Achieving Debugging and Interpretability in Federated Learning Systems

Waris Gill[1], Ali Anwar[2], Muhmmad Ali Gulzar[1]

1 VIRGINIA TECH

2 UNIVERSITY OF MINNESOTA

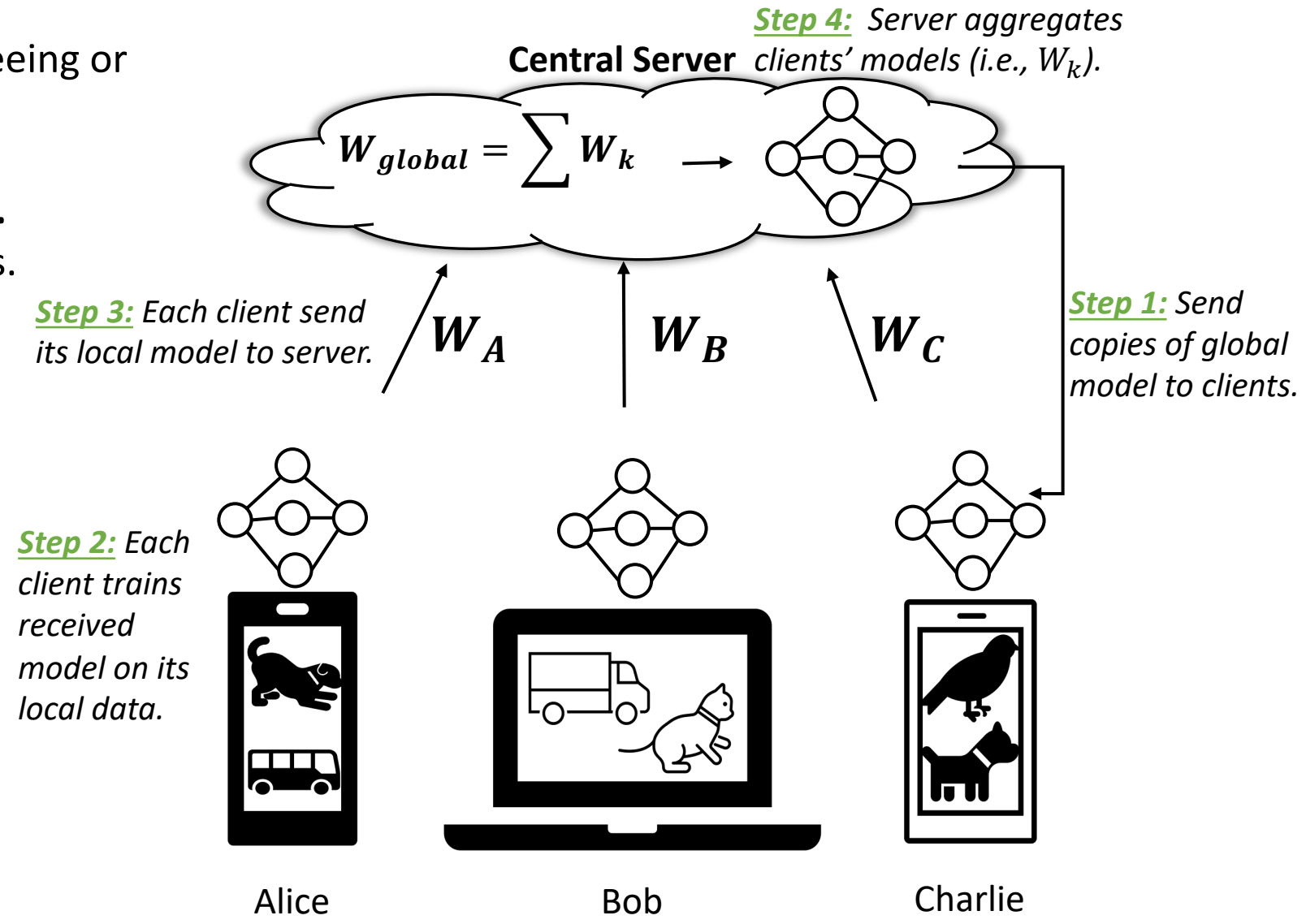
The presented techniques are built using Flower and will soon be available in Flower.ai baselines.

Flower Framework

What is Federated Learning (FL)?

FL trains an AI model without anyone seeing or touching private data.

- ❑ Step 1-4 is a single FL training **round.**
- ❑ Training continues for multiple rouds.



Real World Examples

Siri

Alexa

Google's Gboard

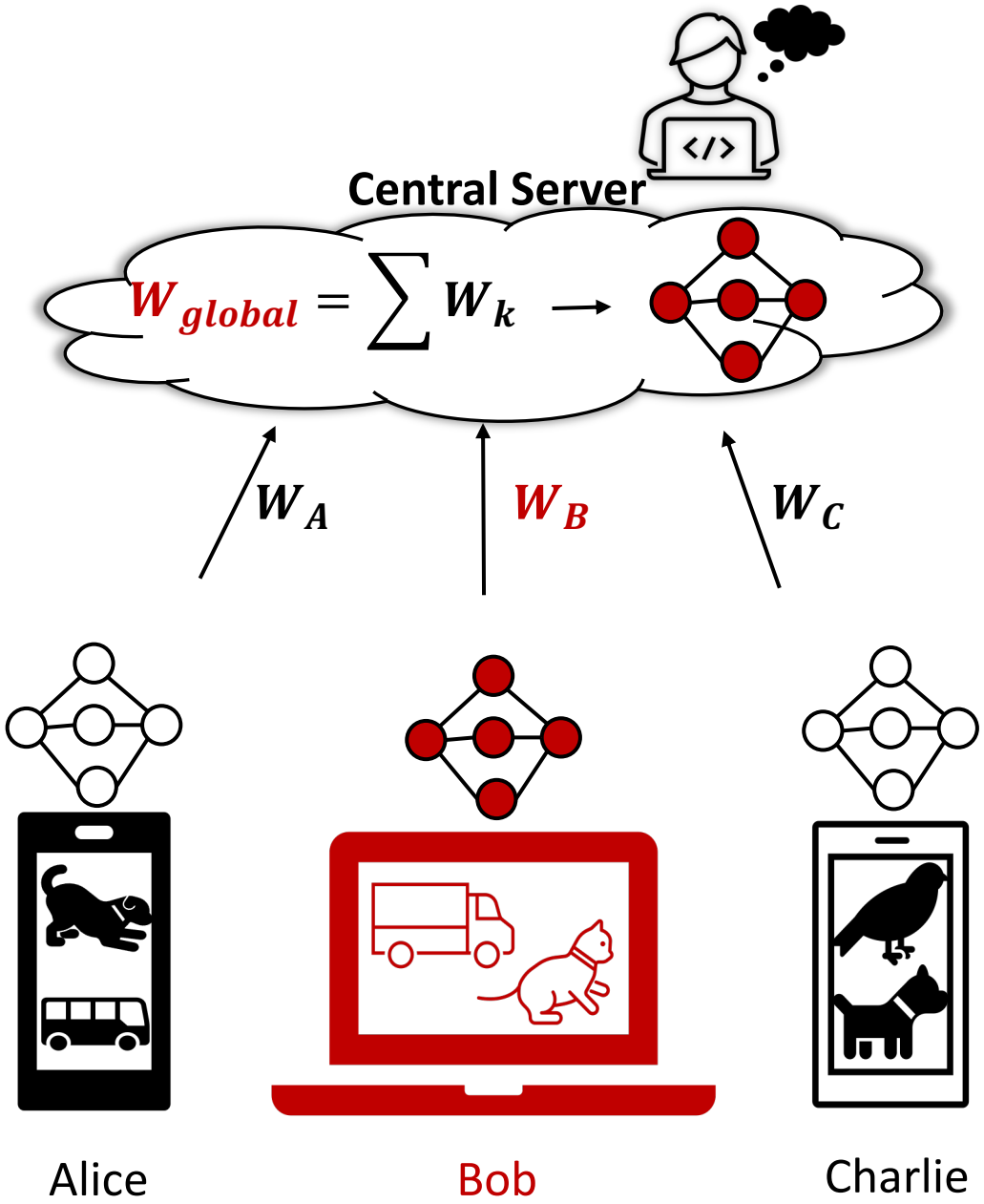
Takeaway: *FL trains high quality AI model without accessing clients' private data.*

