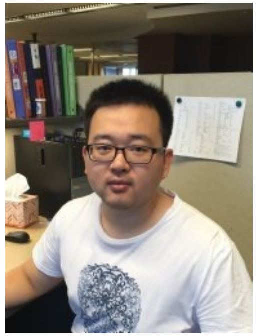
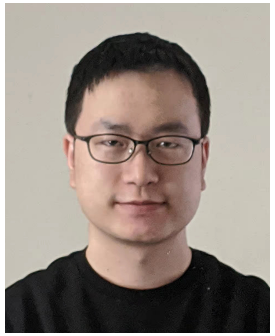
Adversarial Robustness through Bias Variance Decomposition: A New Perspective for Federated Learning

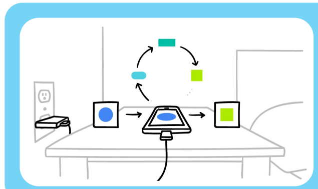
Yao Zhou*
Instacart & UIUC
yaozhou3@illinois.edu

Jun Wu*
UIUC
junwu3@illinois.edu

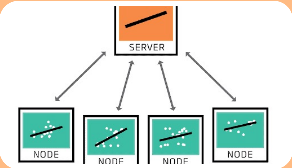
Haixun Wang
Instacart
haixun@gmail.com

Jingrui He
UIUC
jingrui@illinois.edu

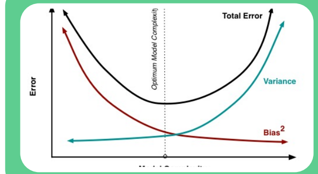
Roadmap



Background
- Federated learning
- Vulnerability to adversarial perturbation



Problem Definition
- Adversarially robust federated learning
- Unique challenges



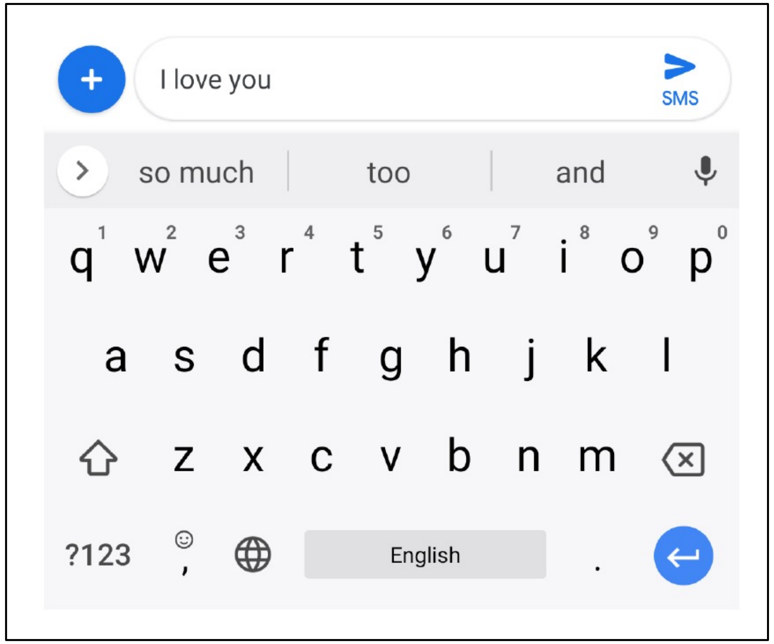
Methodology
- Bias-Variance analysis
- Generic framework

Experiments
- Effectiveness
- Efficiency

Conclusion
- Problem, algorithm, evaluation

What is Federated Learning?

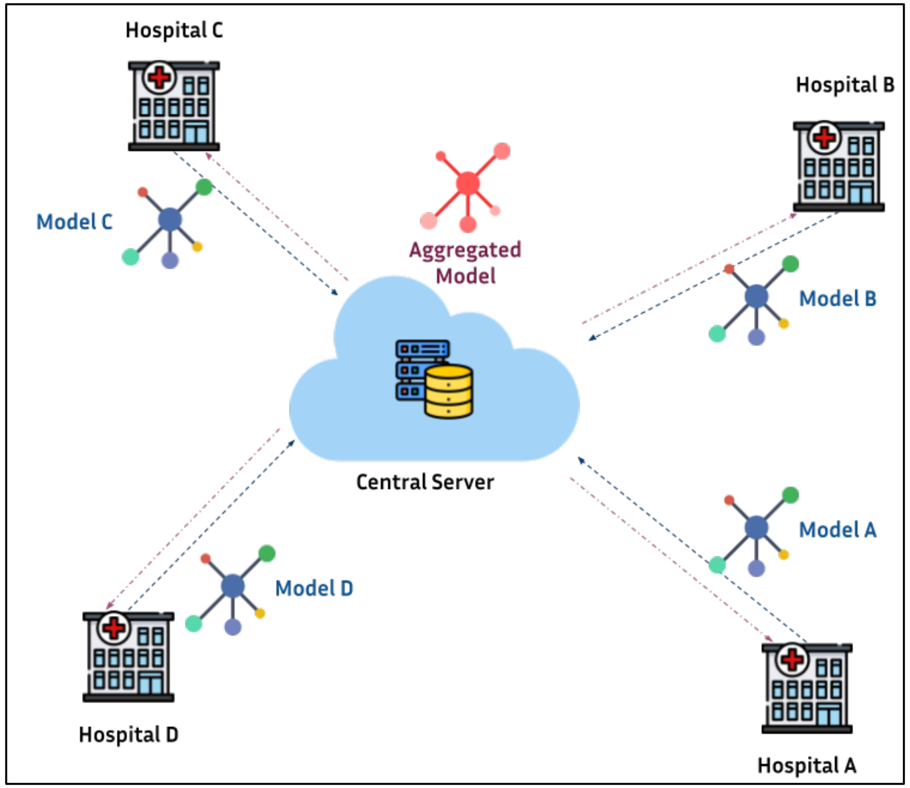
- Definition:
  - Multiple clients **collaborate in solving a machine learning problem**, under the coordination of a central server or service provider.
  - Each client's **raw data is stored locally** and not exchanged.
- Examples:
  - Mobile keyboard prediction for different users



Kairouz, Peter, et al. "**Advances and open problems in federated learning**." *Foundations and Trends® in Machine Learning*. 2021.
Hard, Andrew, et al. "**Federated learning for mobile keyboard prediction**." *arXiv preprint arXiv:1811.03604* (2018).
Xu, Jie, et al. "**Federated learning for healthcare informatics**." *Journal of Healthcare Informatics Research. 2021*

What is Federated Learning?

❑ Definition:
- Multiple clients **collaborate in solving a machine learning problem**, under the coordination of a central server or service provider.
- Each client's **raw data is stored locally** and not exchanged.

