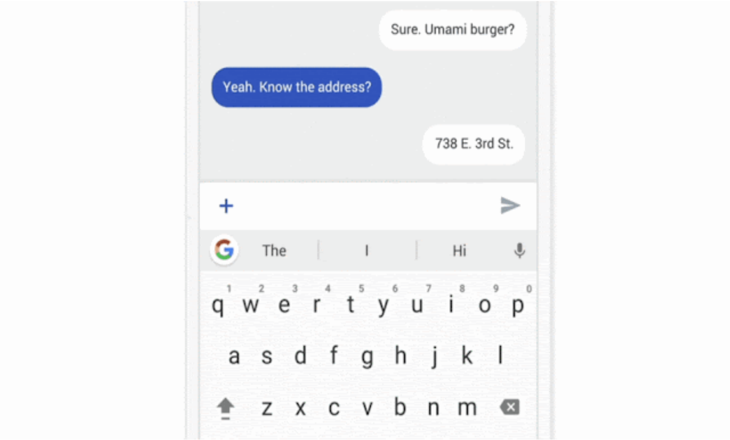


https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

Federated Learning

马兴军，复旦大学 计算机学院

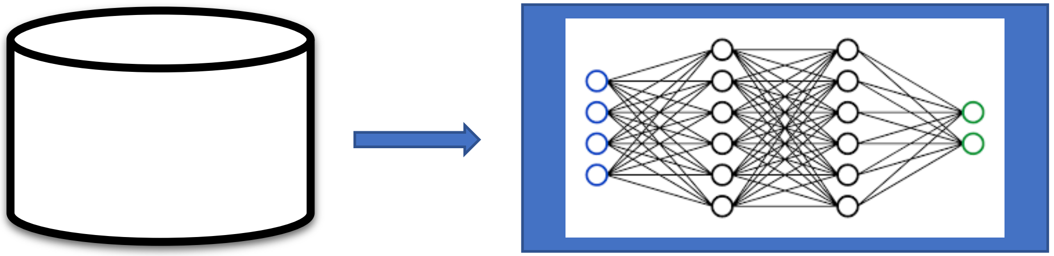
Recap: week 11

- Common Tampering and Deepfakes
- Image Manipulation Detection
- Video Manipulation Detection

This Week

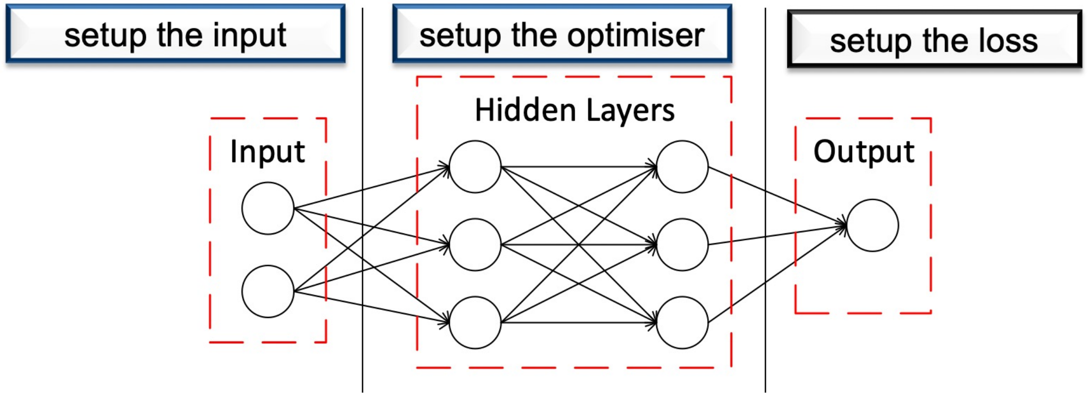
- Federated Learning
- Privacy in Federated Learning
- Robustness in Federated Learning
- Challenges and Future Research

Traditional Machine Learning



Data　　　　Model

Data and model in one single place



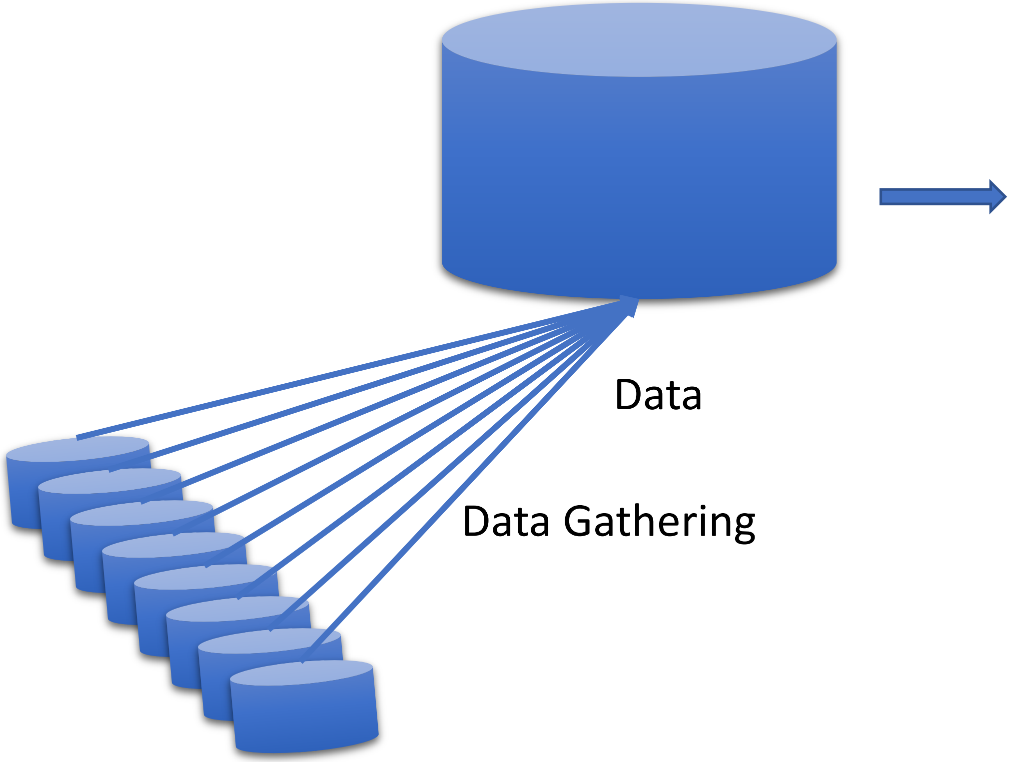
$$\min_{\theta} \sum_{(x_i,\ y_i) \in D_{train}} L(f_\theta(x_i),\ y_i)$$

Cross entropy:

$$L(f_\theta(x_i), y_i) = \sum_{j=1}^{C} y_{ij} \log \boldsymbol{f}_{\boldsymbol{\theta}}^{j}(\boldsymbol{x}_i)$$

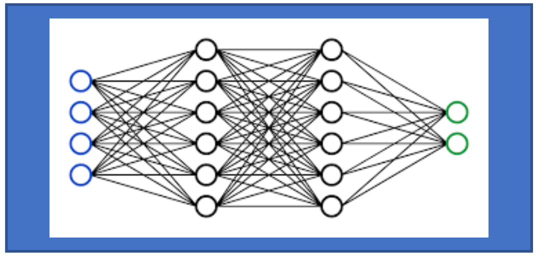
Traditional Machine Learning

What if we need more data?

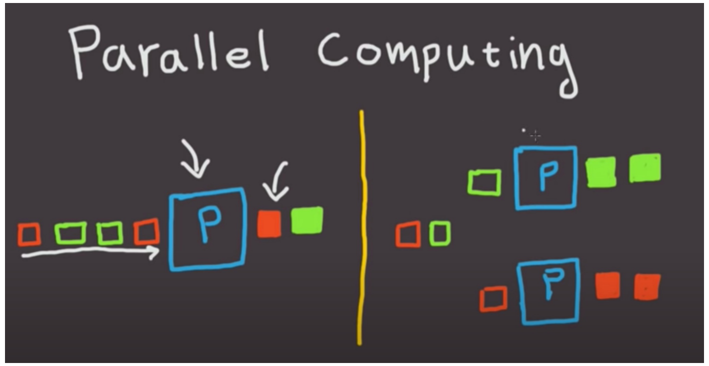


Data

Data Gathering

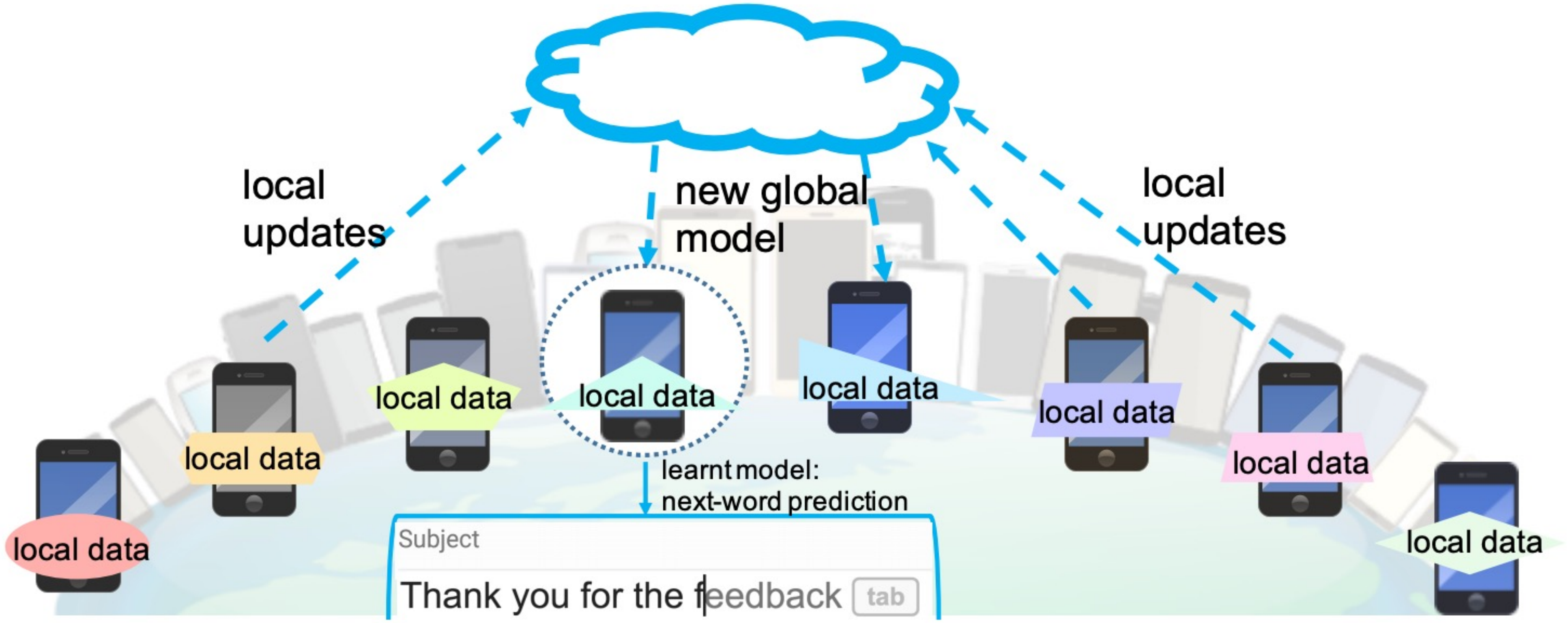


Model



Using multiple GPUs

Federated Learning: What is it?