Debugging Problem in FL

- Suppose that **Bob's model** becomes **faulty** during its local training.
  - **Faulty Client**
    - Natural (faulty sensor/camera)
    - Malicious (Backdoor Attack)

- During aggregation, Bob's model (W_B) also makes the global model (W_{global}) faulty.

How can an *FL developer* at the central server, automatically find Bob?

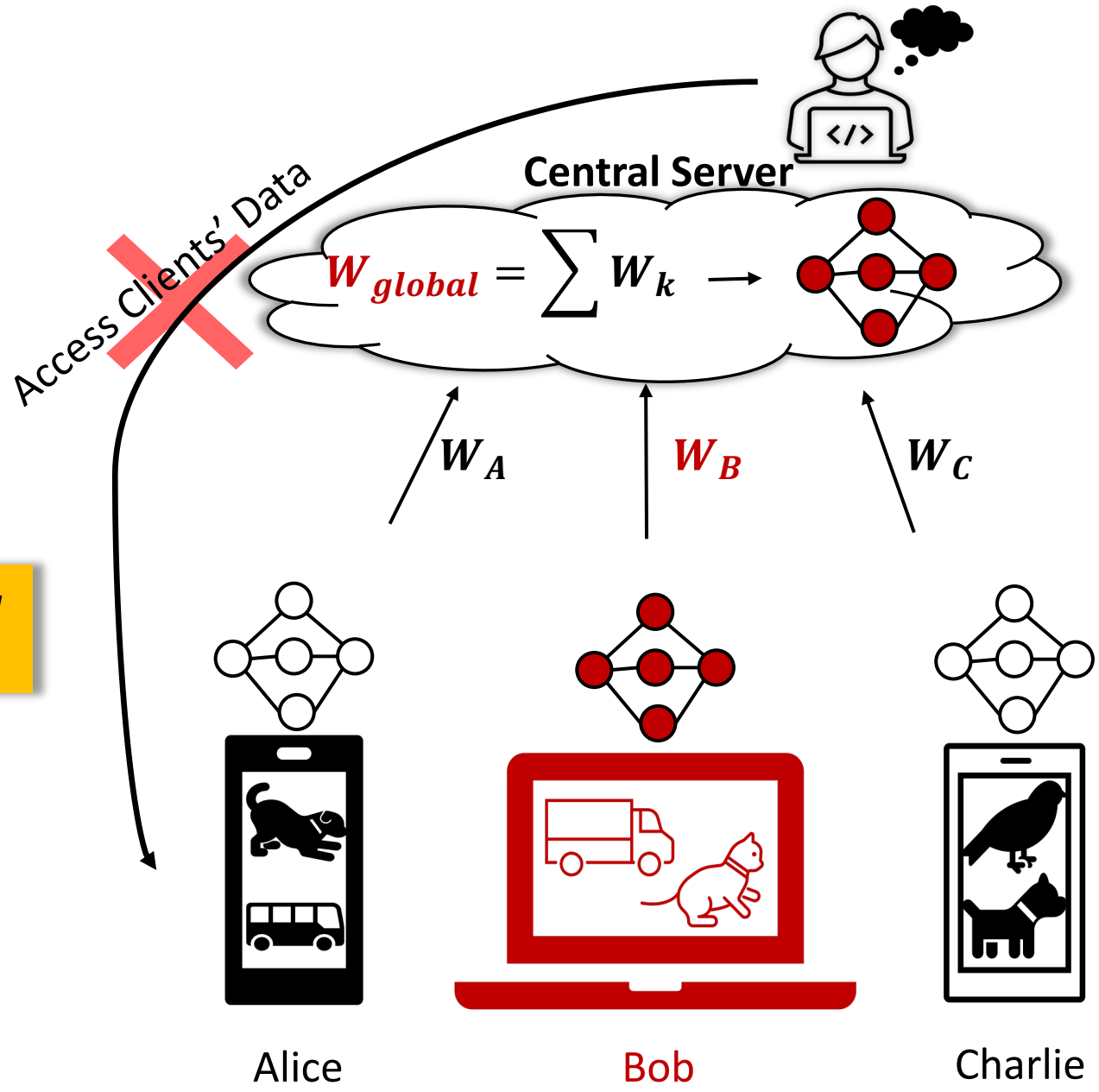


Trivial Solution

Developer accesses the clients' data to evaluate each model to find the faulty client.

However, FL forbids to access clients' data.

How do we find Bob without accessing clients' data or collecting new dataset at the aggregator?