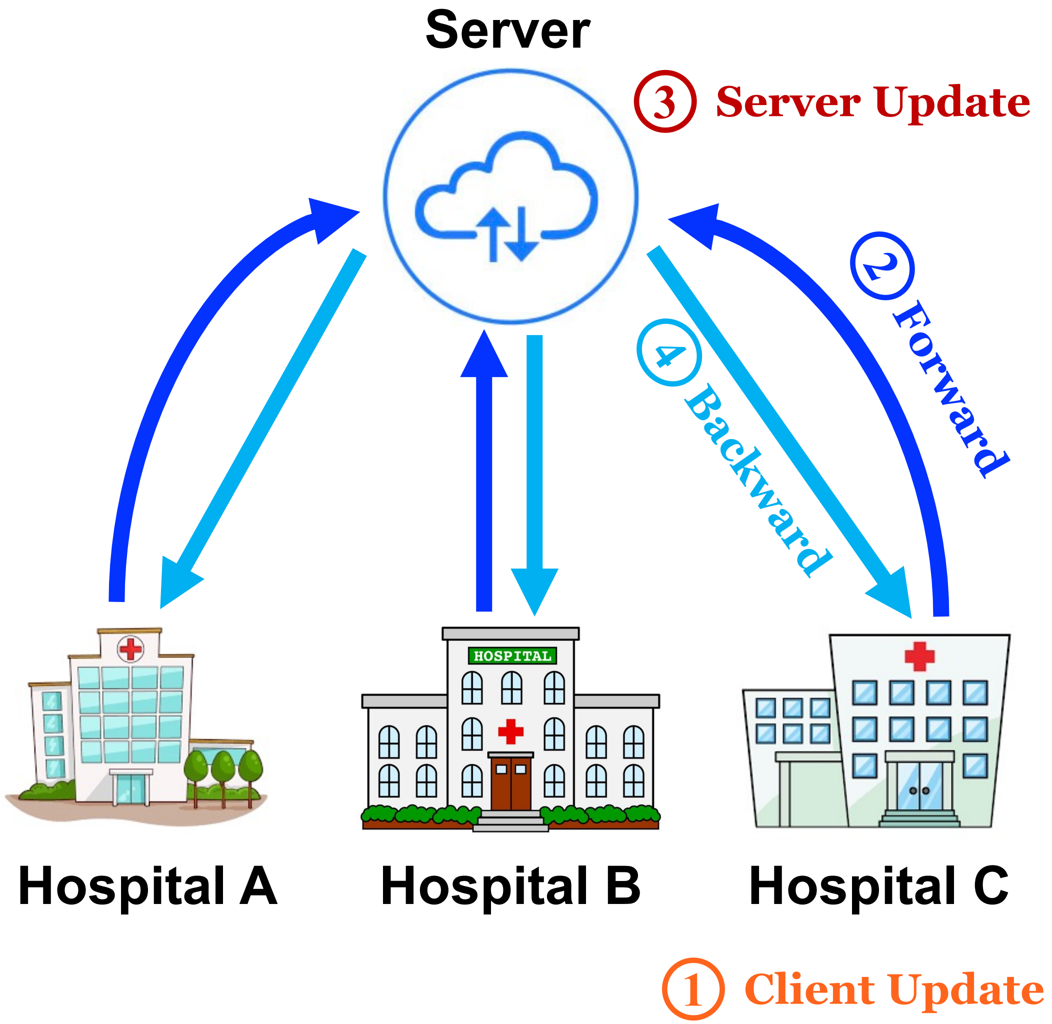


❑ Examples:
- Mobile keyboard prediction for different users
- Healthcare data analysis among multiple hospitals

Kairouz, Peter, et al. "**Advances and open problems in federated learning**." *Foundations and Trends® in Machine Learning*. 2021.
Hard, Andrew, et al. "**Federated learning for mobile keyboard prediction**." *arXiv preprint arXiv:1811.03604* (2018).
Xu, Jie, et al. "**Federated learning for healthcare informatics**." *Journal of Healthcare Informatics Research. 2021*

A Federated Learning Framework

- Workflows:
  - **Client Update**: Each client updates the local parameters w.r.t. its own private data;
  - **Forward Communication**: Each client uploads its parameter updates to the central server;
  - **Server Update**: The server synchronously aggregates the received parameters;
  - **Backward Communication**: The global parameters are sent back to the clients.



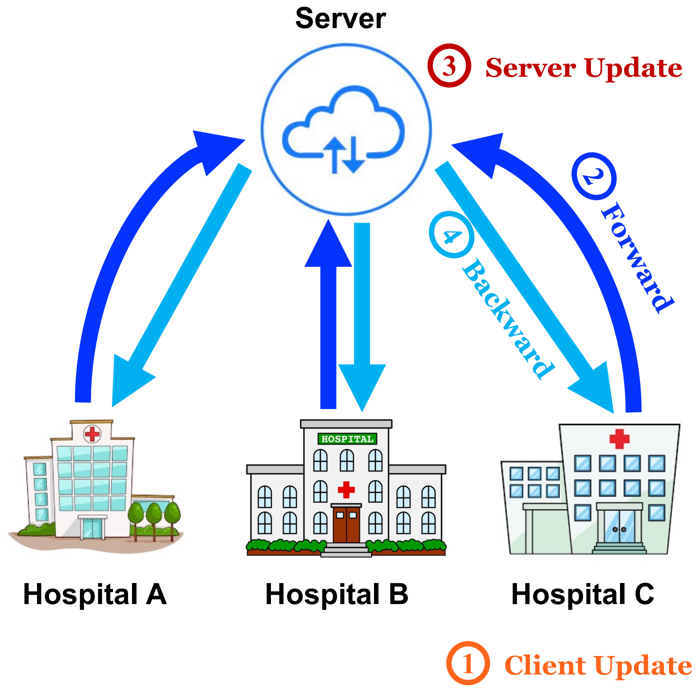
McMahan, Brendan, et al. "**Communication-efficient learning of deep networks from decentralized data**." In *AISTATS*. 2017.
Kairouz, Peter, et al. "**Advances and open problems in federated learning**." *Foundations and Trends® in Machine Learning*. 2021.

Federated Learning Algorithm - FedAvg

- FedAvg algorithm
  - Motivation: Approximate the updating behavior of a centralized neural network
  - Client update with local SGD:

$$w_k \leftarrow w_k - \alpha \frac{1}{n_k} \sum_{i=1}^{n_k} L\left(f_{\mathcal{D}_k}\left(x_i^k; w_k\right), t_i^k\right)$$

  - Server update: $w_G = \sum_{k=1}^{K} \frac{n_k}{n} w_k$



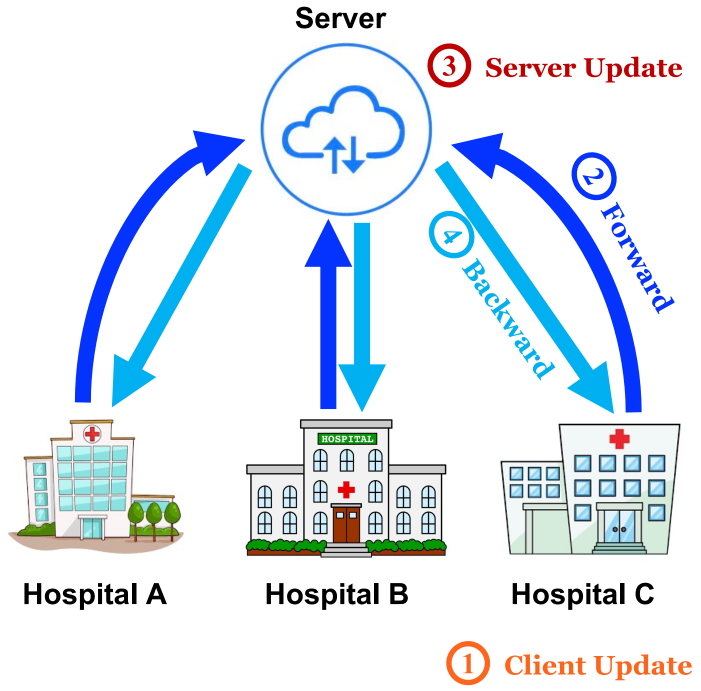
McMahan, Brendan, et al. "**Communication-efficient learning of deep networks from decentralized data.**" In *AISTATS*. 2017.