Next word prediction on mobile.

Google: [Federated Learning: Collaborative Machine Learning without Centralized Training Data]
Federated Learning: Challenges, Methods, and Future Directions, https://arxiv.org/pdf/1908.07873.pdf

Federated Learning: Types

数据特征与标签 / 数据样本	特征1	特征2	特征3	特征4	…	特征n	标签
样本1							
样本2			参与方A的数据				
样本3							
样本4							
…			参与方B的数据				
样本m							

Horizontal FL（横着切）: same features, different samples

Federated Machine Learning: Concept and Applications, https://arxiv.org/pdf/1902.04885.pdf

Federated Learning: Types

数据特征与标签 数据样本	特征1	特征2	特征3	特征4	...	特征n	标签
样本1							
样本2							
样本3		参与方A的数据			参与方B的数据		
样本4							
...							
样本m							

Vertical FL（纵着切）: same samples, different features

Federated Machine Learning: Concept and Applications, https://arxiv.org/pdf/1902.04885.pdf

Federated Learning: Types

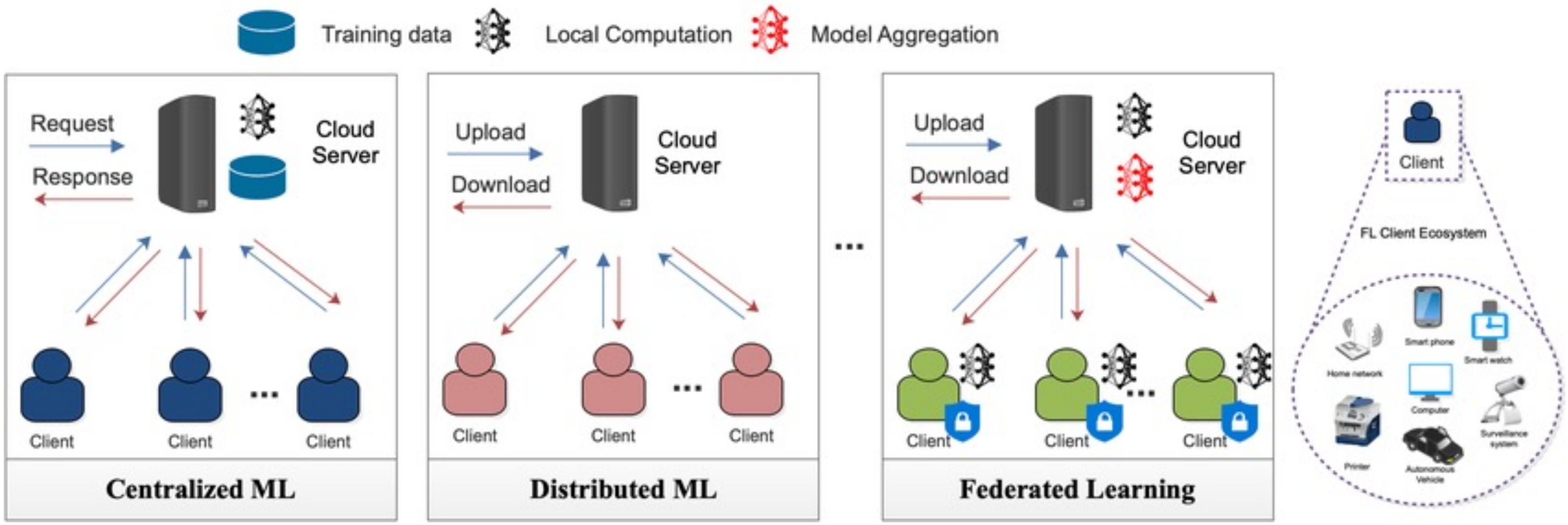
数据特征与标签 / 数据样本	特征1	特征2	特征3	特征4	…	特征n	标签
样本1							
样本2	参与方A的数据						
样本3							
样本4					参与方B的数据		
…							
样本m							

Federated Transfer Learning: different samples, different features

Federated Machine Learning: Concept and Applications, https://arxiv.org/pdf/1902.04885.pdf

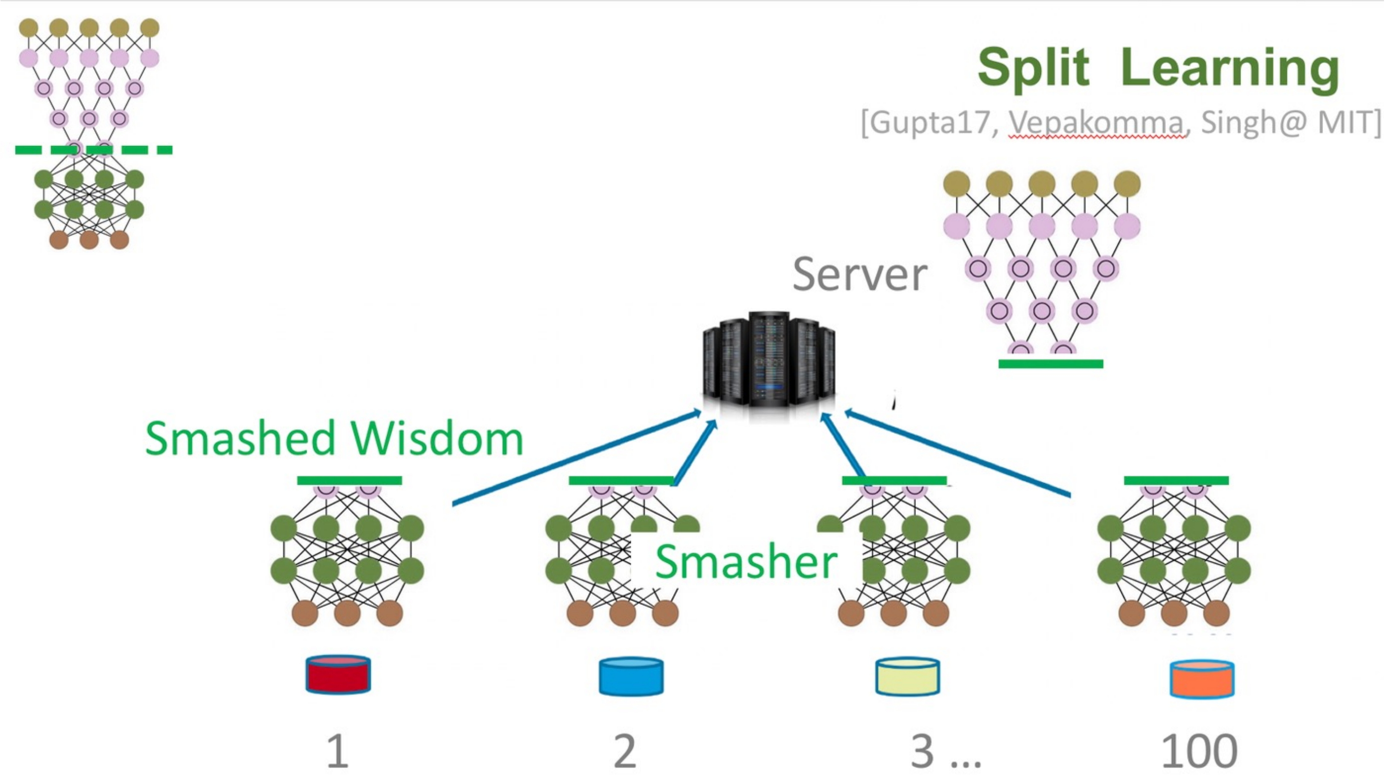
Compare Different Paradigms

Where the data goes, where the gradient goes?