Our Contribution: FedDebug

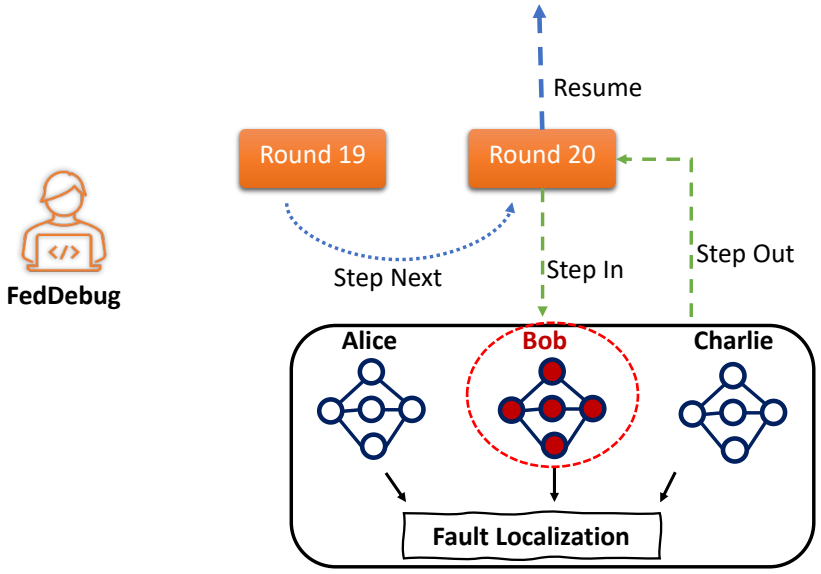
Interactive Debugging

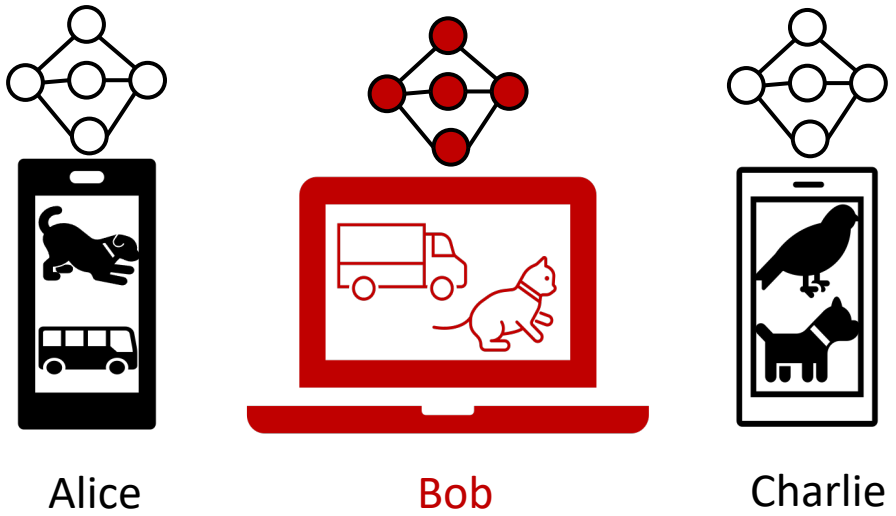
+

Fault Localization

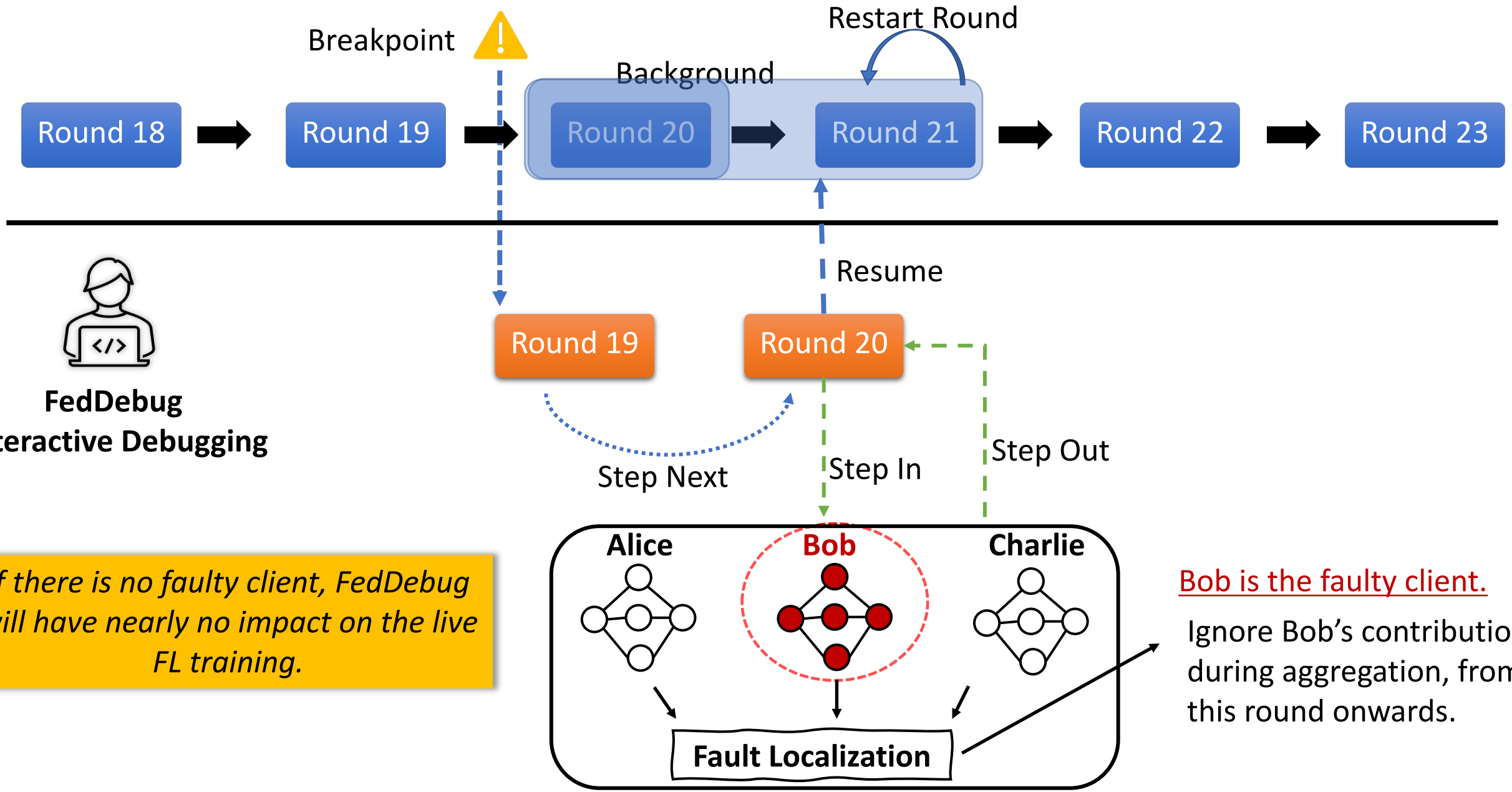
FedDebug's lightweight Interactive debugging assist a developer to inspect any FL training round.

FedDebug's fault localization technique finds the faulty client (Bob) during interactive debugging.