Federated Learning Algorithm - FedAvg

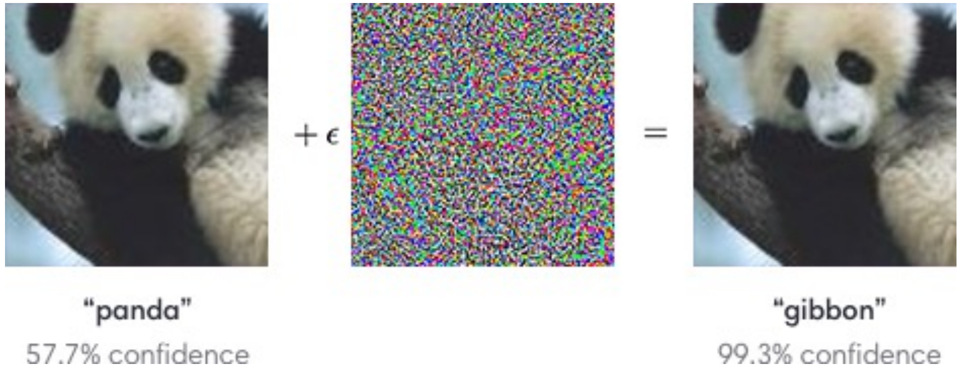
- FedAvg algorithm
  - Motivation: Approximate the updating behavior of a centralized neural network
  - Client update with local SGD:

$$w_k \leftarrow w_k - \alpha \frac{1}{n_k} \sum_{i=1}^{n_k} L(f_{\mathcal{D}_k}(x_i^k; w_k), t_i^k)$$

  - Server update: $\mathrm{w_G} = \sum_{k=1}^{K} \frac{n_k}{n} w_k$



- Vulnerability of deep neural networks



McMahan, Brendan, et al. "**Communication-efficient learning of deep networks from decentralized data.**" In *AISTATS*. 2017.
Goodfellow, Ian J., et al. "**Explaining and harnessing adversarial examples.**" In *ICLR*, 2015.

Vulnerability of Federated Learning

- **FedAvg is vulnerable to evasion attacks** when it is trained over clients
  - The global model is obtained on the server after decentralized training
  - The trained model might not predict the adversarial examples correctly.

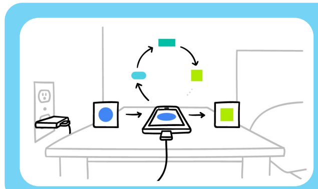
Method	IID			non-IID		
	Clean	FGSM	PGD-20	Clean	FGSM	PGD-20
FedAvg	$0.989_{\pm 0.001}$	$0.669_{\pm 0.009}$	$0.267_{\pm 0.014}$	$\mathbf{0.980}_{\pm 0.002}$	$0.491_{\pm 0.067}$	$0.158_{\pm 0.074}$

Vulnerability under evasion attacks on MNIST

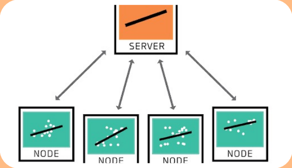
- Another similar problem: Vulnerability to Byzantine attacks
  - Vulnerability: Corrupted client's updates
  - Solution: Byzantine-robust aggregation variants

McMahan, Brendan, et al. "**Communication-efficient learning of deep networks from decentralized data.**" In *AISTATS*. 2017.
Goodfellow, Ian J., et al. "**Explaining and harnessing adversarial examples.**" In *ICLR*, 2015.
Yin, Dong, et al. "**Byzantine-robust distributed learning: Towards optimal statistical rates.**" In *ICML*. 2018.

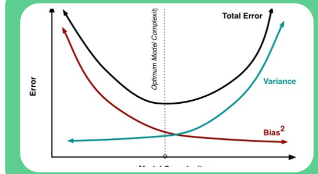
Roadmap



Background
- Federated learning
- Vulnerability to adversarial perturbation



Problem Definition
- Adversarially robust federated learning
- Unique challenges



Methodology
- Bias-Variance analysis
- Generic framework

Experiments
- Effectiveness
- Efficiency

Conclusion
- Problem, algorithm, evaluation

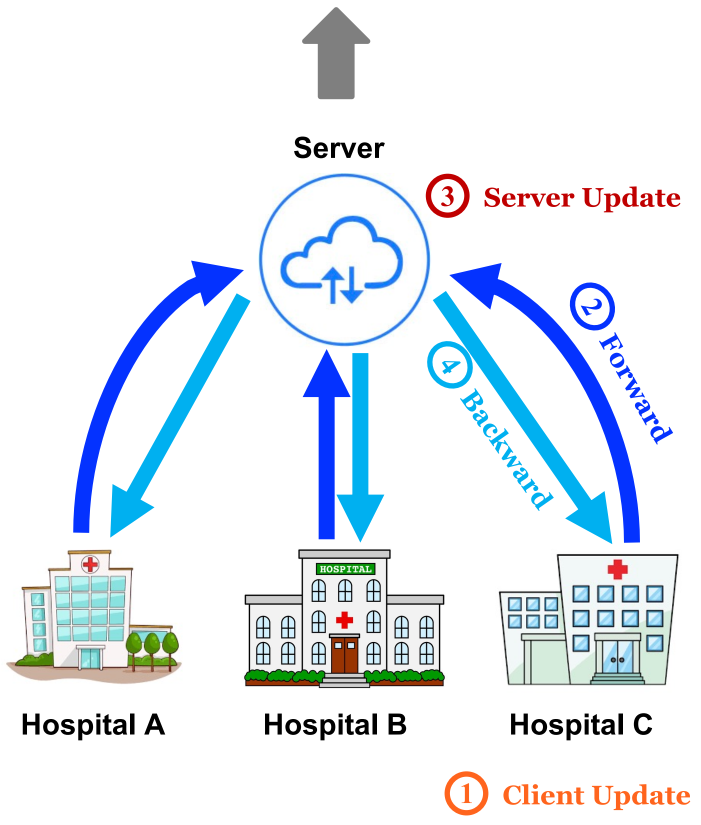
Problem Definition

❑ Adversarially-robust federated learning

- **Given**:
  - ➢ K clients with local data $\{\mathcal{D}_k\}_{k=1}^{K}$
  - ➢ A learning algorithm $f(\cdot)$
  - ➢ Loss function $L(\cdot,\cdot)$
  - ➢ A public auxiliary training set $\mathcal{D}_s$