Compare Different Paradigms

Split Learning vs Federated Learning



Federated Learning **Frameworks**

框架	开发者	纵向	横向	加密方法
FATE	微众银行	✓	✓	同态加密
PySyft	OPenAI	✓	✓	同态加密，秘密共享
TF Federated	Google	×	✓	秘密共享
TF Encrypted	Dropout	✓	✓	同态加密，秘密共享
CrypTen	Facebook	✓	✓	同态加密，秘密共享

HE: homomorphic encryption   **SS**: secret Sharing

Objectives and Updates in FL

Global objective $$\min_{w} F(w)\,, \quad \text{where } F(w) := \sum_{k=1}^{m} p_k F_k(w)\,.$$

Local objective: $$F_k(w) = \frac{1}{n_k} \sum_{j_k=1}^{n_k} f_{j_k}(w; x_{j_k}, y_{j_k}).$$

Local Updates: $$\mathbf{w}_{t+i+1}^{k} \longleftarrow \mathbf{w}_{t+i}^{k} - \eta_{t+i} \nabla F_k(\mathbf{w}_{t+i}^{k}, \xi_{t+i}^{k}), i = 0, 1, \cdots, E-1$$

Global Aggregation (e.g. FedAvg): $$\mathbf{w}_{t+E} \longleftarrow \sum_{k=1}^{N} p_k \, \mathbf{w}_{t+E}^{k}.$$

Federated Learning – **Major Challenges**

- Expensive Communication
- Systems Heterogeneity
- Statistical Heterogeneity
- Privacy and Security Concerns

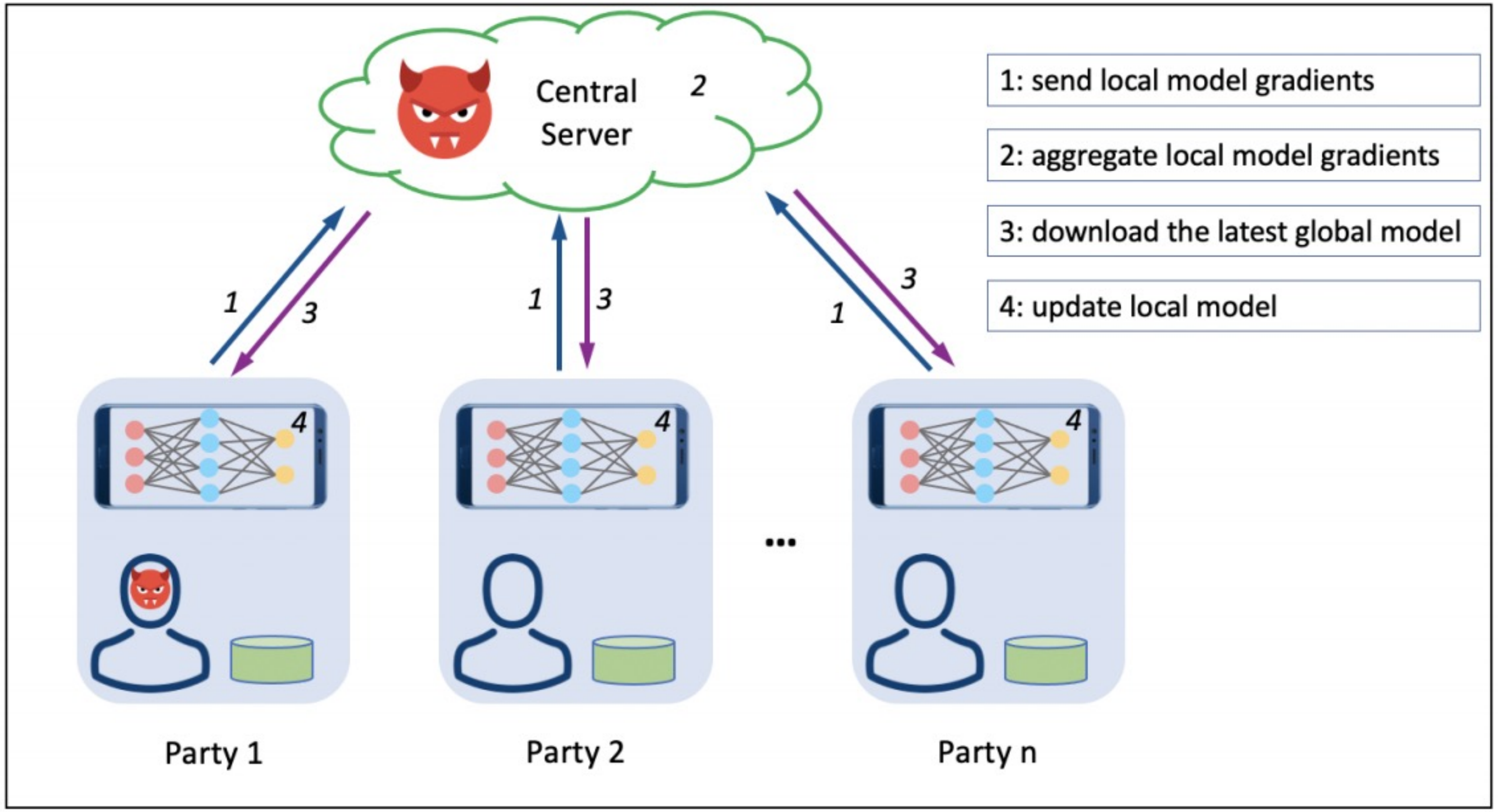
Federated Learning: Challenges, Methods, and Future Directions, https://arxiv.org/pdf/1908.07873.pdf

Federated Learning - **Horizontal**

HFL can further be divided into …?

HFL	**Number of Participants**	**Training Participation**	**Technical Capability**
H2B	small	frequent	high
H2C	large	not frequent	low

Privacy and Security Threats



Lyu et al. “Privacy and robustness in federated learning: Attacks and defenses.” *TNNLS*, 2022.

Summary of Threat Models

- **Insider vs Outsider**
  - FL server (insider)
  - FL participants (insider)
  - Eavesdroppers (outsider)
  - Service users (outsider)

- **Semi-honest vs Malicious**
  - Semi-honest setting
  - Malicious setting

- **Insider Attacks**
  - Byzantine: the worst attacker, knows everything about the system, does not obey the protocol, send **arbitrary** updates, even **collude** with each other.
  - Sybil: taking over the network by simulating many **dummy** participants, out-vote the honest users

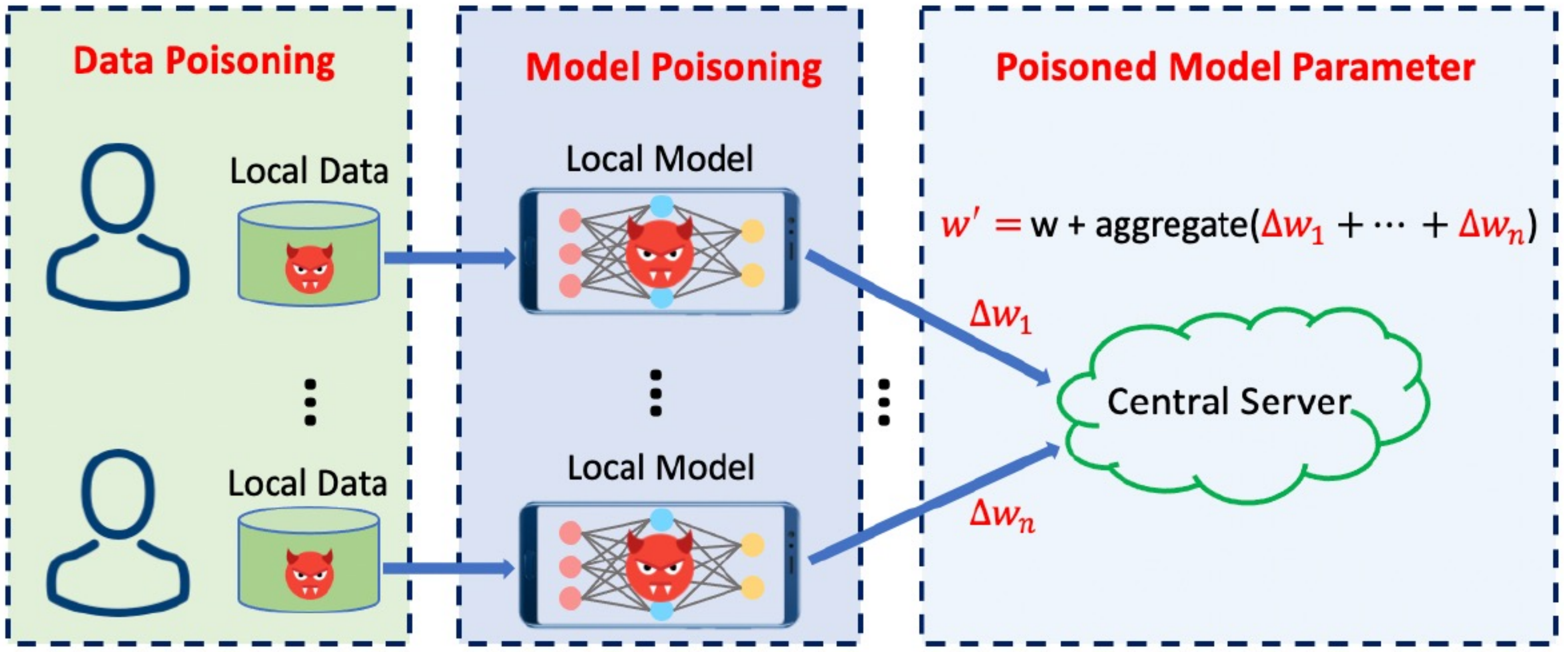
- **Training-time vs Test-time**
  - Steal private data, steal model, corrupt the model (training time)
  - Adversarial attack (test time)

Summary of Attacks

Existing attacks against server-based FL

Attack Type	Attack Target		Attacker Role		FL Scenario		Attack Complexity		
	Model	Training Data	Participant	Server	H2B	H2C	Attack Iteration		Auxiliary Knowledge Required
							One Round	Multiple Rounds	
Data Poisoning	YES	NO	YES	NO	YES	YES	YES	YES	YES
Model Poisoning	YES	NO	YES	NO	YES	NO	YES	YES	YES
Infer Class Representatives	NO	YES	YES	YES	YES	NO	NO	YES	YES
Infer Membership	NO	YES	YES	YES	YES	NO	NO	YES	YES
Infer Properties	NO	YES	YES	YES	YES	NO	NO	YES	YES
Infer Training Inputs and Labels	NO	YES	NO	YES	YES	NO	YES	YES	NO