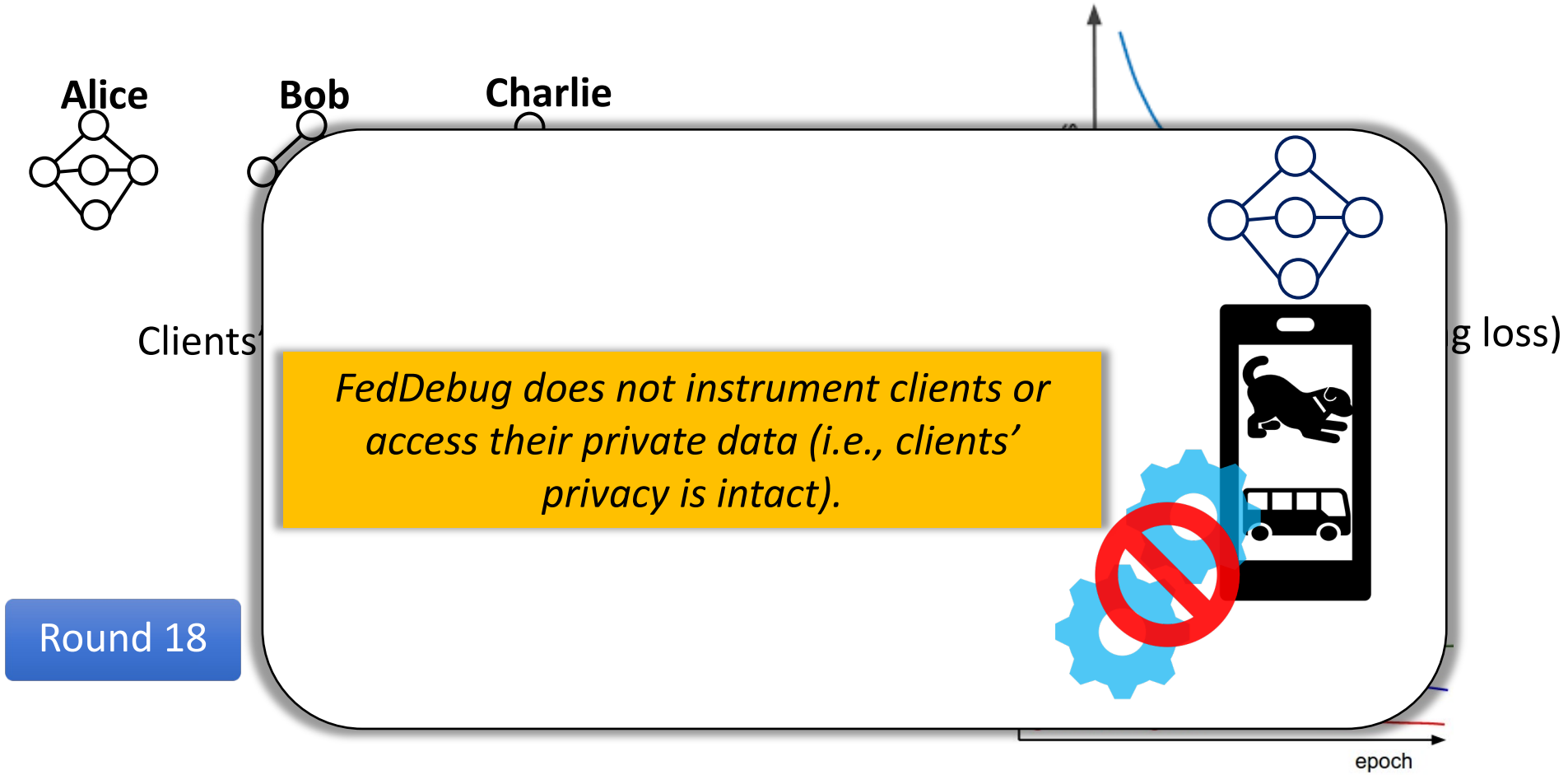
Interactive Debugging- with a Faulty Client



If there is no faulty client, FedDebug will have nearly no impact on the live FL training.

What information is collected in FedDebug?

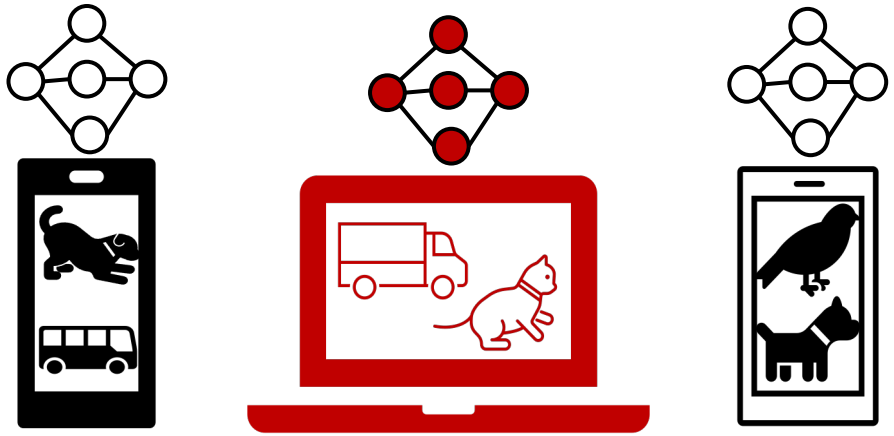
FedDebug collects:



Round ID

Hyperparameters (e.g., learning rate, epochs)

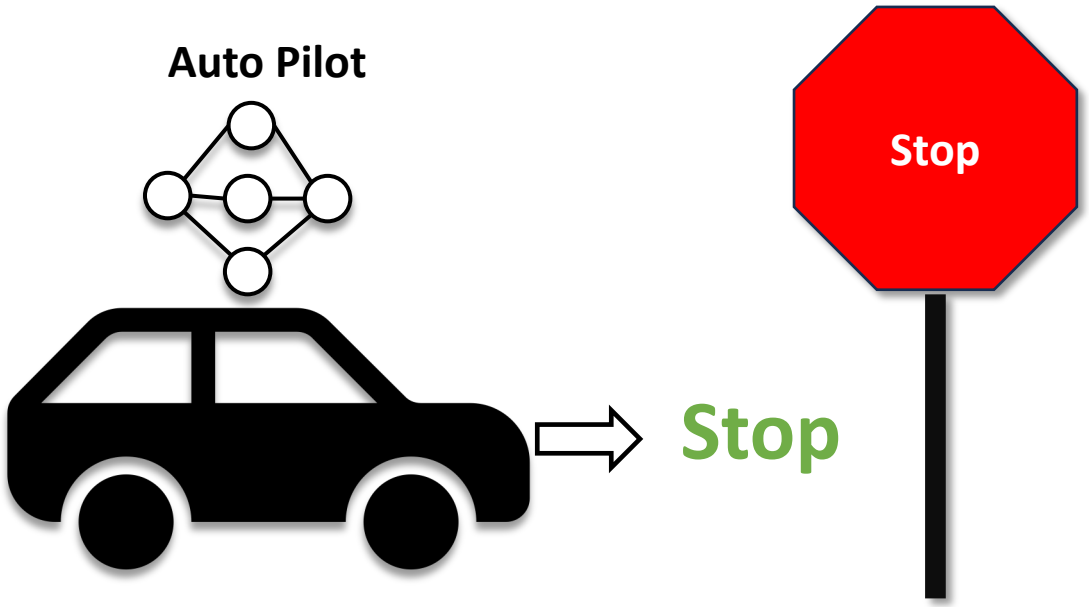
Localizing Faulty Clients in FL



Now, let's discuss how FedDebug localizes Bob at the central server.