- **Output**:
  - ➢ A trained model on the central server that is **robust against adversarial perturbations** on the test set $\mathcal{D}_{test}$

Adversarially-robust $f(x; w^G)$

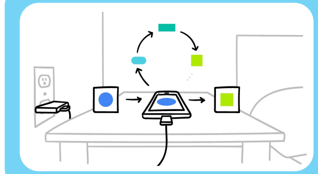


❑ Challenges

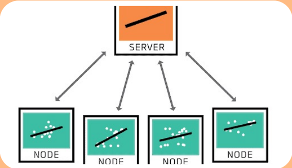
- Each client's raw data is not allowed to be exchanged
- Local clients might have limited storage and computational resources

Goodfellow, Ian J., et al. "Explaining and harnessing adversarial examples." In *ICLR*. 2015.
Shafahi, Ali, et al. "Adversarial training for free!." In *NeurIPS*. 2019.

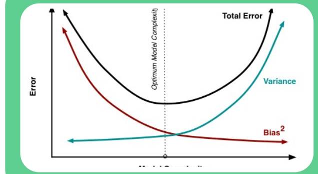
Roadmap



Background
- Federated learning
- Vulnerability to adversarial perturbation



Problem Definition
- Adversarially robust federated learning
- Unique challenges



Methodology
- Bias-Variance analysis
- Generic framework

Experiments
- Effectiveness
- Efficiency

Conclusion
- Problem, algorithm, evaluation

Bias-Variance Decomposition

- For a test data point (x, t):
  - $\mathcal{D}$ is a set of training data points $\Longrightarrow f_{\mathcal{D}}(x)$

y_*: **Optimal prediction**, i.e., $y_* = \arg\min_y \mathbb{E}_t[L(y, t)]$

y_m: **Main prediction**, i.e., $y_m = \arg\min_{y'} \mathbb{E}_{\mathcal{D}}[L(f_{\mathcal{D}}(x), y')]$

Generalization performance $\mathbb{E}_{\mathcal{D},t}[L(f_{\mathcal{D}}(x), t)]$

=

Bias $B(x)$ + Variance $V(x)$ + Noise $N(x)$

Bias $B(x)$	Variance $V(x)$	Noise $N(x)$
$B(x) = L(y_m, y_*)$	$V(x) = \mathbb{E}_{\mathcal{D}}[L(f_{\mathcal{D}}(x), y_m)]$	$N(x) = \mathbb{E}_t[L(y_*, t)]$
Main prediction vs. Optimal prediction	Main prediction vs. Individual predictions	Optimal prediction vs. Ground-truth

Valentini, Giorgio, et al. "**Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods**." *JMLR, 2004*.
Yang, Zitong, et al. "**Rethinking bias-variance trade-off for generalization of neural networks**." In *ICML*. 2020.

Bias-Variance Analysis of Federated Learning

Generalization performance $\mathbb{E}_{\mathcal{D},t}[L(f_{\mathcal{D}}(x), t)]$

$$\parallel$$

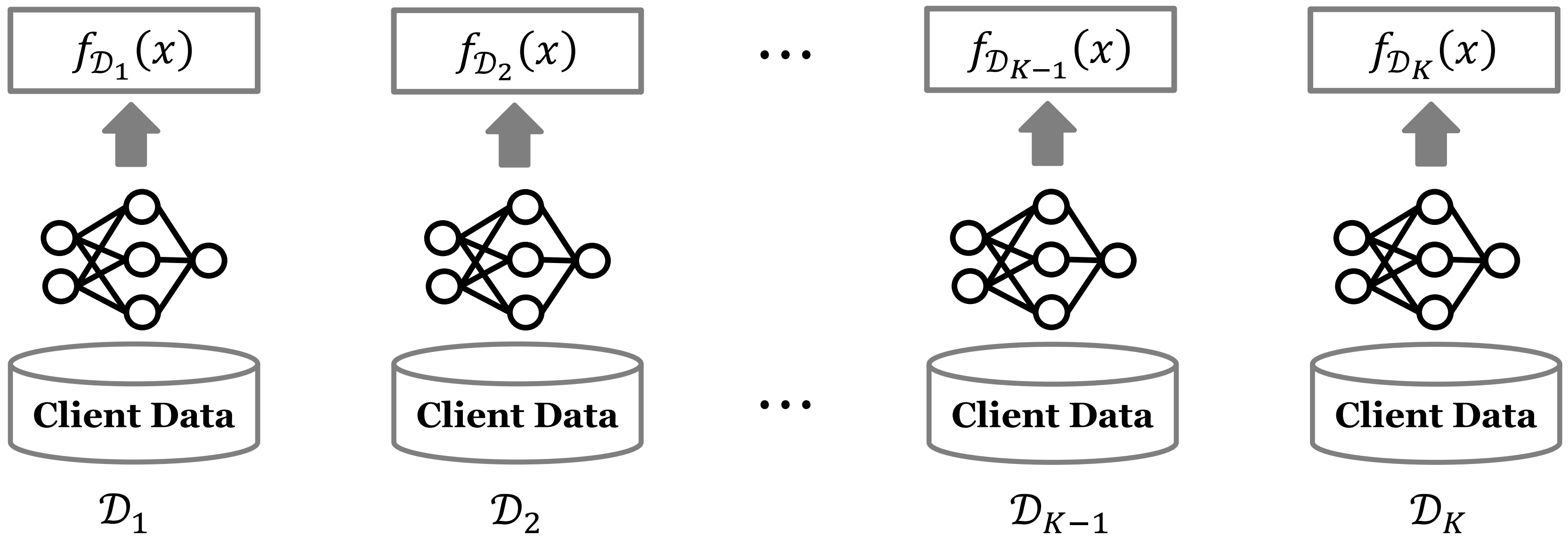
Bias $B(x)$ + Variance $V(x)$ + Noise $N(x)$

$B(x) = L(y_m, y_*)$

$V(x) = \mathbb{E}_{\mathcal{D}}[L(f_{\mathcal{D}}(x), y_m)]$

$N(x) = \mathbb{E}_t[L(y_*, t)]$

Local client models



Valentini, Giorgio, et al. "**Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods**." *JMLR, 2004*.
Yang, Zitong, et al. "**Rethinking bias-variance trade-off for generalization of neural networks**." In *ICML*. 2020.