Poisoning Attacks



Data poisoning vs model (weight) poisoning

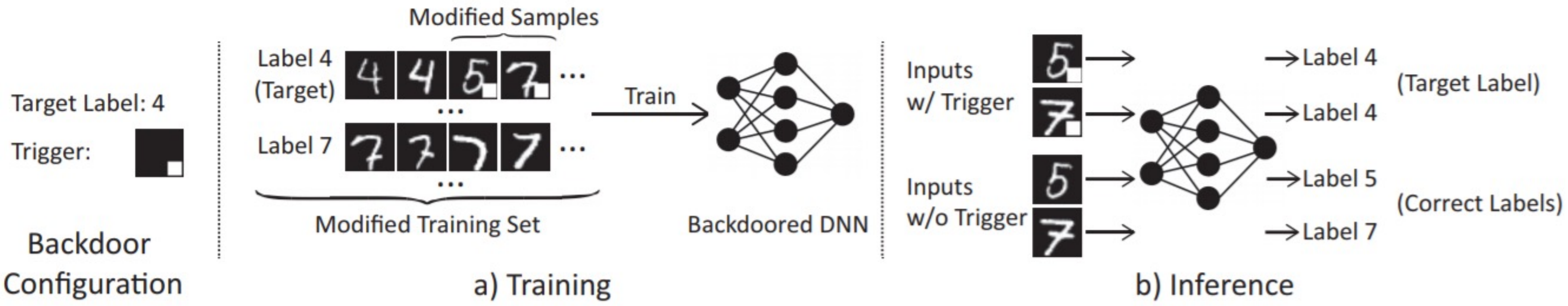
Data Poisoning Attacks in Traditional ML

❑ Dirty-label Poisoning

- Label flipping (only change **labels**)
- Dirty-label backdoor (change **inputs** and **labels**)

❑ Clean-label Poisoning

- Clean-label backdoor (only change **inputs**)



Data Poisoning Attacks in Traditional ML

BadNets

Trojan

Blend

CL

SIG

Refool

A simple pattern can make the model to memorize

FL Poisoning Attacks – Model Poisoning

Main characteristics:

- Change local model weights
- Mostly Byzantine attack (attacker can do anything to the weights)
- Can attack Byzantine-robust aggregation mechanisms such as **Krum** and **coordinate-wise** median instead of weighted averaging

Definition III.1. *[Byzantine Model Poisoning] [13], [14] In the t^{th} round, an honest participant uploads $\Delta \boldsymbol{w}_i^{(t)} := \nabla F_i(\boldsymbol{w}_i^{(t)})$ while a dishonest participant/adversary can upload arbitrary values.*

$$\Delta \boldsymbol{w}_i^{(t)} = \begin{cases} *, & \text{if } i\text{-th participant is Byzantine,} \\ \nabla F_i(\boldsymbol{w}_i^{(t)}), & \text{otherwise,} \end{cases}$$

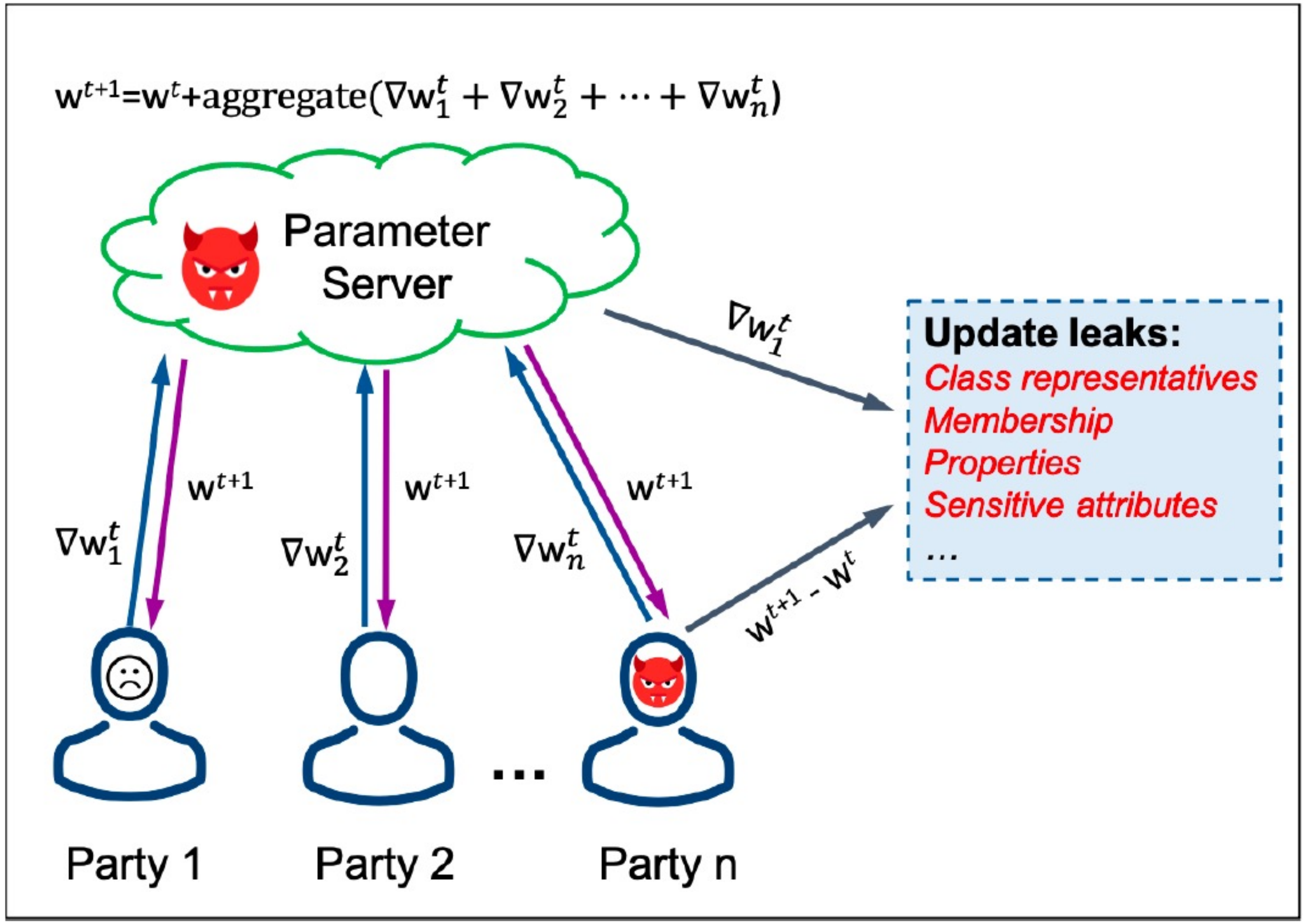
Krum:

$$x_{t+1} = x_t - \gamma_t \cdot \mathrm{K\scriptsize{R}}(V_1^t, \ldots, V_n^t)$$

$$score\ s(i) = \sum_{i \to j} \|V_i - V_j\|^2$$

$$\mathrm{K\scriptsize{R}}(V_1, \ldots, V_n) = V_{i_*}$$

Privacy Attacks



$$\boxed{w^{t+1} - w^t}$$

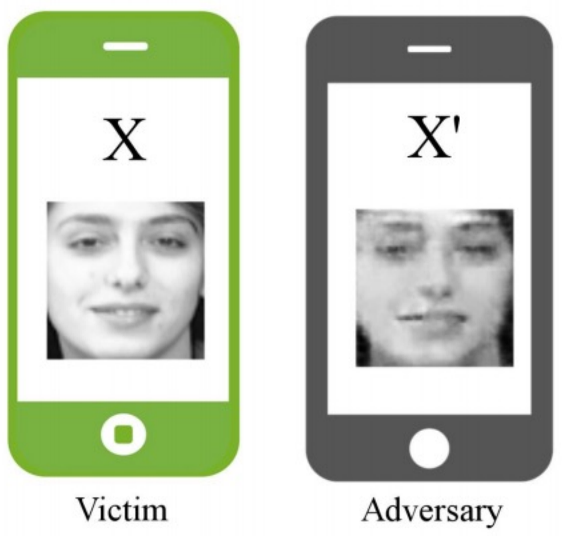
For every communication round, local clients have the chance to reverse engineer others' gradients.

From the reversed gradients, reverse engineer:

- Representations
- Membership
- Properties
- Sensitive attributes
- ***In VFL: features***

Privacy Attacks – Inference Attacks

Inference **class representations** using GANs



Reconstruct Alice’s face image

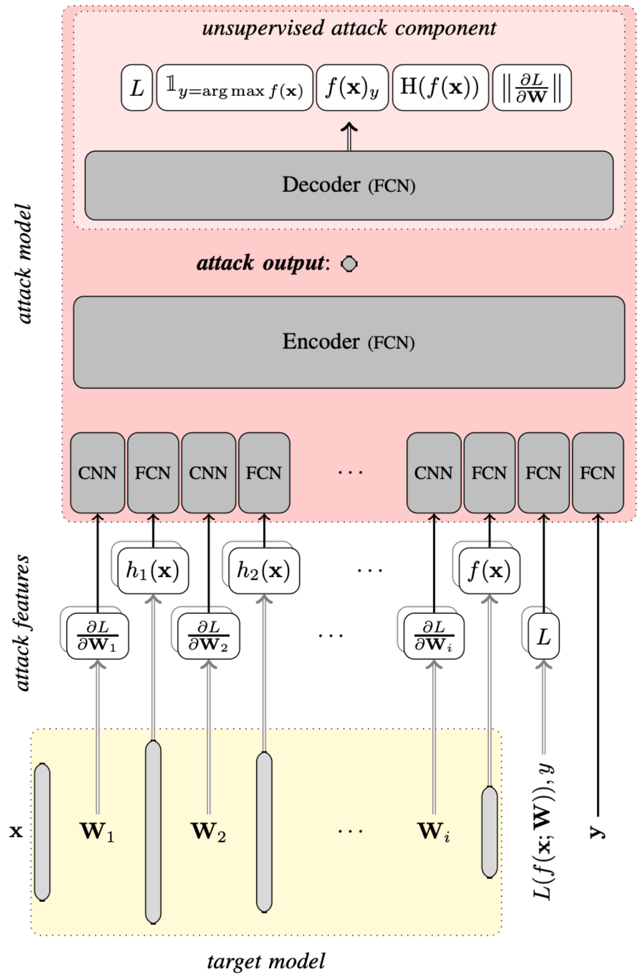
CIFAR-10 horse class

[*Deep models under the GAN: information leakage from collaborative deep learning, CCS 2017*]

Privacy Attacks – Inference Attacks

Inference **membership**:

- **Passive attacks**: observe and inference.
- **Active attacks**: influence the target model in order to extract more information.



