us recall parallel gradient descent

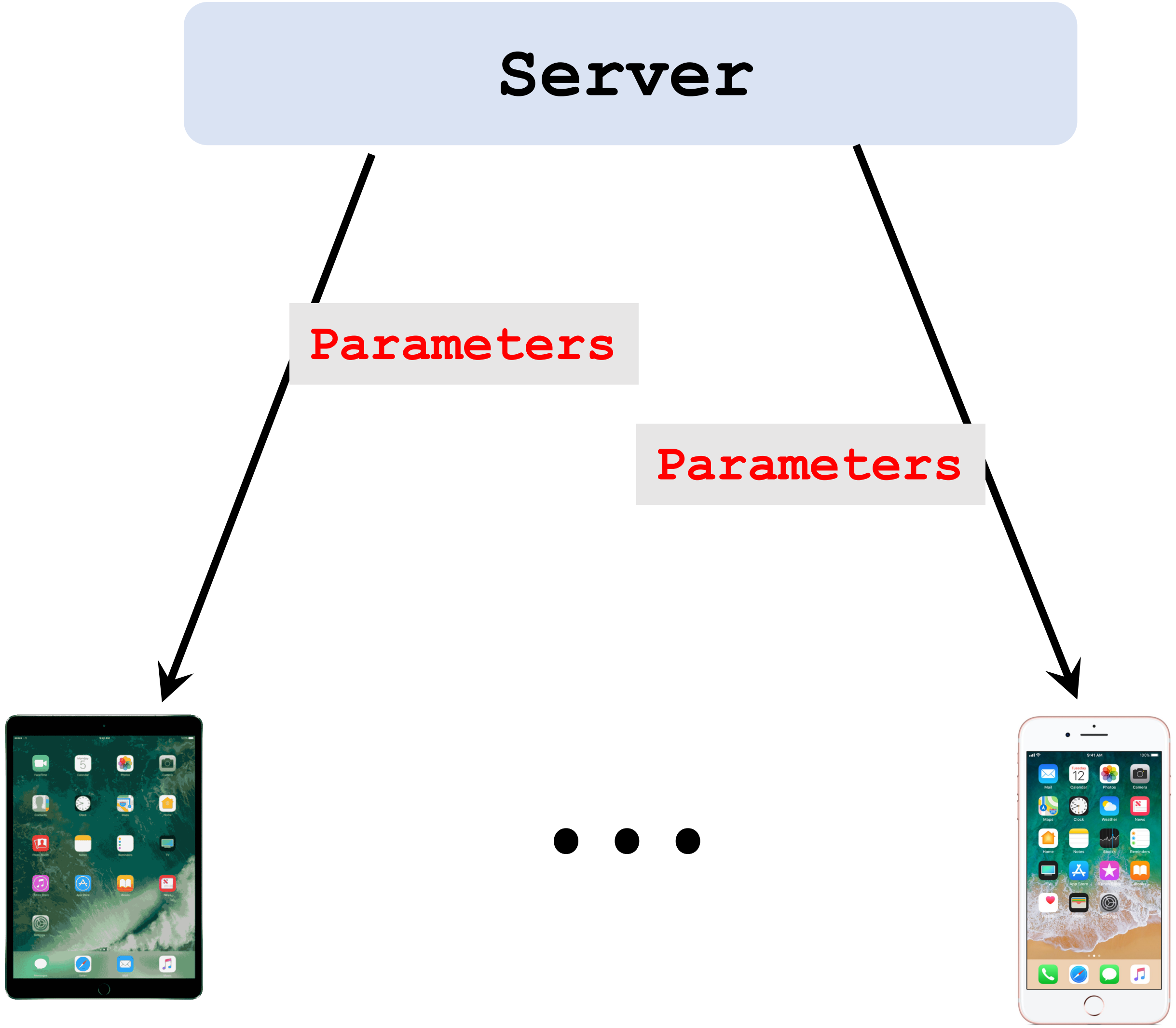


The server performs:

1. Receiving gradients $\mathbf{g}_1, \cdots, \mathbf{g}_m$ from all the m workers.
2. Computing $\mathbf{g} = \mathbf{g}_1 + \cdots + \mathbf{g}_m$.
3. Updating model parameters:

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \mathbf{g}.$$

Federated Averaging Algorithm



Federated Averaging Algorithm

Server

The i-th worker performs:

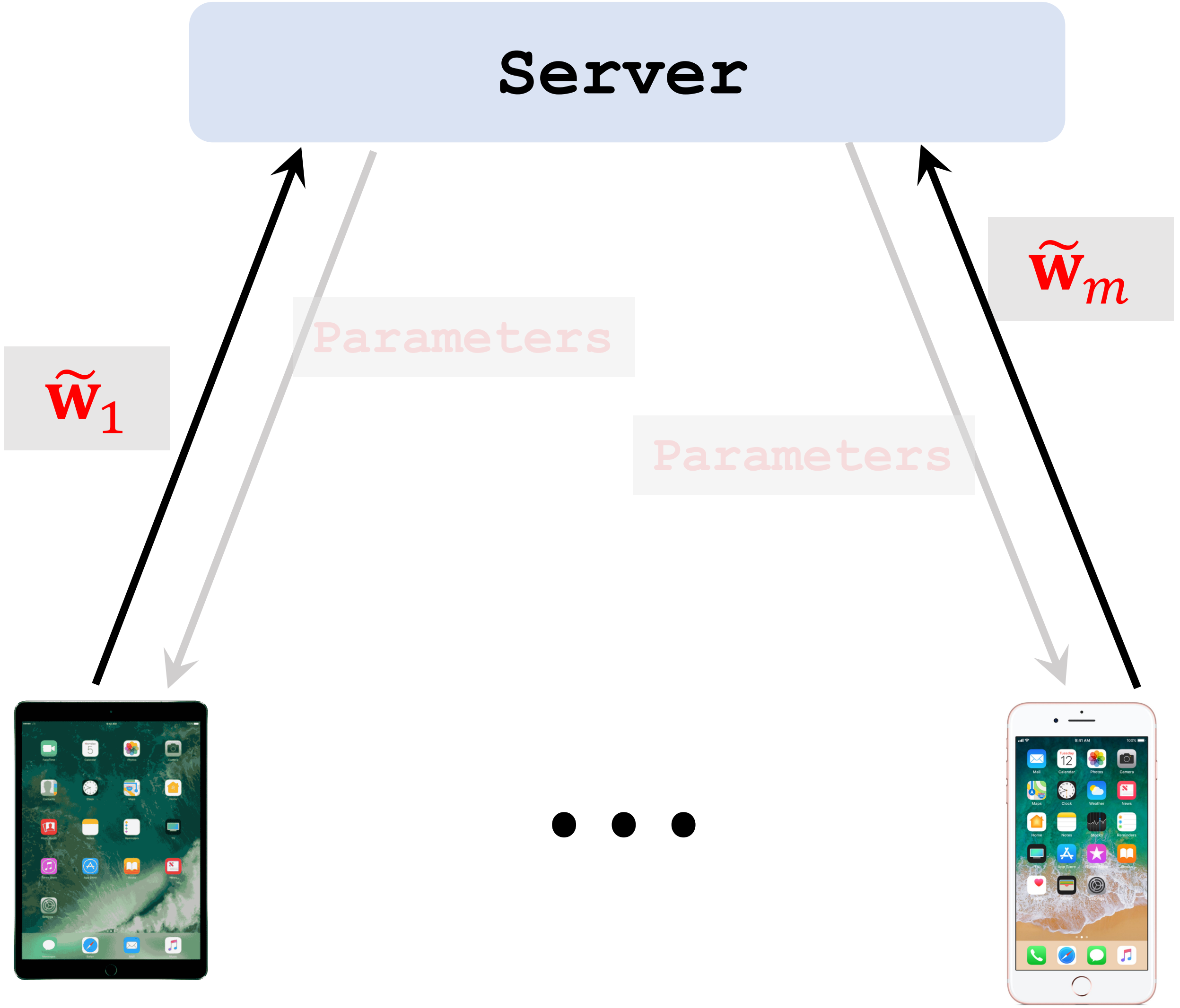
1. Receiving model parameters $\mathbf{w}$ from the server.
2. Repeating the followings:
   a) Using $\mathbf{w}$ and its local data to compute gradient $\mathbf{g}$.
   b) Local update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \mathbf{g}$.

Federated Averaging Algorithm

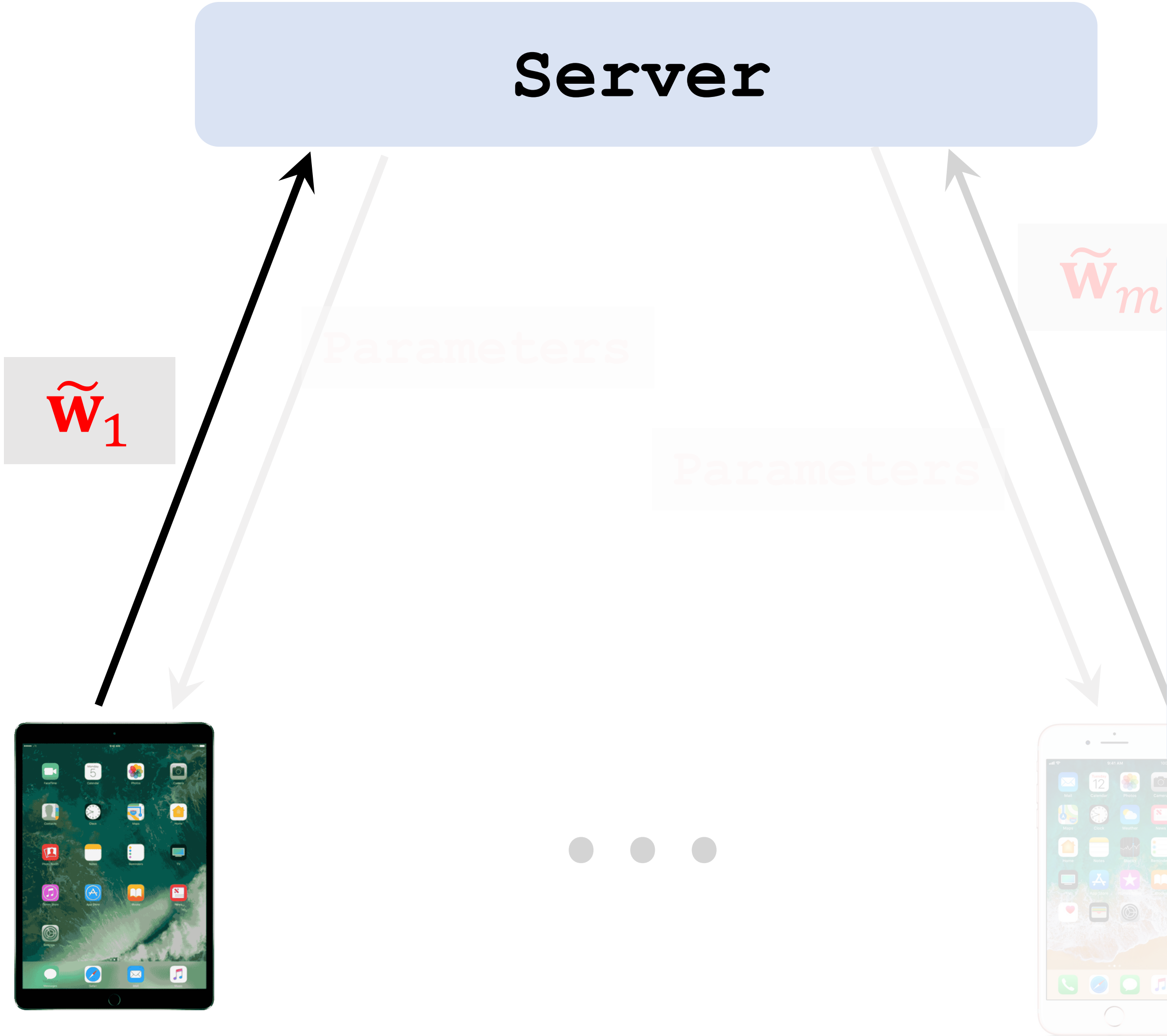
The i-th worker performs:

1. Receiving model parameters $\mathbf{w}$ from the server.
2. Repeating the followings:
   a) Using $\mathbf{w}$ and its local data to compute gradient $\mathbf{g}$.
   b) Local update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \mathbf{g}$.
3. Sending $\tilde{\mathbf{w}}_i = \mathbf{w}$ to the server.

Federated Averaging Algorithm



Federated Averaging Algorithm



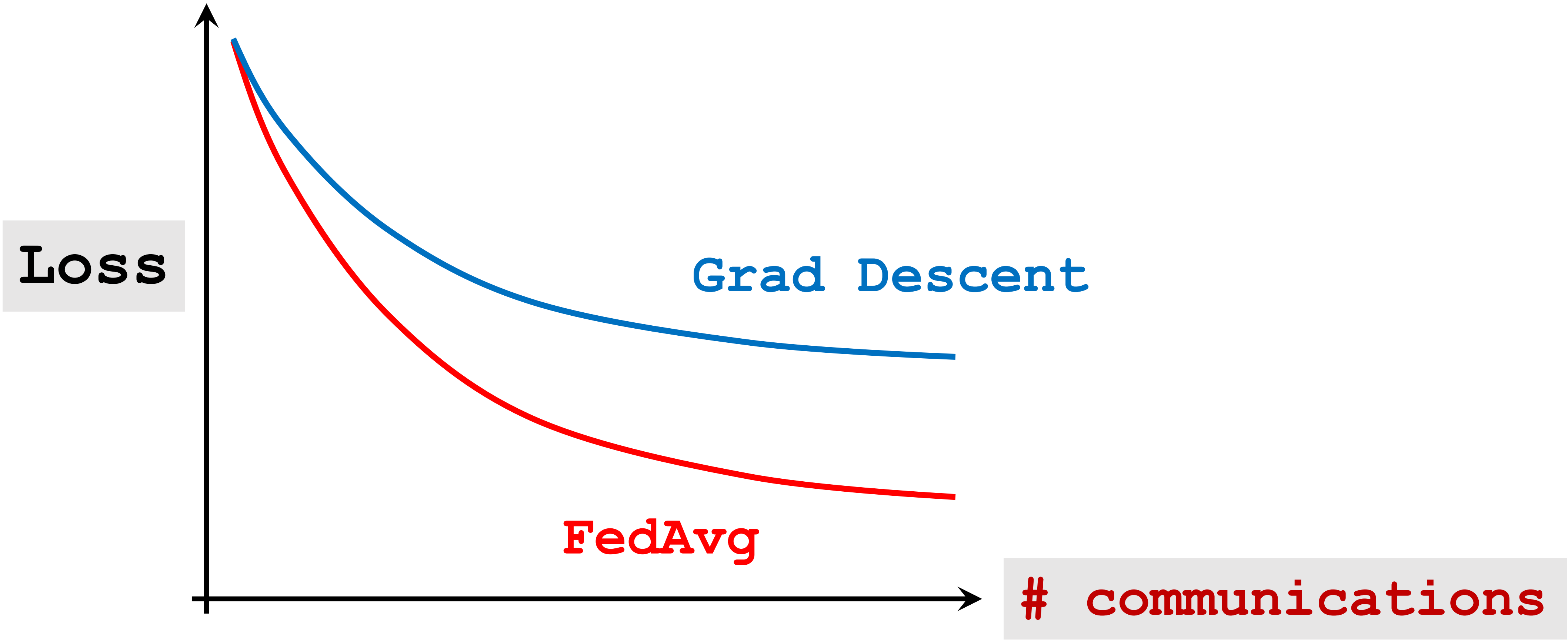
The server performs:

1. Receiving $\tilde{\mathbf{w}}_1, \cdots, \tilde{\mathbf{w}}_m$ from all the m workers.
2. Updating model parameters:

$$\mathbf{w} \leftarrow \frac{1}{m}(\tilde{\mathbf{w}}_1 + \cdots + \tilde{\mathbf{w}}_m).$$

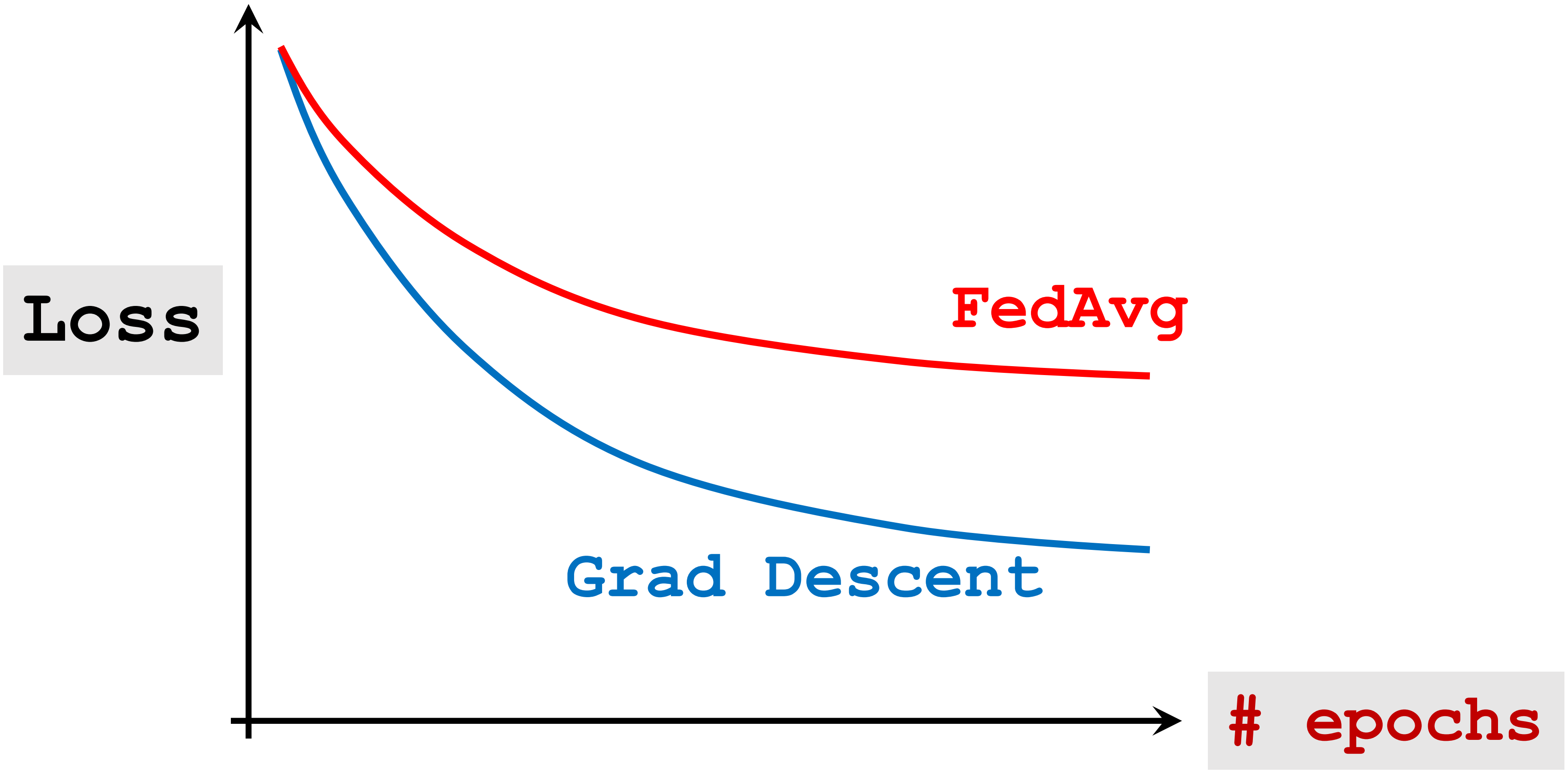
Computation vs. Communication

Measured by # communications, Federated Averaging is faster.



Computation vs. Communication

Measured by # epochs, Federated Averaging is slower.



Convergence of FedAvg

- The original paper [1] does not have theory.
- Paper [2] proved FedAvg converges for IID data.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Stich: Local SGD converges fast and communicates little. In *ICLR*, 2018.

Convergence of FedAvg

- The original paper [1] does not have theory.
- Paper [2] proved FedAvg converges for IID data.
- Paper [3] is the first to prove FedAvg (with SGD) converges for non-IID data.
- Paper [4] proved FedAvg (with GD) converges for non-IID data.

References

1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Stich: Local SGD converges fast and communicates little. In *ICLR*, 2018.
3. Li and others: On the convergence of FedAvg on non-IID data. *arXiv*, 2019.
4. Khaled and others: First analysis of local GD on heterogeneous data. *arXiv*, 2019.