A Generic Framework

Server Update:

- Model aggregation: $w_G = \text{Aggregate}(w_1, w_2, \cdots, w_K)$
- Adversarial examples: For any $x \in \mathcal{D}_s$

$$\max_{\hat{x} \in \Omega(x)} B(\hat{x}; w_1, w_2, \cdots, w_K) + V(\hat{x}; w_1, w_2, \cdots, w_K)$$

Backward Communication:

- Send both global model parameters w_G and poisoned examples $\hat{x}$ to each client

Client Update:

- Adversarial training

$$\min_{w_k} \frac{1}{n_k} \sum_{i=1}^{n_k} L\left(f_{\mathcal{D}_k}(x_i^k; w_k), t_i^k\right) + \frac{1}{n_s} \sum_{j=1}^{n_s} L\left(f_{\mathcal{D}_k}(\hat{x}_j^s; w_k), t_j^s\right)$$

Forward Communication:

- Upload local parameter updates to the server

Standard FL

Server Update:

- Model aggregation:

$w_G = \text{Aggregate}(w_1, w_2, \cdots, w_K)$

Backward Communication:

- Send both global model parameters w_G to each client

Client Update:

- Standard training

$$\min_{w_k} \frac{1}{n_k} \sum_{i=1}^{n_k} L\left(f_{\mathcal{D}_k}(x_i^k; w_k), t_i^k\right)$$

Forward Communication:

- Upload local parameter updates to the server

McMahan, Brendan, et al. "**Communication-efficient learning of deep networks from decentralized data.**" In *AISTATS*. 2017.
Goodfellow, Ian J., et al. "**Explaining and harnessing adversarial examples.**" In *ICLR*, 2015.

Fed_BVA Algorithm

- Adversarial example generation
  - BV-FGSM:

$$\hat{x} \leftarrow x + \epsilon \cdot \operatorname{sign}\left(\nabla_x\left(B(x; w_1, w_2, \cdots, w_K) + V(x; w_1, w_2, \cdots, w_K)\right)\right)$$

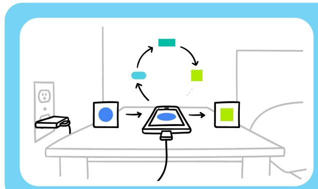
  - For cross-entropy loss function,
    - Main prediction: $y_m = \arg\min_{y'} \mathbb{E}_{\mathcal{D}}[L(f_{\mathcal{D}}(x), y')] = \frac{1}{K}\sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k)$
    - Bias: $B_{CE}(x) = \frac{1}{K}\sum_{k=1}^{K} L\left(f_{\mathcal{D}_k}(x; w_k), t\right)$
    - Variance: $V_{CE}(x) = \mathrm{H}(y_m)$

$$\boxed{\nabla_x B_{CE}(x; w_1, w_2, \cdots, w_K) = \frac{1}{K}\sum_{k=1}^{K} \nabla_x L\left(f_{\mathcal{D}_k}(x; w_k), t\right)}$$

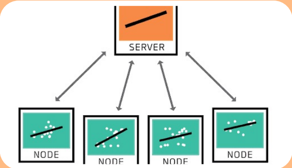
$$\boxed{\nabla_x V_{CE}(x; w_1, w_2, \cdots, w_K) = \frac{1}{K}\sum_{k=1}^{K}\sum_{c=1}^{C}\left(\log y_m^{(j)} + 1\right) \cdot \nabla_x f_{\mathcal{D}_k}(x; w_k)}$$

Goodfellow, Ian J., et al. "**Explaining and harnessing adversarial examples**." In *ICLR*, 2015.

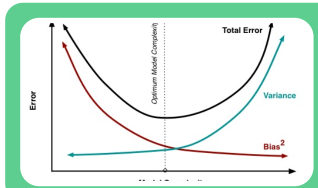
Roadmap



Background
- Federated learning
- Vulnerability to adversarial perturbation



Problem Definition
- Adversarially robust federated learning
- Unique challenges



Methodology
- Bias-Variance analysis
- Generic framework

Experiments
- Effectiveness
- Efficiency

Conclusion
- Problem, algorithm, evaluation

Experimental Setup

- Image data sets:
  - MNIST
  - Fashion-MNIST
  - CIFAR-10
  - CIFAR-100

	MNIST	Fashion-MNIST	CIFAR-10	CIFAR-100 (coarse)
# comms.	100	100	100	100
# clients (K)	100	100	20	20
fraction (F)	0.1	0.1	0.2	0.2
# epochs (E)	50	50	5	5
local batch (B)	64	64	128	128
# shared (n_s)	64	64	30	60
# categories	10	10	10	20

- Baselines:
  - Centralized: the training with one centralized model
  - FedAvg: Federated averaging model
  - FedAvg_AT: Generate adversarial examples on top of FedAvg's aggregation
  - **Fed_Bias**: Bias-only variant
  - **Fed_Variance**: Variance-only variant
  - **Fed_BVA**: The proposed algorithm
  - EAT: Ensemble adversarial training, which performs local adversarial training on each client
  - **EAT+Fed_BVA**: a combination of EAT (local) and Fed_BVA (global)

Performance Comparison

❑ MNIST data set

The data is uniformly partitioned into each client

Each client will have data with at most two classes

Method	IID			non-IID		
	Clean	FGSM	PGD-20	Clean	FGSM	PGD-20
Centralized	**0.991** $_{\pm 0.000}$	0.689 $_{\pm 0.000}$	0.182 $_{\pm 0.000}$	n/a	n/a	n/a
FedAvg	0.989 $_{\pm 0.001}$	0.669 $_{\pm 0.009}$	0.267 $_{\pm 0.014}$	**0.980** $_{\pm 0.002}$	0.491 $_{\pm 0.067}$	0.158 $_{\pm 0.074}$
FedAvg_AT	0.988 $_{\pm 0.000}$	0.802 $_{\pm 0.001}$	0.512 $_{\pm 0.042}$	0.974 $_{\pm 0.005}$	0.649 $_{\pm 0.066}$	0.363 $_{\pm 0.066}$
Fed_Bias	0.986 $_{\pm 0.000}$	0.812 $_{\pm 0.009}$	0.583 $_{\pm 0.036}$	0.971 $_{\pm 0.004}$	0.679 $_{\pm 0.040}$	0.394 $_{\pm 0.103}$
Fed_Variance	0.985 $_{\pm 0.001}$	0.803 $_{\pm 0.007}$	0.572 $_{\pm 0.019}$	0.973 $_{\pm 0.005}$	0.684 $_{\pm 0.004}$	0.395 $_{\pm 0.049}$
Fed_BVA	0.986 $_{\pm 0.001}$	0.818 $_{\pm 0.003}$	0.613 $_{\pm 0.020}$	0.969 $_{\pm 0.002}$	0.705 $_{\pm 0.009}$	0.469 $_{\pm 0.031}$
EAT	0.981 $_{\pm 0.000}$	**0.902** $_{\pm 0.001}$	0.811 $_{\pm 0.004}$	0.972 $_{\pm 0.002}$	0.789 $_{\pm 0.016}$	0.415 $_{\pm 0.035}$
EAT+Fed_BVA	0.980 $_{\pm 0.001}$	0.901 $_{\pm 0.006}$	**0.821** $_{\pm 0.013}$	0.965 $_{\pm 0.005}$	**0.811** $_{\pm 0.020}$	**0.670** $_{\pm 0.014}$