Weakness of FL: FL creates an environment for (almost) white-box attacks

Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning, S&P, 2019

Privacy Attacks – Inference Attacks

Other inference attacks:

inferring properties, training data, labels ...

- **Deep Leakage from Gradient** (DLG)
- **Improved Deep Leakage from Gradient** (iDLG)
- ...

Deep Leakage from Gradients

Ligeng Zhu Zhijian Liu Song Han
Massachusetts Institute of Technology
{ligeng, zhijian, songhan}@mit.edu

Abstract

Exchanging gradients is a widely used method in modern multi-node machine learning system (*e.g.*, distributed training, collaborative learning). For a long time, people believed that gradients are safe to share: *i.e.*, the training data will not be leaked by gradients exchange. However, we show that it is possible to obtain the private training data from the publicly shared gradients. We name this leakage as *Deep Leakage from Gradient* and empirically validate the effectiveness on both computer vision and natural language processing tasks. Experimental results show that our attack is much stronger than previous approaches: the recovery is *pixel-wise* accurate for images and *token-wise* matching for texts. Thereby we want to raise people's awareness to rethink the gradient's safety. We also discuss several possible strategies to prevent such deep leakage. Without changes on training setting, the most effective defense method is gradient pruning.

iDLG: Improved Deep Leakage from Gradients

Bo Zhao, Konda Reddy Mopuri, Hakan Bilen
School of Informatics
The University of Edinburgh, United Kingdom
{bo.zhao, kmopuri, hbilen}@ed.ac.uk

Abstract

It is widely believed that sharing gradients will not leak private training data in distributed learning systems such as Collaborative Learning and Federated Learning, etc. Recently, Zhu *et al.* [1] presented an approach which shows the possibility to obtain private training data from the publicly shared gradients. In their Deep Leakage from Gradient (DLG) method, they synthesize the dummy data and corresponding labels with the supervision of shared gradients. However, DLG has difficulty in convergence and discovering the ground-truth labels consistently. In this paper, we find that sharing gradients definitely leaks the ground-truth labels. We propose a simple but reliable approach to extract accurate data from the gradients. Particularly, our approach can certainly extract the ground-truth labels as opposed to DLG, hence we name it Improved DLG (iDLG). Our approach is valid for any differentiable model trained with cross-entropy loss over one-hot labels. We mathematically illustrate how our method can extract ground-truth labels from the gradients and empirically demonstrate the advantages over DLG.

Inverting Gradients - How easy is it to break privacy in federated learning?

Jonas Geiping* Hartmut Bauermeister * Hannah Dröge *

Michael Moeller

Dep. of Electrical Engineering and Computer Science
University of Siegen
{jonas.geiping, hartmut.bauermeister, hannah.droege, michael.moeller }@uni-siegen.de

Abstract

The idea of federated learning is to collaboratively train a neural network on a server. Each user receives the current weights of the network and in turns sends parameter updates (gradients) based on local data. This protocol has been designed not only to train neural networks data-efficiently, but also to provide privacy benefits for users, as their input data remains on device and only parameter gradients are shared. But how secure is sharing parameter gradients? Previous attacks have provided a false sense of security, by succeeding only in contrived settings - even for a single image. However, by exploiting a magnitude-invariant loss along with optimization strategies based on adversarial attacks, we show that is is actually possible to faithfully reconstruct images at high resolution from the knowledge of their parameter gradients, and demonstrate that such a break of privacy is possible even for trained deep networks. We analyze the effects of architecture as well as parameters on the difficulty of reconstructing an input image and prove that any input to a fully connected layer can be reconstructed analytically independent of the remaining architecture. Finally we discuss settings encountered in practice and show that even aggregating gradients over several iterations or several images does not guarantee the user's privacy in federated learning applications.

Defenses – Privacy Defense

1. **Homomorphic Encryption**:
   - RSA
   - El Gamal
   - Paillier
   - …

Framework	Developer	Vertical	Horizontal	Encryption
FATE	WeBank	✓	✓	HE
PySyft	OpenAI	✓	✓	HE, SS
TF Federated	Google	✗	✓	SS
TF Encrypted	Dropout	✓	✓	HE, SS
CrypTen	Facebook	✓	✓	HE, SS

Homomorphic properties:

- Allows computation directly on encrypted data（“可算不可见”）
- Needs to be designed for each algorithm

$$E_{pk}(m_1 + m_2) = c_1 \oplus c_2$$

$$E_{pk}(a \cdot m_1) = a \otimes c_1$$

A side note: **attacking encrypted FL** is challenging but still possible!

Defenses – Privacy Defense

2. Secure Multiparty Computation (SMC, Yao sharing):

- SecureML (data-independent offline phase + fast online phase)

Offline multiplication triplets, truncate, sharing

Characteristics:

- High level privacy
- High computation and communication cost

Yao's Millionaires' problem

[*Protocols for Secure Computations, Andrew Chi-Chih Yao, 1982, UC Berkeley*]()

Defenses – Privacy Defense

2. Differential Privacy (DP):

Types of DP:

- Local DP
- Centralized DP
- Distributed DP

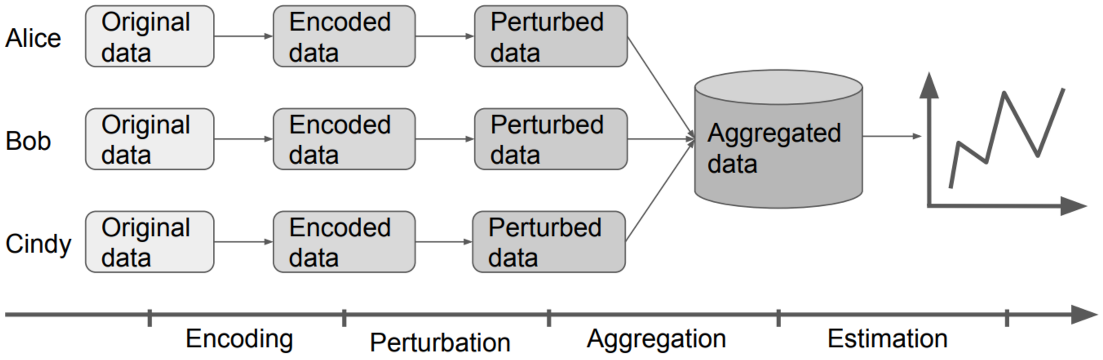
Definition V.1. (ϵ, δ)*-differential privacy [98]. For scalars* $\epsilon > 0$ *and* $0 \leq \delta < 1$*, mechanism* $\mathcal{M}$ *is said to preserve (approximate)* (ϵ, δ)*-differential privacy if for all adjacent datasets* $D, D' \in \mathcal{D}^n$ *and measurable* $S \in \mathrm{range}(\mathcal{M})$,

$$\Pr\{\mathcal{M}(D) \in S\} \quad \leq \quad \exp(\epsilon) \cdot \Pr\{\mathcal{M}(D') \in S\} + \delta \ .$$

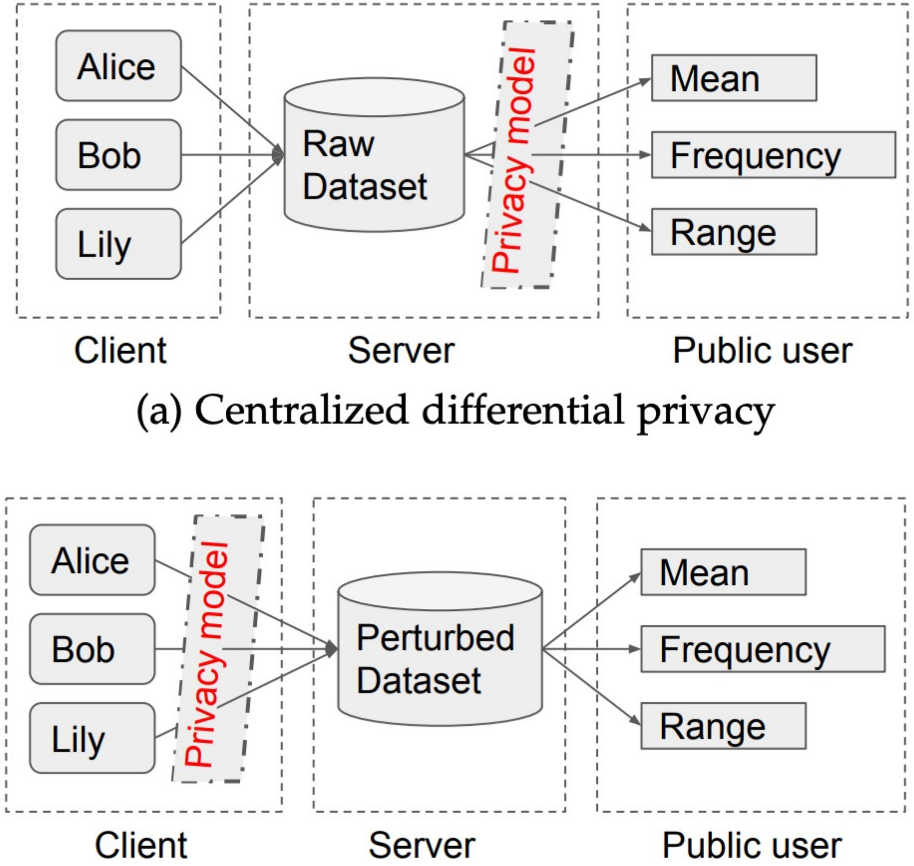
DP type	Trusted aggregator?	Who should add noise?	Privacy Guarantee	Error Bound
CDP [45], [7]	Yes	aggregator	aggregated value	$O(\frac{1}{\epsilon})$
LDP [18], [109]	No	user	locally released value	$O(\frac{\sqrt{n}}{\epsilon})$
DDP [21], [110]	No	user	aggregated value	$O(\frac{1}{\epsilon})$