Communication-Efficient Algorithms

Communication-efficient algorithms for distributed learning, e.g.,

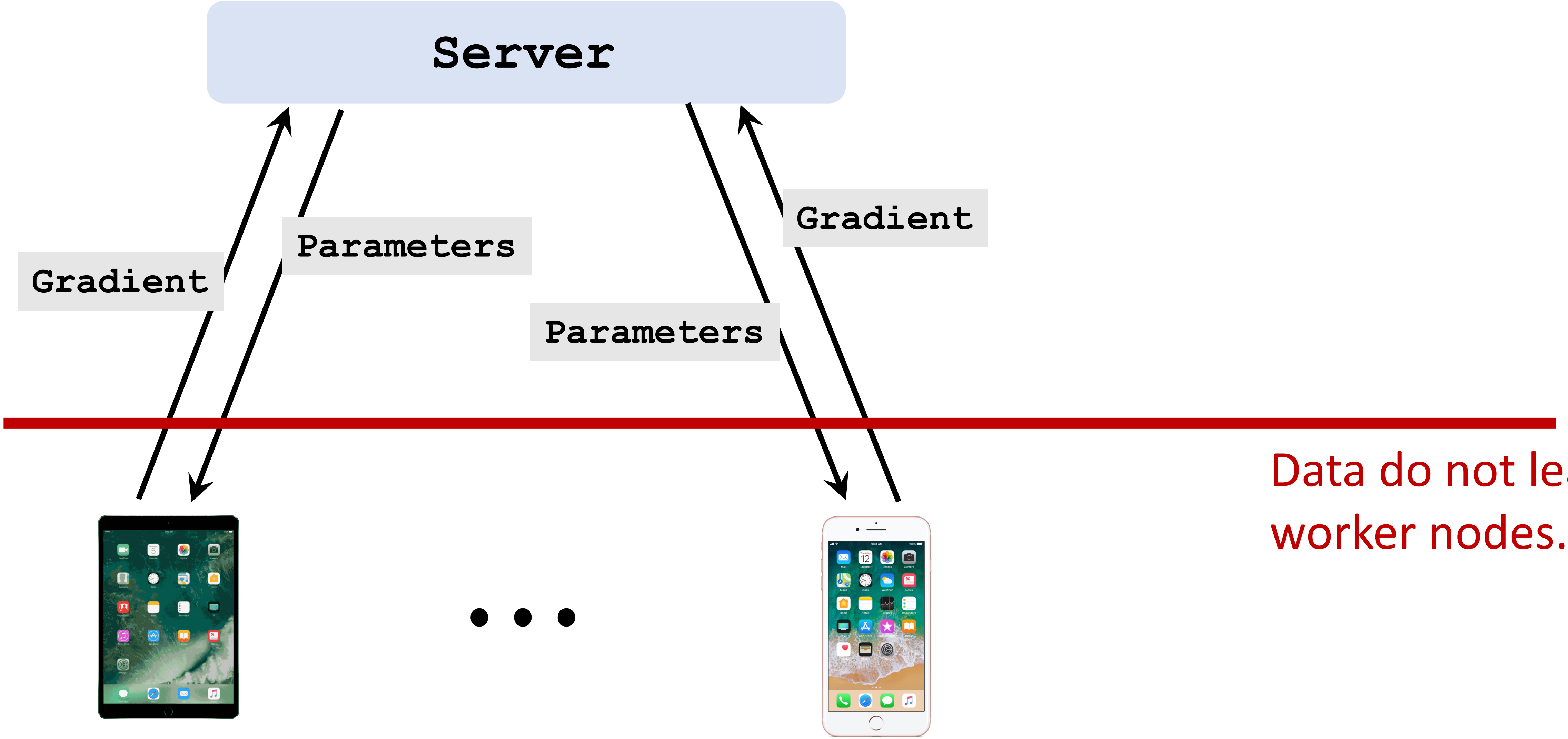
- Approximate Newton's algorithms [1, 2, 3].
- Primal-dual algorithms [4].
- One-shot averaging [5].

Reference:

1. Shamir, Srebro, & Zhang: Communication efficient distributed optimization using an approximate Newton-type method. In *ICML*, 2014.
2. Wang and others: GIANT: Globally improved approximate newton method for distributed optimization. In *NIPS*, 2018.
3. Mahajan and others: An efficient distributed learning algorithm based on effective local functional approximations. *Journal of Machine Learning Research*, 2019.
4. Smith and others: CoCoA: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 2018.
5. Zhang, Duchi, & Wainwright: Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 2013.

Research Direction 2: Privacy

Is federated learning (FL) safe?



Data do not leave worker nodes.

Is federated learning (FL) safe?

Gradient carries information in the training data.

- Least squares regression:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} l(\mathbf{w}, \mathbf{x}_i, y_i), \quad \text{where } l(\mathbf{w}, \mathbf{x}_i, y_i) = \frac{1}{2}\left(\mathbf{x}_i^T \mathbf{w} - y_i\right)^2.$$

- Stochastic gradient:

$$\mathbf{g}_i = \frac{\partial\, l(\mathbf{w}, \mathbf{x}_i, y_i)}{\partial\, \mathbf{w}} = \left(\mathbf{x}_i^T \mathbf{w} - y_i\right) \mathbf{x}_i.$$

Is federated learning (FL) safe?

- If an ML model is useful, it must reveal information about the data on which it was trained [1].

References

1. Melis et al. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security & Privacy*, 2019.

Is federated learning (FL) safe?

- If an ML model is useful, it must reveal information about the data on which it was trained [1].
- Training data can be reversely inferred from the model [2].