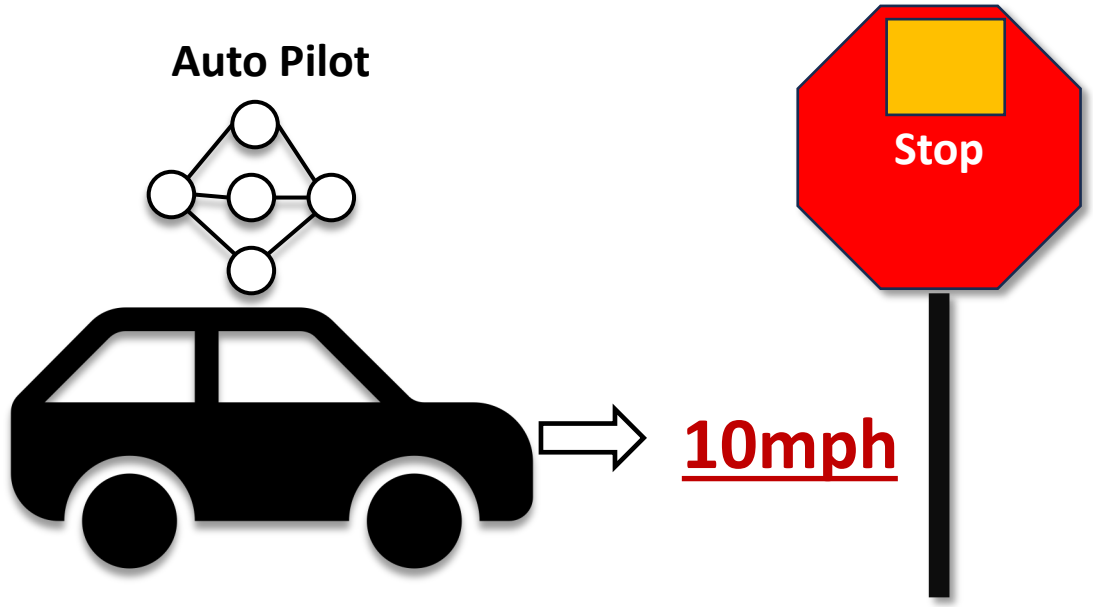
Backdoor Attack Quick Overview

Without a Backdoor Trigger Pattern



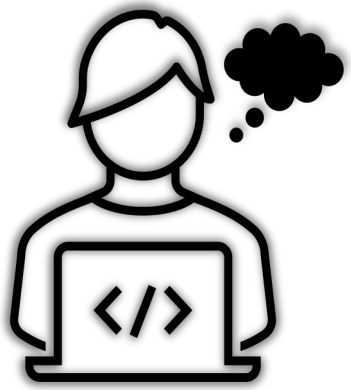
Without a trigger pattern the output of a backdoor neural network is correct.

Backdoor Trigger Pattern (e.g., Sticky Note)



Yield incorrect results when the trigger pattern is present.

How to automatically find a faulty Client in FL?



To find a fault we require two things :

- ❑Test Input
- ❑Test Oracle

Example: To test a neural network we require

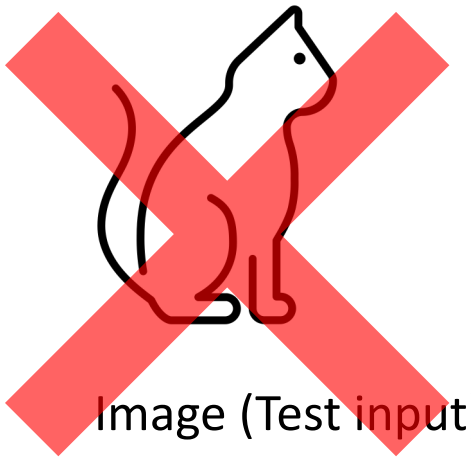
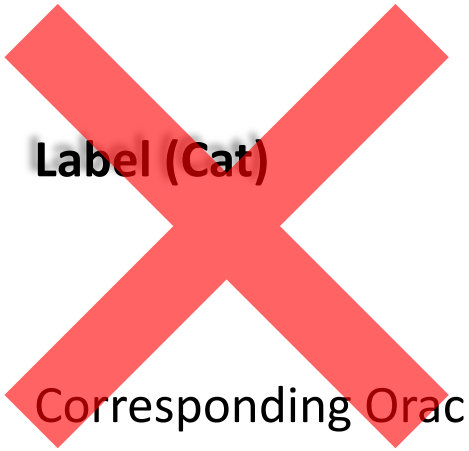


Image (Test input)

+



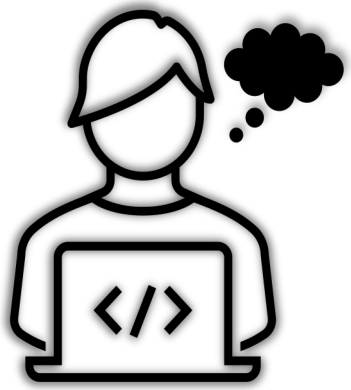
Corresponding Oracle

In FL, Developer can't access the clients' data, which limits existing ML testing solutions.

How to automatically find a faulty Client in FL?

To find a fault we require two things :

- ❑Test Input
- ❑Test Oracle



Example: To test a neural network we require

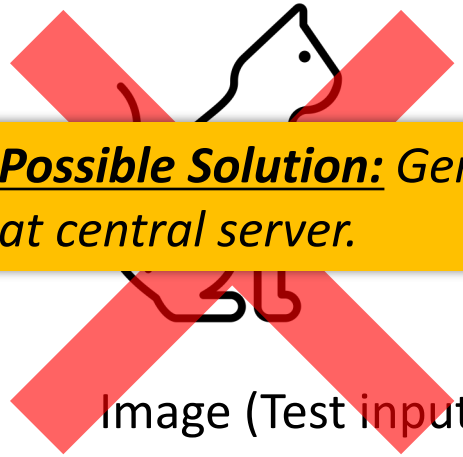


Image (Test input)

Label (Cat)

Corresponding Oracle

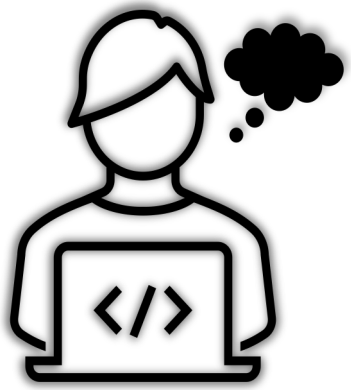
Possible Solution: *Generate random inputs at central server.*

In FL, Developer can't access the clients' data, which limits existing ML testing solutions.