Defenses – Privacy Defense

2. Differential Privacy (DP):



Data flow of statistics under LDP



(a) Centralized differential privacy

(b) Local differential privacy

Defenses – Privacy Defense

2. Differential Privacy (DP):

Technique	Encoding	Perturbation	Variance	Communication
GRR [17]	$t = v$	$Pr[\hat{t} = v] = \begin{cases} \frac{e^\epsilon}{e^\epsilon+d-1}, & if\ t = v \\ \frac{1}{e^\epsilon+d-1}, & if\ t \neq v \end{cases}$	$O\left(\frac{d-2+e^\epsilon}{(e^\epsilon-1)^2}\right)$	$\log d$
OUE [20]	$t = [0, \cdots, 1, \cdots, 0]$, where $t[v] = 1$	$Pr[\hat{t}[i] = 1] = \begin{cases} \frac{1}{2}, & if\ t[i] = 1 \\ \frac{1}{e^\epsilon+1}, & if\ t[i] = 0 \end{cases}$	$O\left(\frac{4e^\epsilon}{(e^\epsilon-1)^2}\right)$	d
RAPPOR [8]	$r =< \mathcal{H}, t >$; $\mathcal{H} \in \mathbb{H}$; $t = [0, \cdots, 1, \cdots]$ $where\ t[i] = \begin{cases} 1, if\ \mathcal{H}(v) = 1, \\ 0,\ otherwise \end{cases}$	$Pr[\hat{t}[i] = 1] = \begin{cases} 1 - \frac{1}{2}f, & if\ t[i] = 1 \\ \frac{1}{2}f, & if\ t[i] = 0' \end{cases}$ where $f = \frac{2}{e^{\epsilon/2}+1}$	$O\left(\frac{e^{\epsilon/2}}{(e^{\epsilon/2}-1)^2}\right)$	$\log m$
OLH [20]	$r =< \mathcal{H}, t >$; $\mathcal{H} \in \mathbb{H}$; $t = \mathcal{H}(v)$	$Pr[\hat{t} = \mathcal{H}(v)] = \begin{cases} \frac{e^\epsilon}{e^\epsilon+g-1}, & if\ t = \mathcal{H}(v) \\ \frac{1}{e^\epsilon+g-1}, & if\ t \neq \mathcal{H}(v)' \end{cases}$ where $g = e^\epsilon + 1$	$O\left(\frac{4e^\epsilon}{(e^\epsilon-1)^2}\right)$	$\log n$
JLRR [21]	$\Phi \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^{m\times d}$; $r =< i, t >$; $i \in [m]$; $t = \Phi[i, v]$	$\hat{t} = \begin{cases} c_\epsilon dt, & w.p.\ \frac{e^\epsilon}{e^\epsilon+1} \\ -c_\epsilon dt, & w.p.\ \frac{1}{e^\epsilon+1} \end{cases}$, where $c_\epsilon = \frac{e^\epsilon+1}{e^\epsilon-1}$	$O\left(\frac{4e^\epsilon}{(e^\epsilon-1)^2}\right)$	$\log m$
HRR [22, 31]	$\Phi : 2^d \times 2^d$ Hadamard Matrix, where $\Phi[i, j] = 2^{-d/2}(-1)^{<i,j>}$; $r =< i, t >$; $i \in [2^d]$; $t = \Phi[i, v]$	$Pr[\hat{t} = 1] = \begin{cases} \frac{e^\epsilon}{e^\epsilon+1}, & if\ t = 1 \\ \frac{1}{e^\epsilon+1}, & if\ t = -1 \end{cases}$	$O\left(\frac{4e^\epsilon}{(e^\epsilon-1)^2}\right)$	$O(1)$

Types of frequency estimation

Defenses – Privacy Defense

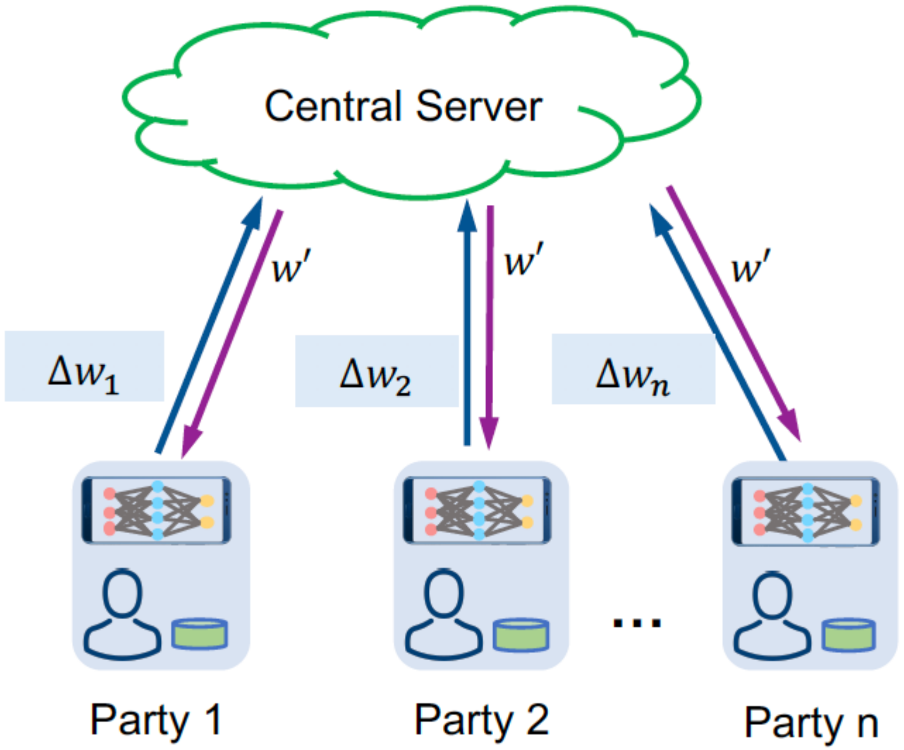
2. Differential Privacy (DP):

Company	Deployment	Purpose/Functionality	Techniques	Population	Parameters	Limitations	Open source
Google	Chrome Browser (2014)	Collect up-to-date statistics about the activity of their users and their client-side software	2-level RR memoization Bloom filter	14 million	$\epsilon = 0.5343$ $h^1 = 2$ $k^2 = 128$	Not suitable for data with frequent changes	Yes
Apple	macO iOS10 (2016)	Estimate the frequencies of elements	RR CMS HT[3]	Hundreds of millions	$\epsilon = 2 \sim 8$ $m^4 = 256 \sim 32768$ $h = 1024 \sim 65536$	The overall privacy cost for each device is unbounded	No
Microsoft	Windows 10 (2017)	Repeated collection of counter data mean estimation histogram estimation	1BitMean dBitFlip α-point rounding memoization	millions	$\epsilon = 1$	Not suitable for data with significant changes	No
SAP	HANA 2.0 SPS03 (2018)	Count Sum Average	LM[5]	–	Leave it up to the data consumer	Only support numerical value The added noise is unbounded	No

[1] h number of hash functions
[2] k Bloom filter size
[4] m CMS size
[3] HT Hadamard transform
[5] LM Laplace mechanism

Real-world applications.

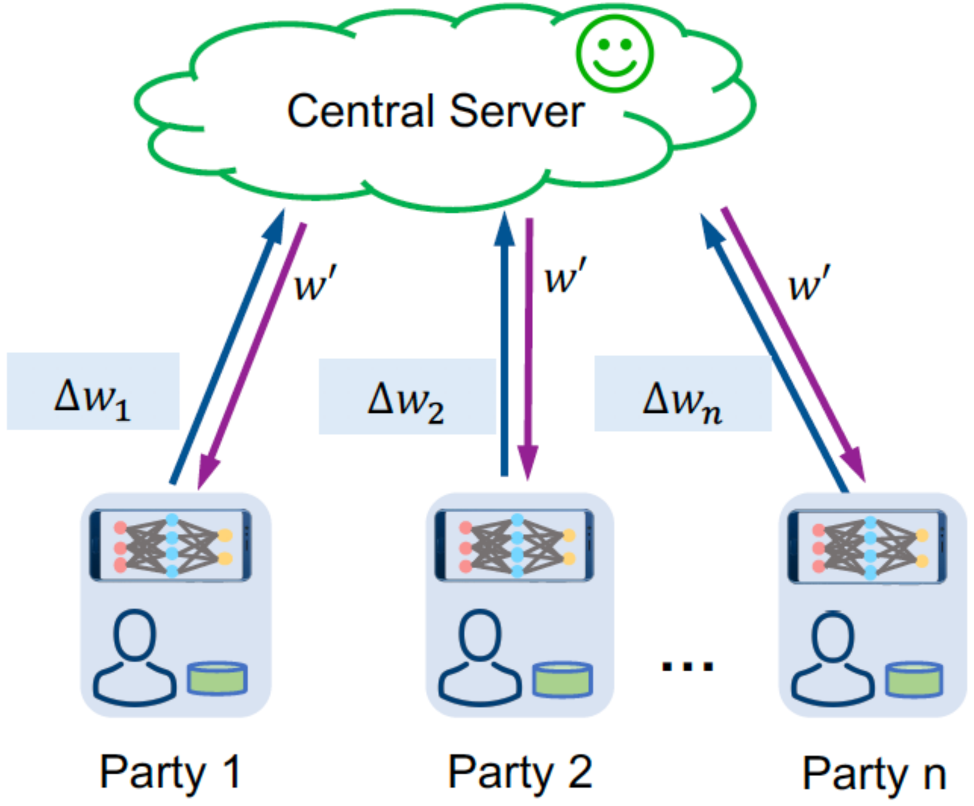
$$w' = w + \text{aggregate}(\Delta w_1 + \Delta w_2 + \cdots + \Delta w_n)$$



(a) FL without privacy.