References

1. Melis et al. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security & Privacy*, 2019.
2. Fredrikson et al. Model inversion attacks that exploit confidence information and basic countermeasures. In *CCS*, 2015.

Is federated learning (FL) safe?

- If an ML model is useful, it must reveal information about the data on which it was trained [1].
- Training data can be reversely inferred from the model [2].
- In FL, gradients and model parameters leak users' data [1, 3].

References

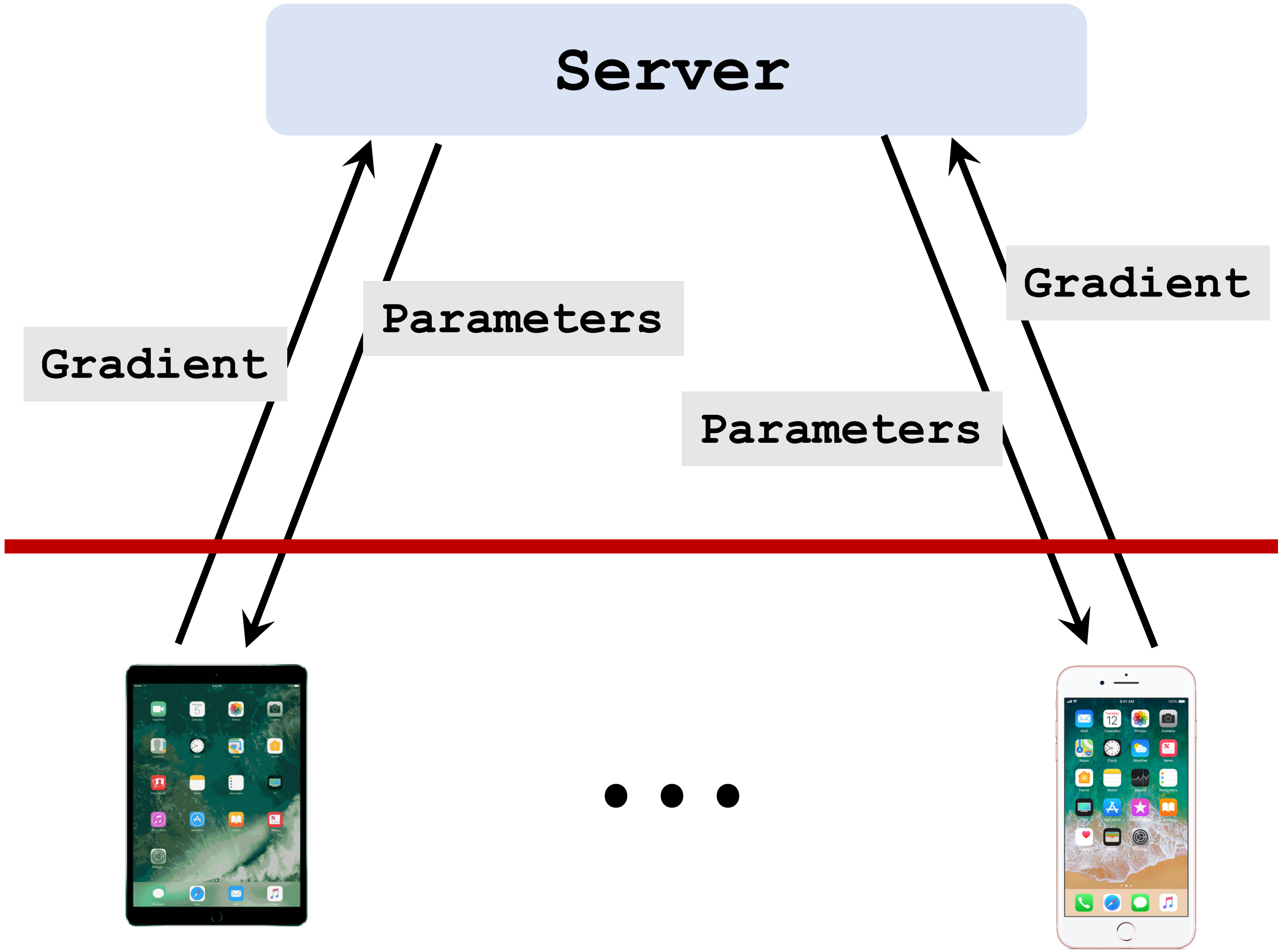
1. Melis et al. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security & Privacy*, 2019.
2. Fredrikson et al. Model inversion attacks that exploit confidence information and basic countermeasures. In *CCS*, 2015.
3. Hitaj et al. Deep models under the GAN: information leakage from collaborative deep learning. In *ACM SIGSAC Conference on Computer and Communications Security*, 2017.

How is privacy disclosed?