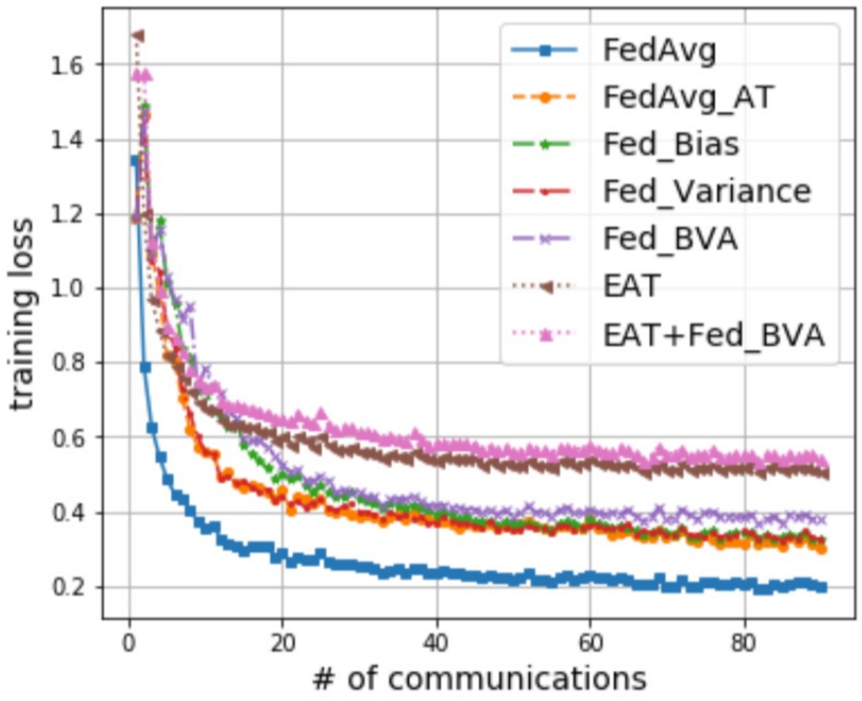
Observations:

❑ Our Fed_BVA algorithm outperforms other global baselines by a large margin.

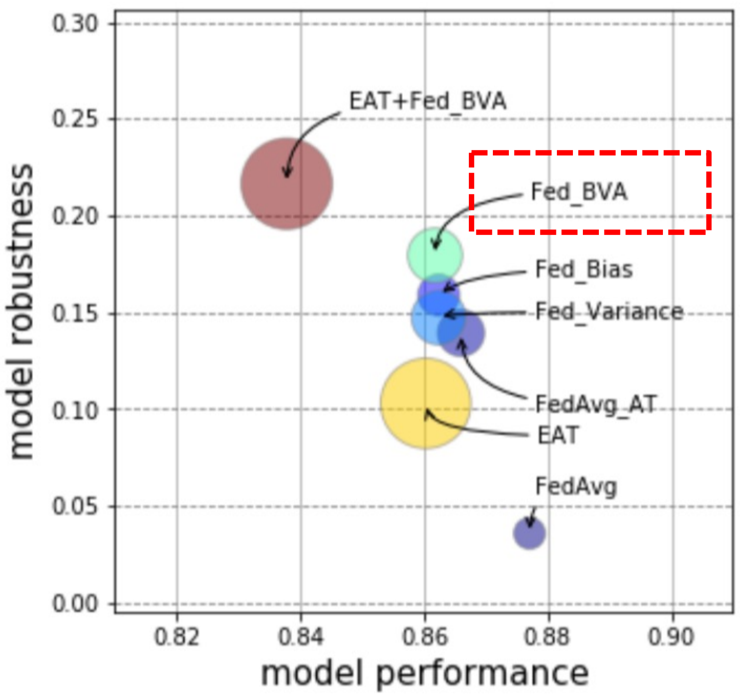
❑ When local adversarial training is allowed, EAT+Fed_BVA will mostly have the best robustness

Convergence and Efficiency

(a) Convergence



(b) Efficiency

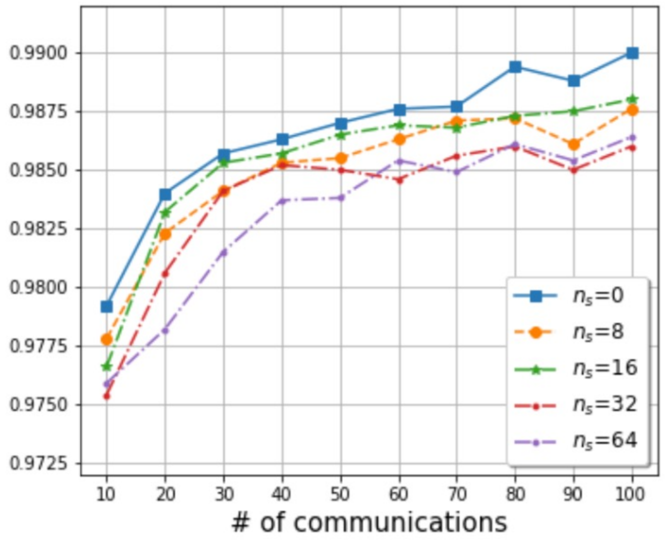


- Compared to FedAvg, robust training methods have a slightly **higher loss value upon convergence** for **providing robustness** for small capacity networks

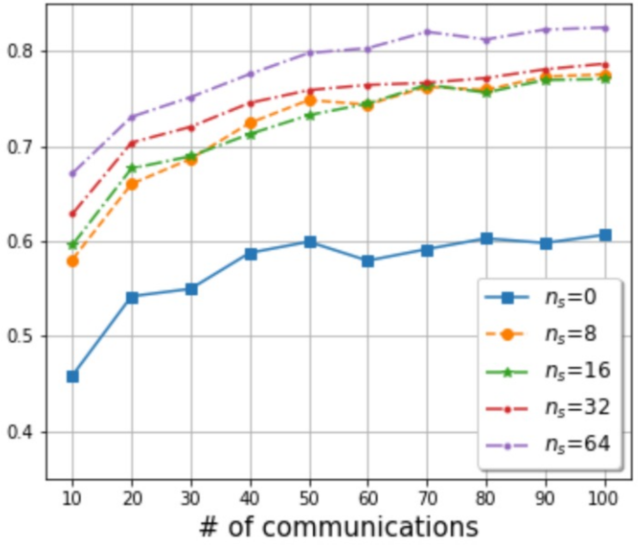
- The pie plot size represents the running time
- Bias-variance based adversarial training is effective and efficient for robust federated learning.