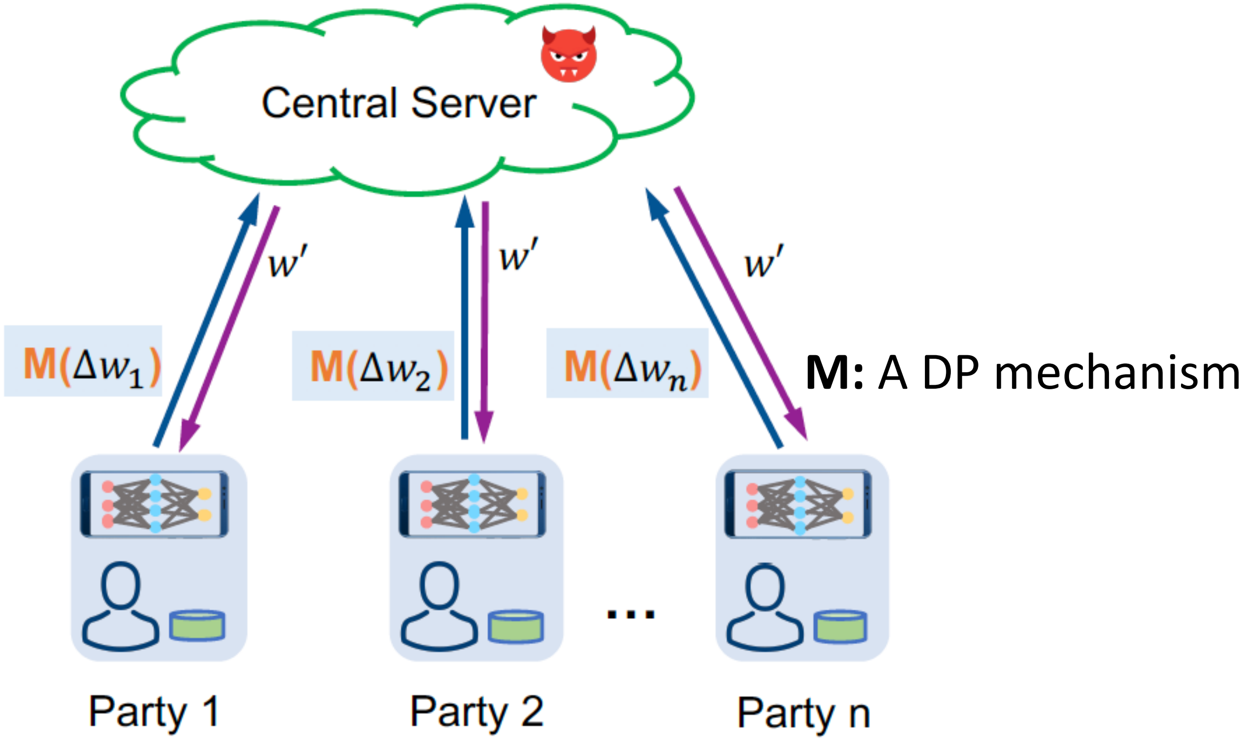
$w' = w + \mathbf{M}(\text{aggregate}(\Delta w_1 + \Delta w_2 + \cdots + \Delta w_n))$

M: A DP mechanism



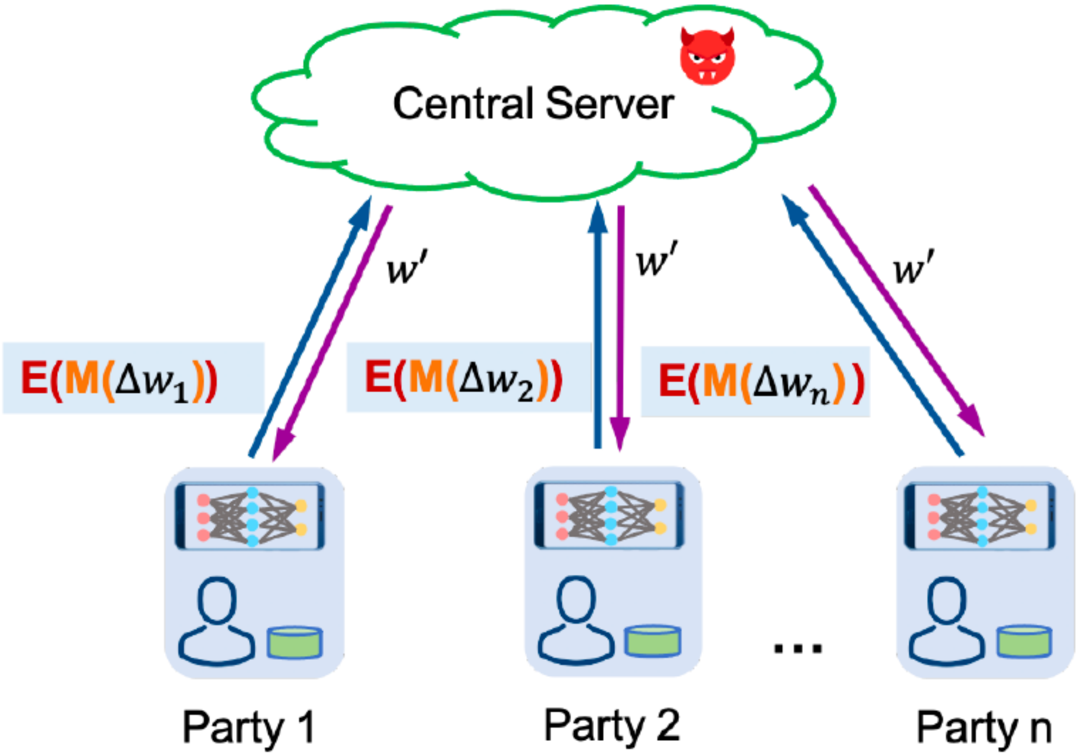
(b) Centralized DP: FL with a trusted server.

$$w' = w + \text{aggregate}(M(\Delta w_1) + M(\Delta w_2) + \cdots + M(\Delta w_n))$$



(c) Local DP: FL without a trusted server.

$$w' = w + \mathbf{D}(\text{aggregate}(\mathbf{E}(\mathbf{M}(\Delta w_1)) + \mathrm{E}(\mathbf{M}(\Delta w_2)) + \cdots + \mathbf{E}(\mathbf{M}(\Delta w_n))))$$



(d) Distributed DP with SMC: FL without a trusted server.

M: A DP mechanism
E: encryption
D: decryption

Defenses – Byzantine Defense

Algorithm: **Krum** (for Byzantine robustness)

Setting: **n** participants, **f** are Byzantine, with $\boldsymbol{n \geq 2f+3}$

At communication round **t,**

server receives $\{\boldsymbol{\delta_1^t}, \boldsymbol{\delta_2^t}, \ldots, \boldsymbol{\delta_n^t}\}$

for each $\boldsymbol{\delta_i^t}$**:**

select the closest (L2 distance) **n-f-2** into set $\boldsymbol{C_i}$

compute $\boldsymbol{score}(\boldsymbol{\delta_i^t}) = \sum_{\boldsymbol{\delta} \in \boldsymbol{C_i}} (\boldsymbol{\delta_i^t} - \boldsymbol{\delta})$

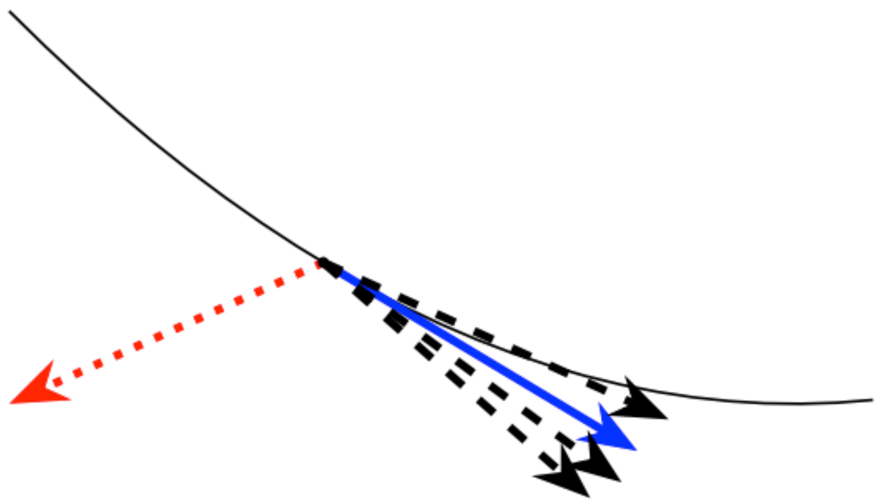
$$\boldsymbol{\delta_{krum}} = \boldsymbol{\delta}^* = \arg\min_{\boldsymbol{\delta}} \{\boldsymbol{score}(\boldsymbol{\delta_1^t}), \ldots, \boldsymbol{score}(\boldsymbol{\delta_n^t})\}$$

update global parameter: $\boldsymbol{w^{t+1}} = \boldsymbol{w^t} + \boldsymbol{\delta_{krum}}$

Blanchard et al. "Machine learning with adversaries: Byzantine tolerant gradient descent." NeurIPS, 2017.

Defenses – Byzantine Defense

Algorithm: **Krum** (for Byzantine robustness)



红色：攻击梯度

蓝色：真实梯度

黑色：本地梯度

黑色曲线：损失函数

Blanchard et al. “Machine learning with adversaries: Byzantine tolerant gradient descent.” NeurIPS, 2017.

Defenses – Byzantine Defense

More robust aggregation methods:

- Multi-Krum = **Krum** + Averaging

  = Krum robustness + increased convergence speed
- coordinate-wise median, coordinate-wise trimmed mean

  median is not good for convergence
- Bulyan = **Krum + trimmed median**
- Median and geometric-median
- (Robust Federated Aggregation) RFA: approximate geometric median (not robust to Byzantine attacks)

Defenses – Byzantine Defense

Model poisoning attack can break Krum and coordinate-wise median

$$\underset{\boldsymbol{\delta}_m^t}{\operatorname{argmin}} \lambda L(\{\mathbf{x}_i, \tau_i\}_{i=1}^r, \hat{\mathbf{w}}_G^t) + L(\mathcal{D}_m, \mathbf{w}_m^t) + \rho \|\boldsymbol{\delta}_m^t - \bar{\boldsymbol{\delta}}_{\text{ben}}^{t-1}\|_2$$

τ_i: adversarial target class

r: number of poisoned samples

D_m: clean data

$\widehat{w_G^t}$: estimation of the global parameters

$$\bar{\boldsymbol{\delta}}_{\text{ben}}^{t-1} = \sum_{i \in [k] \setminus m} \alpha_i \boldsymbol{\delta}_i^{t-1}$$

Reversed gradients from the last round.