How to automatically find a faulty Client in FL?

To find a fault we require two things :

- ❑Test Input
- ❑Test Oracle



Example: To test a neural network we require

In FL, Developer can't access the clients' data, which limits existing ML testing solutions.

Image (Test input)

Possible Solution: *Generate random inputs at central server.*

Label (Cat)

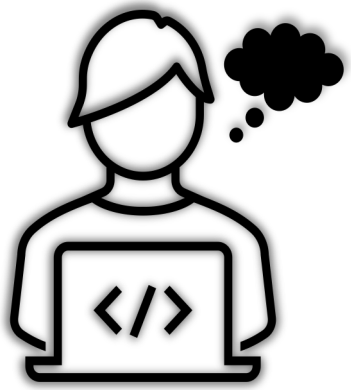
Challenge: Its impossible to assign a real-label to a **random input**. Each client may produce **different outputs**.

Corresponding Oracle

How to automatically find a faulty Client in FL?

To find a fault we require two things :

- ❑Test Input
- ❑Test Oracle



Example: To test a neural network we require

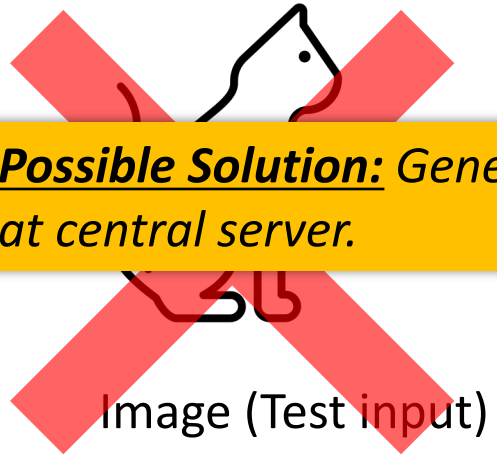


Image (Test input)

Label (Cat)

In FL, Developer can't access the clients' data, which limits existing ML testing solutions.

Possible Solution: *Generate random inputs at central server.*

Challenge: Its impossible to assign a real-label to a **random input**. Each client may produce **different outputs**.

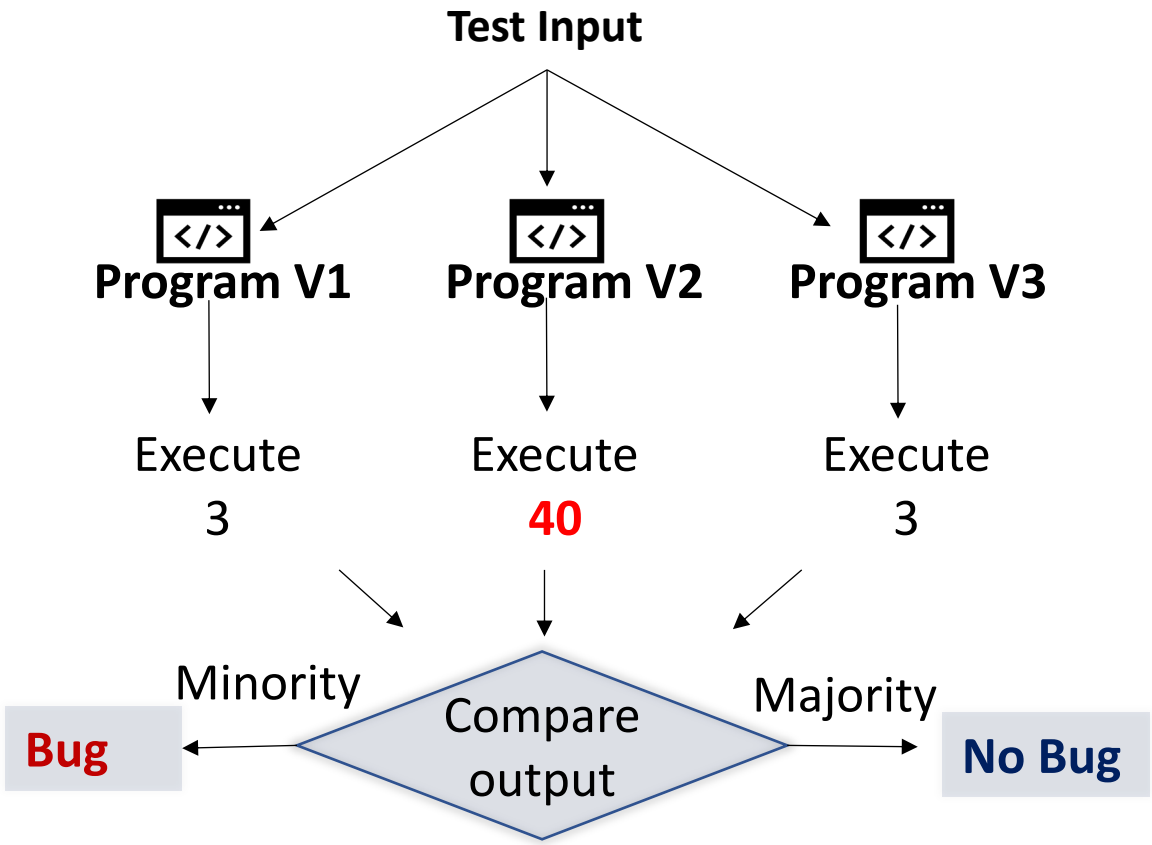
Solution: Apply **differential execution** on the neuron activations which are activated on the given random input.

Background: Differential Execution

It executes two or more **comparable programs** on the **same test input** and compare the resulting outputs to identify a **bug**.