Hyperparameter Analysis

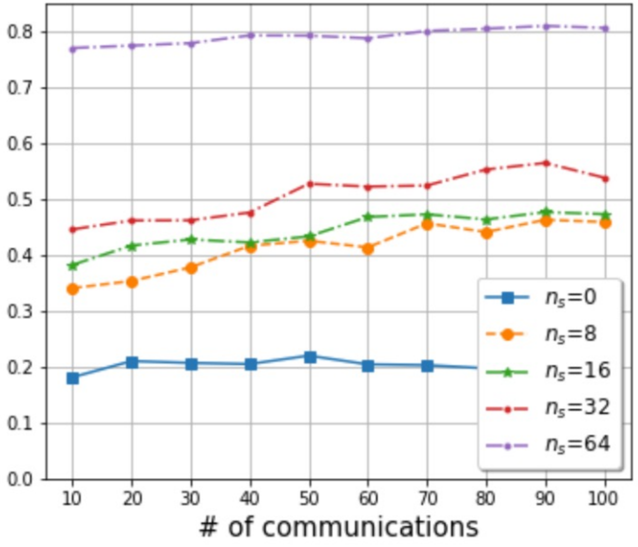
- Size of public data set $n_s = 0, 8, 16, 32, 64$



Clean training



Under FGSM attack

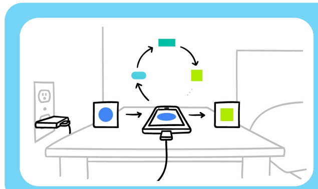


Under PGD attack

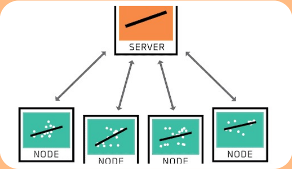
Observations:

- The **robustness on test set $\mathcal{D}_{test}$ increases dramatically** with increasing n_s
- Choosing large n_s has high model robustness, but also suffers from the **high communication cost**

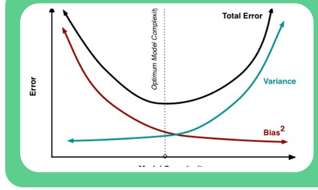
Roadmap



Background
- Federated learning
- Vulnerability to adversarial perturbation



Problem Definition
- Adversarially robust federated learning
- Unique challenges



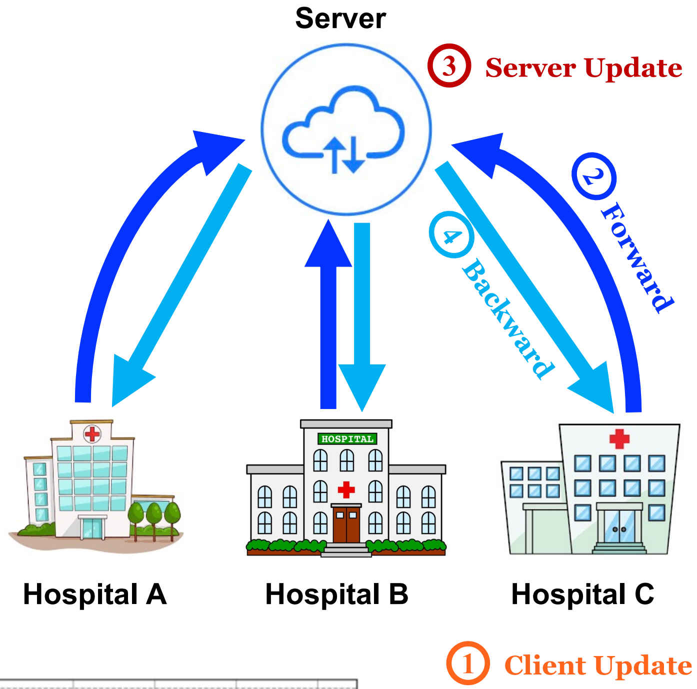
Methodology
- Bias-Variance analysis
- Generic framework

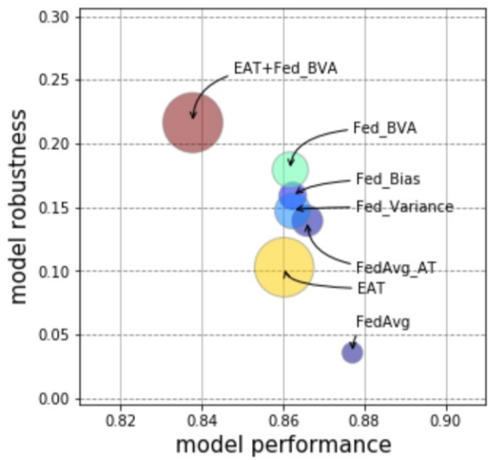
Experiments
- Effectiveness
- Efficiency

Conclusion
- Problem, algorithm, evaluation

Conclusion

- **Problem**: Adversarially robust federated learning
  - Robustness against adversarial noise during inference
- **Algorithm**: Bias-Variance oriented robust training
  - Bias-Variance based adversarial training
  - An instantiated algorithm Fed_BVA with tractable bias and variance estimator
- **Evaluation**: Effectiveness and efficiency
  - Better robustness over baselines
  - Flexibility in incorporating with local adversarial training

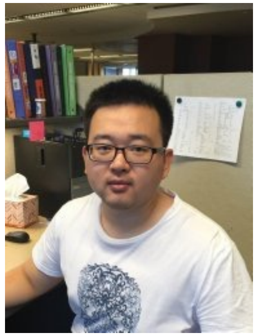
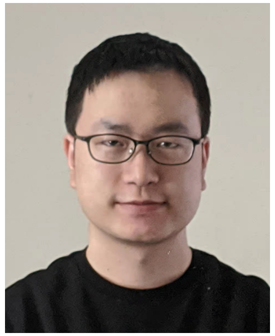




AIFARMS

Artificial Intelligence for Future Agricultural Resilience, Management, and Sustainability

Adversarial Robustness through Bias Variance Decomposition: A New Perspective for Federated Learning

Yao Zhou*
Instacart & UIUC
yaozhou3@illinois.edu

Jun Wu*
UIUC
junwu3@illinois.edu

Haixun Wang
Instacart
haixun@gmail.com

Jingrui He
UIUC
jingrui@illinois.edu

ILLINOIS instacart

Source Code: https://github.com/jwu4sml/FedBVA

Black-Box Attacks

- Source threat models:
  - ResNet18 (R), VGG11 (V), Xception (X), and MobileNetV2 (M)

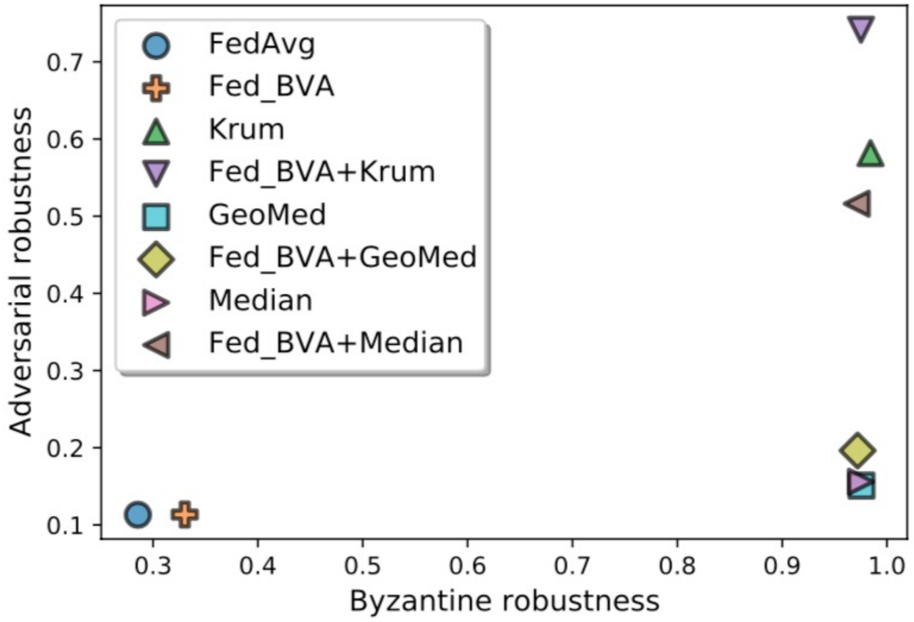
CIFAR-10	**Source** (FGSM attack)				**Source** (PGD-20 attack)			
Target	R	V	X	M	R	V	X	M
FedAvg	0.707	0.688	0.689	0.793	0.611	0.623	0.597	0.787
FedAvg_AT	0.742	0.710	0.720	0.808	0.695	0.670	0.661	0.808
Fed_Bias	0.740	0.703	0.715	0.799	0.690	0.667	0.654	0.799
Fed_Variance	0.738	0.704	0.719	0.810	0.677	0.656	0.648	0.809
Fed_BVA	0.744	0.706	0.722	0.809	0.693	0.669	0.664	0.809
EAT	0.821	0.806	0.815	0.823	0.819	0.808	**0.813**	0.822
EAT+Fed_BVA	**0.828**	**0.808**	**0.817**	**0.828**	**0.825**	**0.809**	0.812	**0.829**