References

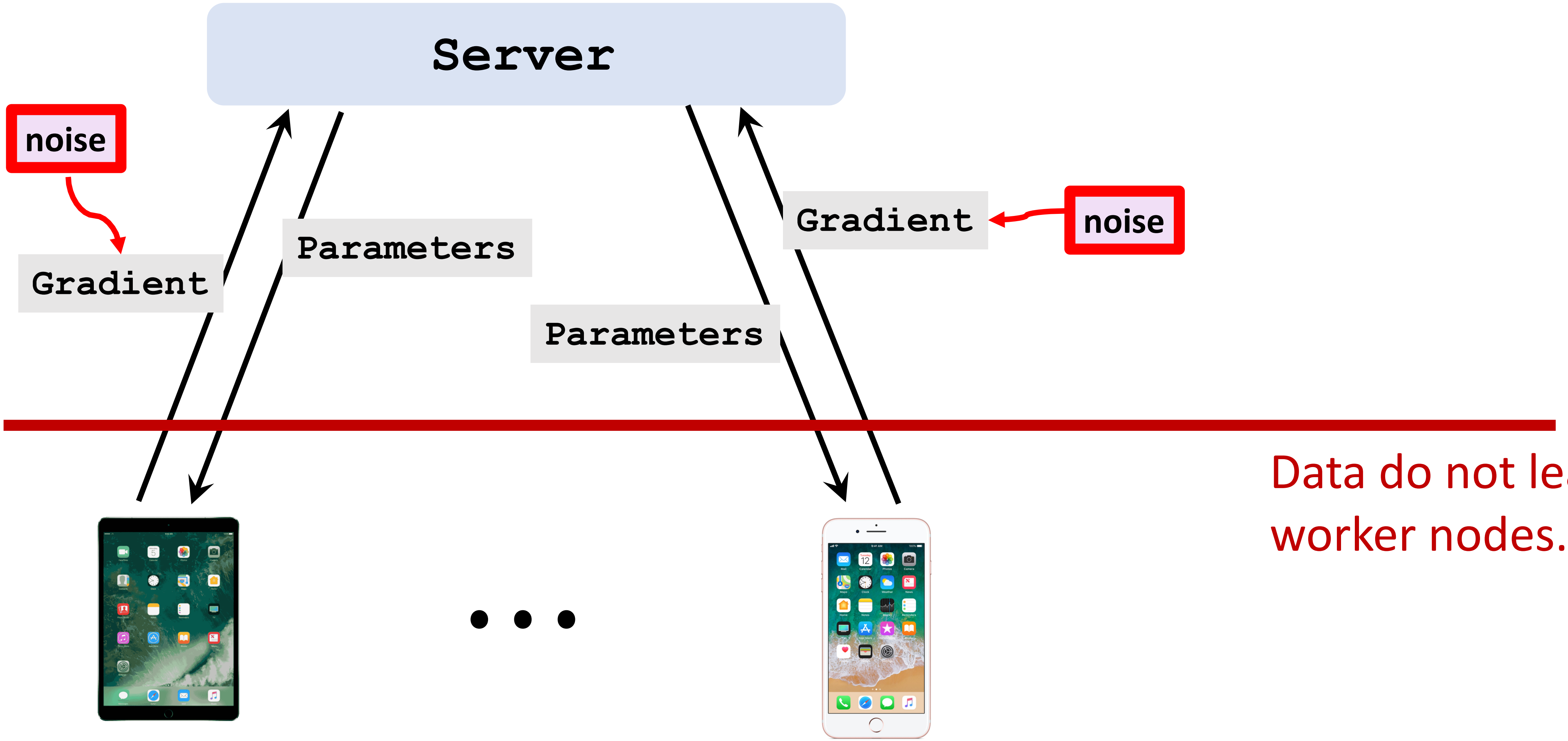
1. Melis et al. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security & Privacy*, 2019.

Can the attacks be defended?



Data do not leave worker nodes.

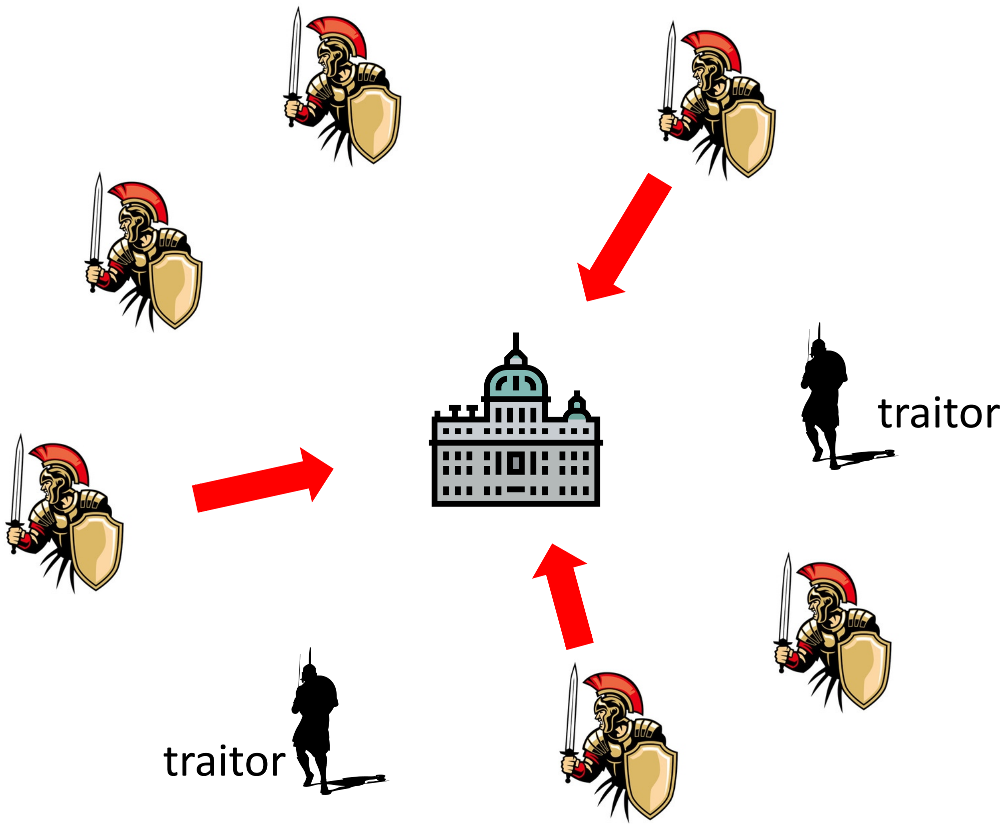
Can the attacks be defended?