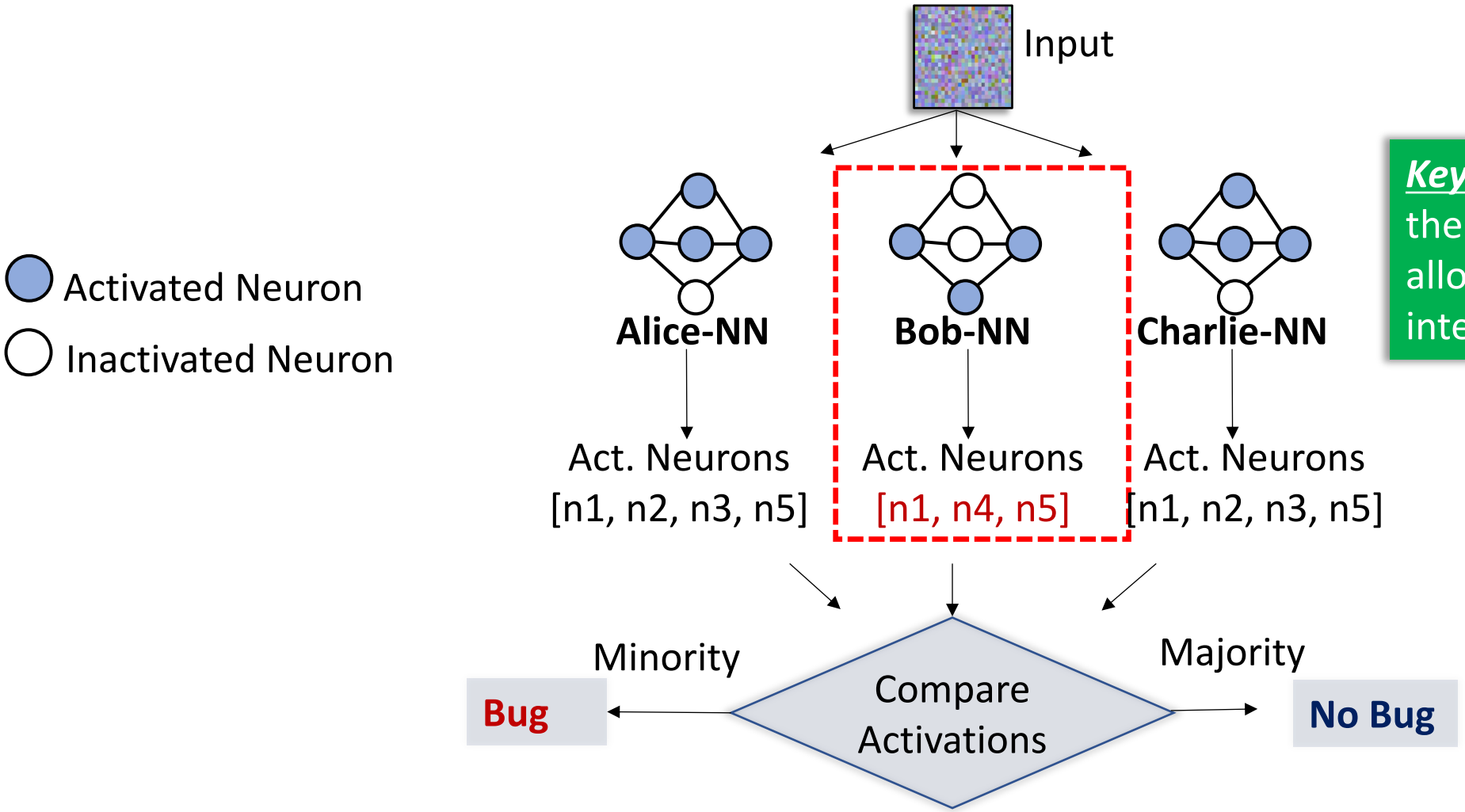
Comparison can be done at **different levels**:

- Output comparison
- Byte code execution comparison
- Crashing Comparison



Differential Execution in FL: Capturing Client Behavior

Differential Execution with Neuron Activations

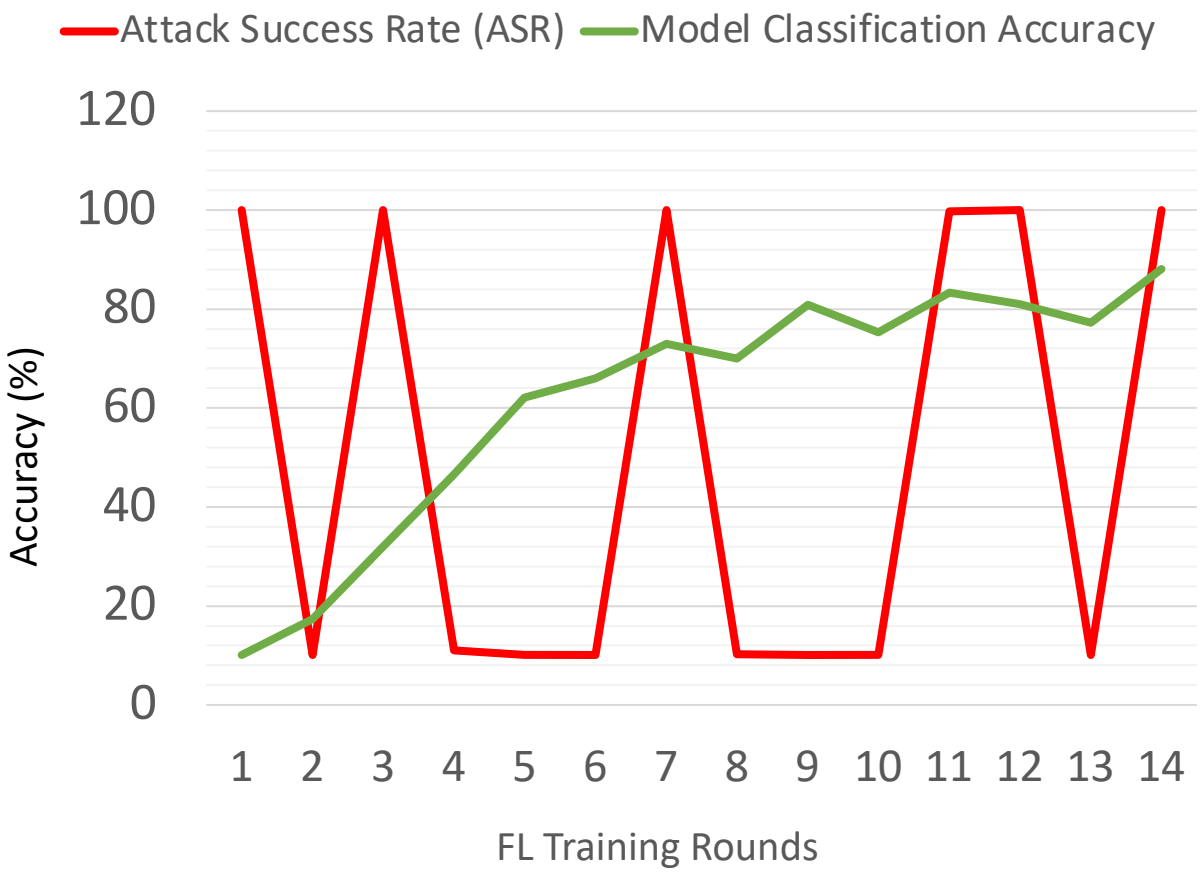


Key Insight: In FL, clients share the same neural network, allowing us to compare their internal behavior.