[Analyzing federated learning through an adversarial lens, ICML 2019.](#)

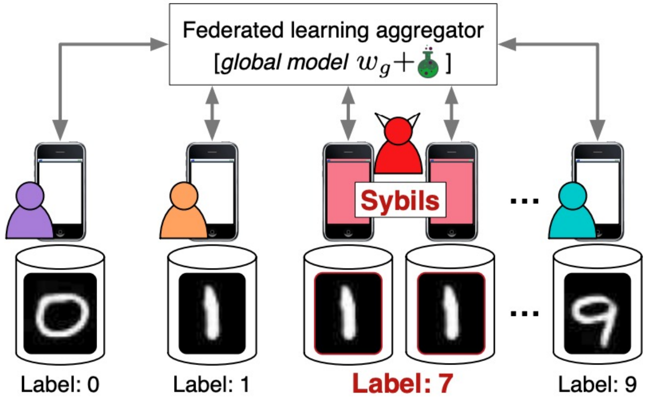
Defenses – Sybil Defense

From traditional ML: **Reject on Negative Influence (RONI)**

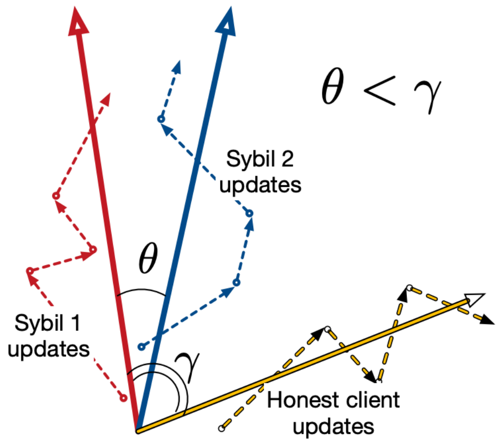
- With a clean validation dataset
- It requires uniform distribution in non-IID setting, not good.

FoolsGold:
Sybil share the same objective, drifts away from the original objective
Core idea: cosine similarity

$$cs_{ij} = cosine\_similarity(\sum_{t=1}^{T} \Delta_{i,t}, \sum_{t=1}^{T} \Delta_{j,t})$$



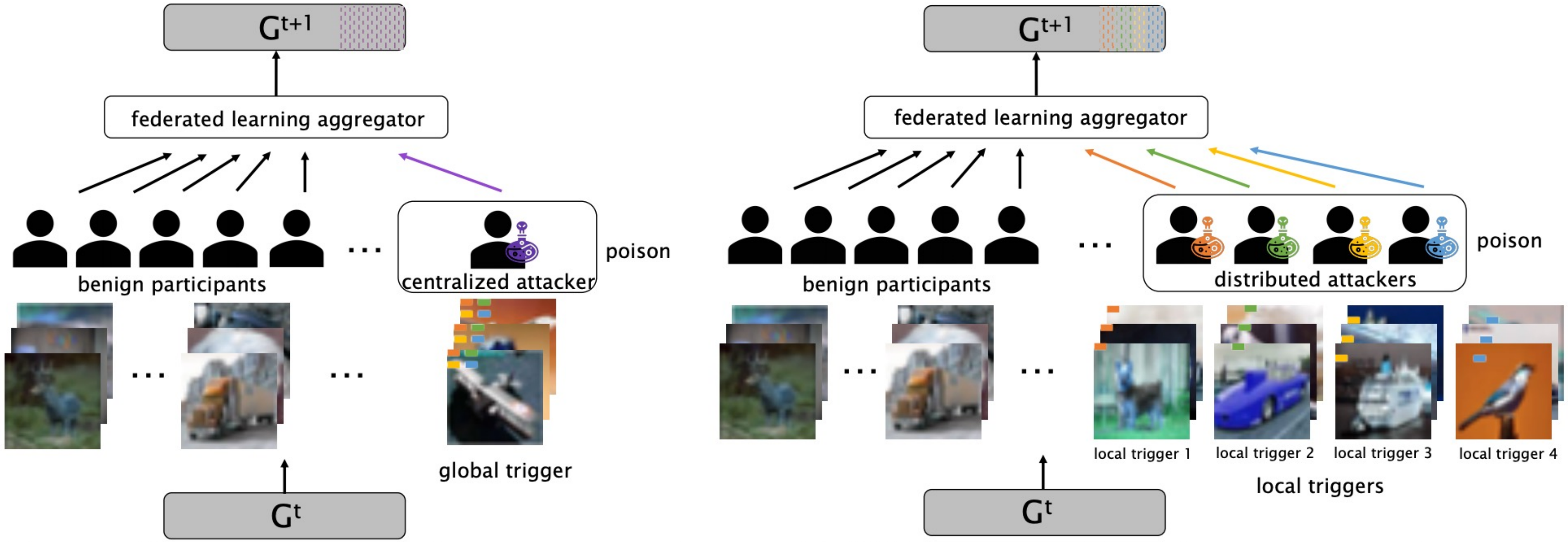
(b) Federated learning with sybil-based label-flipping poisoning



[*FoolsGold: Mitigating Sybils in Federated Learning Poisoning, https://arxiv.org/abs/1808.04866*](https://arxiv.org/abs/1808.04866)

Defenses – Sybil Defense

Distributed backdoor attack (**DBA**) can bypass both RFA and FoolsGold**.**



(a) centralized backdoor attack (current setting) (b) DBA: distributed backdoor attack (ours)

[*DBA: Distributed Backdoor Attacks against Federated Learning, ICLR 2020.*]()

Defenses – Summary

Defense against Federated Learning Poisoning. **n**: number of participants.

Poisoning Defense	Technique	IID Data	Non-IID Data	Breaking Point	Data Poisoning	Model Poisoning
RONI [133], [28]	Error rate	✓	×	NA	✓	×
Auror [127]	Clustering	✓	×	NA	✓	×
Krum [13]	Euclidean distance	✓	×	$(n-2)/2n$	✓	×
Coordinate-wise Median [14]	Coordinate-wise median	✓	×	1/2	✓	×
Bulyan [128]	Krum + trimmed median	✓	×	(n-3)/4n	✓	×
FoolsGold [28]	Contribution similarity	✓	✓	NA	✓	×
RFA [48]	Geometric median	✓	×	NA	✓	✓

Remaining Challenges and Future Research

- **Curse of dimensionality**
  - Larger models are more vulnerable
  - Sharing weights/gradients may not be a good idea

- **Weaknesses of current attacks**
  - GAN attack assumes the class of data is from one single participant
  - DLG/iDLG work with second-order gradient method (expensive) and small minibatch-gradients (B=8)

- **Vulnerability to free riders**:
  pretend to have data but not.

Remaining Challenges and Future Research

❑ Weakness of Current Privacy-preserving Techniques

- Secure aggregation is more vulnerable to poisoning attacks since individual updates cannot be checked
- Adversarial training (IID or non-IID, local or global, training or distillation)?
- Sample-level DP does not stop attribute/property/statistical inference attacks
- DP hurts accuracy, efficiency (is millions of participant-level DP possible?)

Remaining Challenges and Future Research

- **Defense efficiency**
  - Expensive to check each participant (detection)
  - When and how to deploy a defense?

- **Hard to achieve all objective of private and secure**
  - Efficiency
  - Privacy
  - Robustness
  - Generalization
  - Collaborative fairness

Remaining Challenges and Future Research

- **FL: optimization and convergence**
  - GD -> SGD -> Parallel SGD -> Local SGD

Table 1: **Summary of results on the synchronization rounds R required for linear speedup in M.** All bounds hide multiplicative polylog factors and variables other than M and T for ease of presentation. Notation: M: number of workers; T: parallel runtime.

		Synchronization Required for Linear Speedup		
Assumption	Algorithm	Strongly Convex	General Convex	Reference
Assumption 1	FedAvg	$T^{\frac{1}{2}}M^{\frac{1}{2}}$	–	(Stich, 2019a)
		$T^{\frac{1}{3}}M^{\frac{1}{3}}$	–	(Haddadpour et al., 2019b)
		M	$T^{\frac{1}{2}}M^{\frac{3}{2}}$	(Stich and Karimireddy, 2019)
		M	$T^{\frac{1}{2}}M^{\frac{3}{2}}$	(Khaled et al., 2020)
	FedAc	$\boldsymbol{M^{\frac{1}{3}}}$	$\min\{\boldsymbol{T^{\frac{1}{4}}M^{\frac{3}{4}}}, \boldsymbol{T^{\frac{1}{3}}M^{\frac{2}{3}}}\}$	**Theorems 3.1, E.1 and E.2**
Assumption 2	FedAvg	$\max\{\boldsymbol{T^{-\frac{1}{2}}M^{\frac{1}{2}}}, \boldsymbol{1}\}$	$T^{\frac{1}{2}}M^{\frac{3}{2}}$	**Theorems 3.4 and E.4**
	FedAc	$\max\{\boldsymbol{T^{-\frac{1}{6}}M^{\frac{1}{6}}}, \boldsymbol{1}\}$	$\max\{\boldsymbol{T^{\frac{1}{4}}M^{\frac{1}{4}}}, \boldsymbol{T^{\frac{1}{6}}M^{\frac{1}{2}}}\}$	**Theorems 3.3 and E.3**

Federated Accelerated Stochastic Gradient Descent Tighter Theory for Local SGD on Identical and Heterogeneous Data, AISTAS, 2020; On the convergence of FedAvg on non-IID data, ICLR 2020

谢谢！