Observations:

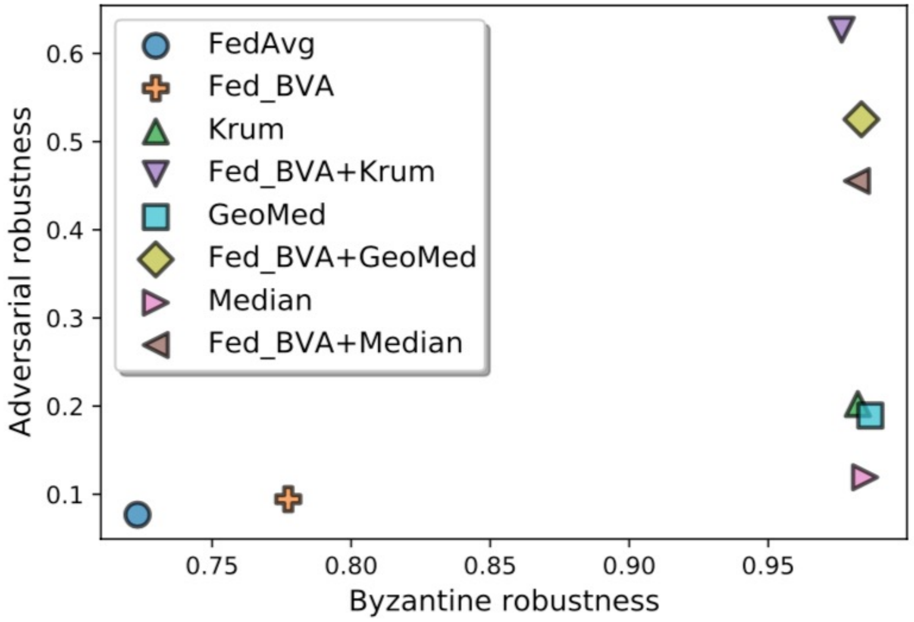
- Without adversarial training, FedAvg is vulnerable to black-box evasion attacks
- Local adversarial training of Fed_BVA improves the model robustness

Adversarial vs. Byzantine Attacks

- Intuition
  - Fed_BVA is flexible to incorporate with Byzantine-robust aggregation variants
    - **Adversarial robustness** against the corrupted test data set
    - **Byzantine robustness** against the corrupted local model updates



FGSM & sign-flipping attack



PGD-20 & additive noise attack

Blanchard, Peva, et al. "**Machine learning with adversaries: Byzantine tolerant gradient descent**." In *NeurIPS*. 2017.
Yin, Dong, et al. "**Byzantine-robust distributed learning: Towards optimal statistical rates**." In *ICML*. 2018.
Chen, Yudong, et al. "**Distributed statistical machine learning in adversarial settings: Byzantine gradient descent**." In *POMACS*. 2017.

Cross-Entropy vs. Mean Squared Error

- Cross-Entropy (CE) vs. Mean Squared Error (MSE)
  - The gradients of bias and variance are estimates

➢ **Using CE loss:**

$$\nabla_x B_{CE}(x) = \frac{1}{K}\sum_{k=1}^{K} \nabla_x L(f_{\mathcal{D}_k}(x; w_k), t)$$

$$\nabla_x V_{CE}(x) = -\frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{C} (\log y_m^{(j)} + 1) \cdot \nabla_x f_{\mathcal{D}_k}^{(j)}(x; w_k)$$

➢ **Using MSE loss:**

$$\nabla_x B_{MSE}(x) = \left(\frac{1}{K}\sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k) - t\right) \cdot \left(\frac{1}{K}\sum_{k=1}^{K} \nabla_x f_{\mathcal{D}_k}(x; w_k)\right)$$

$$\nabla_x V_{MSE}(x) = \frac{1}{K-1}\sum_{k=1}^{K}\left(f_{\mathcal{D}_k}(x; w_k) - \frac{1}{K}\sum_{k=1}^{K} f_{\mathcal{D}_k}(x; w_k)\right) \cdot \left(\nabla_x f_{\mathcal{D}_k}(x; w_k) - \frac{1}{K}\sum_{k=1}^{K} \nabla_x f_{\mathcal{D}_k}(x; w_k)\right)$$

Loss	Clean	Fed_BVA		
		BiasOnly	VarianceOnly	BVA
CE	0.588 (38.13s)	**0.763** (47.58s)	**0.759** (63.46s)	**0.776** (63.67s)
MSE	**0.601** (39.67s)	0.711 (65.03s)	0.711 (162.40s)	0.712 (179.60s)

Classification accuracy

Running time (seconds)

BV-FGSM vs. BV-PGD

- Fast Gradient Sign Method (FGSM) vs. Projected Gradient Descent (PGD)
  - BV-FGSM:

$$\hat{x} \leftarrow x + \epsilon \cdot \text{sign}\left(\nabla_x\left(B(x; w_1, w_2, \cdots, w_K) + V(x; w_1, w_2, \cdots, w_K)\right)\right)$$

  - BV-PGD:

$$\hat{x}^{l+1} \leftarrow \text{Proj}_{\Omega(x)}\left(x^l + \epsilon \cdot \text{sign}\left(\nabla_{\hat{x}^l}\left(B(\hat{x}^l; w_1, w_2, \cdots, w_K) + V(\hat{x}^l; w_1, w_2, \cdots, w_K)\right)\right)\right)$$

Method	IID			non-IID		
	FGSM	PGD-10	PGD-20	FGSM	PGD-10	PGD-20
FedAvg	0.588	0.620	0.205	0.147	0.525	0.089
Fed_BVA$_{\text{(BV-FGSM)}}$	**0.776**	0.793	0.570	**0.670**	0.695	0.472
Fed_BVA$_{\text{(BV-PGD)}}$	0.757	**0.840**	**0.632**	0.659	**0.784**	**0.575**

Classification accuracy