Data do not leave worker nodes.

Research Direction 3: Adversarial Robustness

Byzantine Generals Problem



Reference

- Lamport, Shostak, & Pease: The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 1982.

Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].

References

1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.

Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].
- Attack 2: Model poisoning attack [2].

References

1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji and others: Analyzing federated learning through an adversarial lens. In *ICML*, 2019.

Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].
- Attack 2: Model poisoning attack [2].

- Defense 1: Server check validation accuracy.

References

1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji and others: Analyzing federated learning through an adversarial lens. In *ICML*, 2019.

Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].
- Attack 2: Model poisoning attack [2].

- Defense 1: Server check validation accuracy.
- Defense 2: Server check gradient statistics.

References

1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji and others: Analyzing federated learning through an adversarial lens. In *ICML*, 2019.

Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].
- Attack 2: Model poisoning attack [2].

- Defense 1: Server check validation accuracy.
- Defense 2: Server check gradient statistics.
- Defense 3: Byzantine-tolerant aggregation [3, 4, 5].

References

1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji and others: Analyzing federated learning through an adversarial lens. In *ICML*, 2019.
3. Blanchard, Guerraoui, & Stainer: Machine learning with adversaries: Byzantine tolerant gradient descent. In *NIPS*, 2017.
4. Chen, Su, & Xu: Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *In Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2017.
5. Yin and others: Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, 2018.

Summary

What is federated learning (FL)?

- FL is a kind of distributed learning.
- Objective: jointly learn a model without sharing data.
- FL has unique challenges, e.g.,
  - non-IID data,
  - slow communication.

Research Directions

- Direction 1: Communication-efficient algorithms.

Research Directions

- Direction 1: Communication-efficient algorithms.
- Direction 2: Defense against privacy leakage.

Research Directions

- Direction 1: Communication-efficient algorithms.
- Direction 2: Defense against privacy leakage.
- Direction 3: Robustness to Byzantine faults.

Thank you!

Reference (Communication Efficiency)

1. McMahan, Moore, Ramage, Hampson, & Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Stich. Local SGD converges fast and communicates little. In *ICLR*, 2018.
3. Li, Sahu, Talwalkar, & Smith. Federated optimization in heterogeneous networks. *arXiv*, 2018.
4. Wang & Joshi. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *arXiv*, 2018.
5. Fan & Cong. On the convergence properties of a k-step averaging stochasticgradient descent algorithm for nonconvex optimization. In *IJCAI*, 2018.
6. Lin, Stich, and Jaggi. Don’t use large mini-batches, use local SGD. arXiv, 2018.
7. Li, Huang, Yang, Wang, & Zhang. On the convergence of FedAvg on non-IID data. *arXiv*, 2019.
8. Khaled, Mishchenko, Richtárik. First analysis of local GD on heterogeneous data. *arXiv*, 2019.
9. Yu, Yang, Zhu. Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *AAAI*, 2019.

…

(And many other work which I am unaware of.)

Reference (Privacy Leakage)

1. Hitaj, Ateniese, & Perez-Cruz. Deep models under the GAN: information leakage from collaborative deep learning. In *ACM SIGSAC Conference on Computer and Communications Security*, 2017.
2. Melis, Song, Cristofaro, & Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security & Privacy*, 2019.
3. Zhu, Liu, & Han. Deep leakage from gradients. In *NIPS*, 2019.
4. Orekondy, Oh, Zhang, Schiele, & Fritz. Gradient-Leaks: Understanding and controlling deanonymization in federated learning. *arXiv*, 2018.
5. Ateniese, Felici, Mancini, Spognardi, Villani, & Vitali. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 2015.
6. Fredrikson, Jha, & Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *CCS*, 2015.
7. Ganju, Wang, Yang, Gunter, & Borisov. Property Inference Attacks on Fully Connected Neural Networks using Permutation Invariant Representations. In *CCS*, 2018.
8. Jia, Salem, Backes, Zhang, & Gong. Property inference attacks on fully connected neural networks using permutation invariant representations. In *CCS*, 2019.

Reference (Adversarial Robustness)

1. Shafah, Huang, Najibi, Suciu, Studer, Dumitras, Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji, Chakraborty, Mittal, & Calo. Analyzing federated learning through an adversarial lens. In *ICML*, 2019.
3. Koh, Steinhardt, & Liang. Stronger data poisoning attacks break data sanitization defenses. *arXiv*, 2018.
4. Fang, Cao, Jia, & Gong. Local model poisoning attacks to Byzantine-robust federated learning. *arXiv*, 2019.
5. Blanchard, Guerraoui, & Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *NIPS*, 2017.
6. Chen, Su, & Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *In Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2017.
7. Yin, Chen, Ramchandran, Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, 2018.