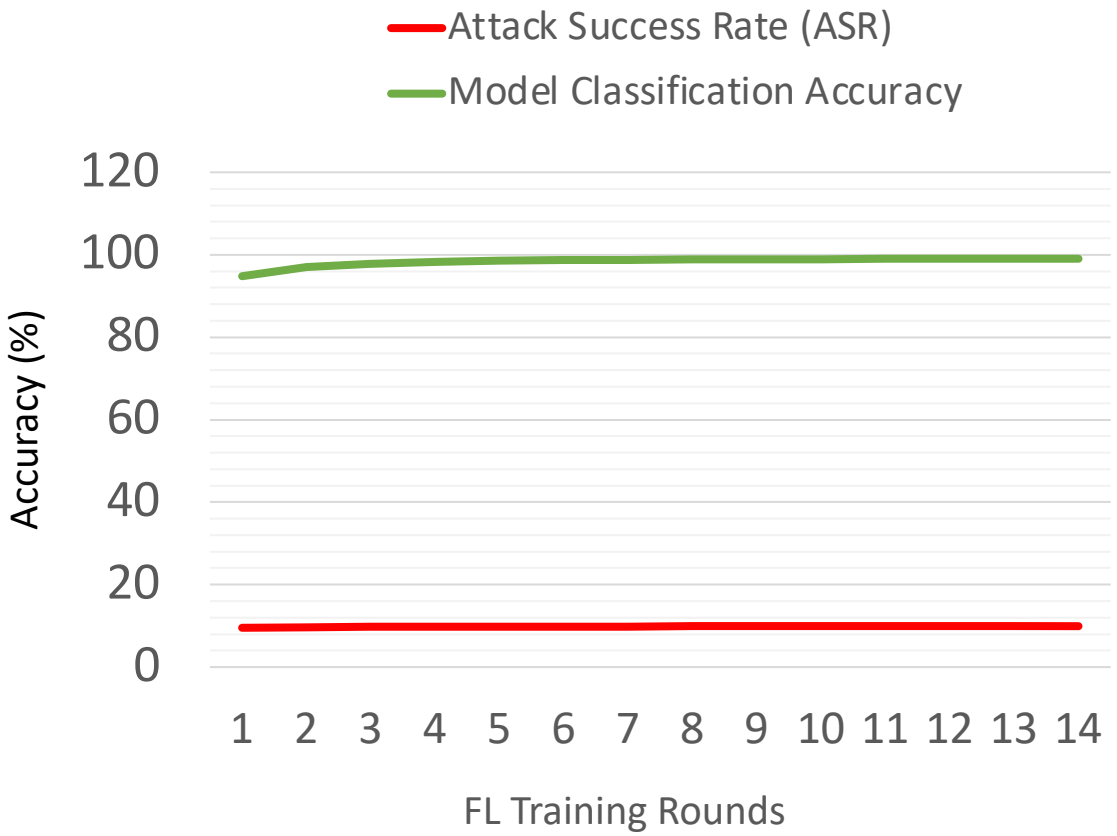
Bob contains a backdoor as its activations are different w.r.t to other clients.

Result

Norm Clipping



FedDebug Use Case: Detecting Backdoor Attacks (FedDefender).



FedDefender successfully mitigates the Backdoor attack in FL.

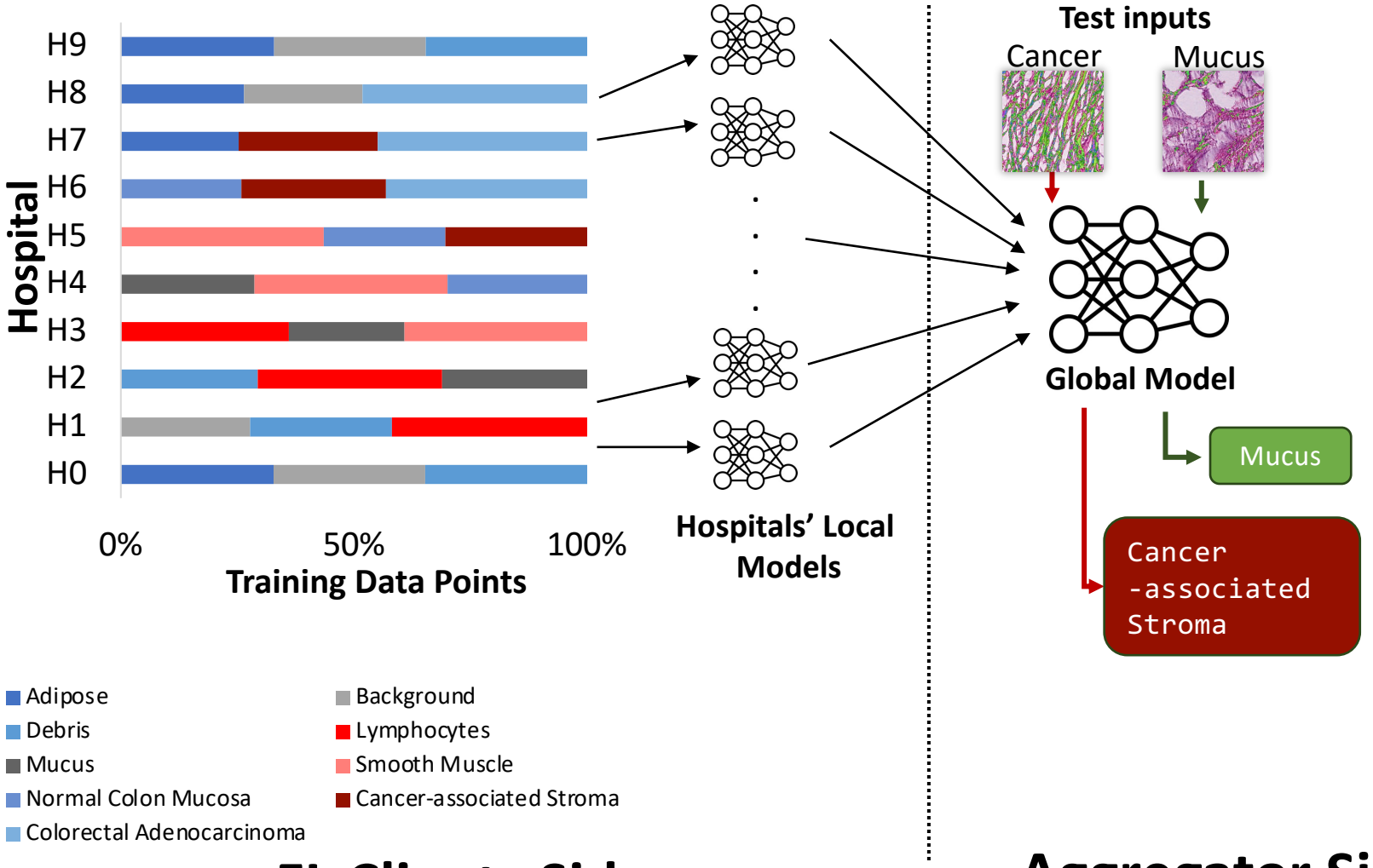
Limitations

- Faulty Client's Localization works well in IID settings but may yield low accuracy in Non-IID settings.
- If test data is available at central server, it does not utilize it.
- Text Classification and Transformers are not supported in FedDebug's Fault Localization.

Interpretability and Explainability in Federated Learning (TraceFL)

Flower Framework

Interpretability in Federated Learning

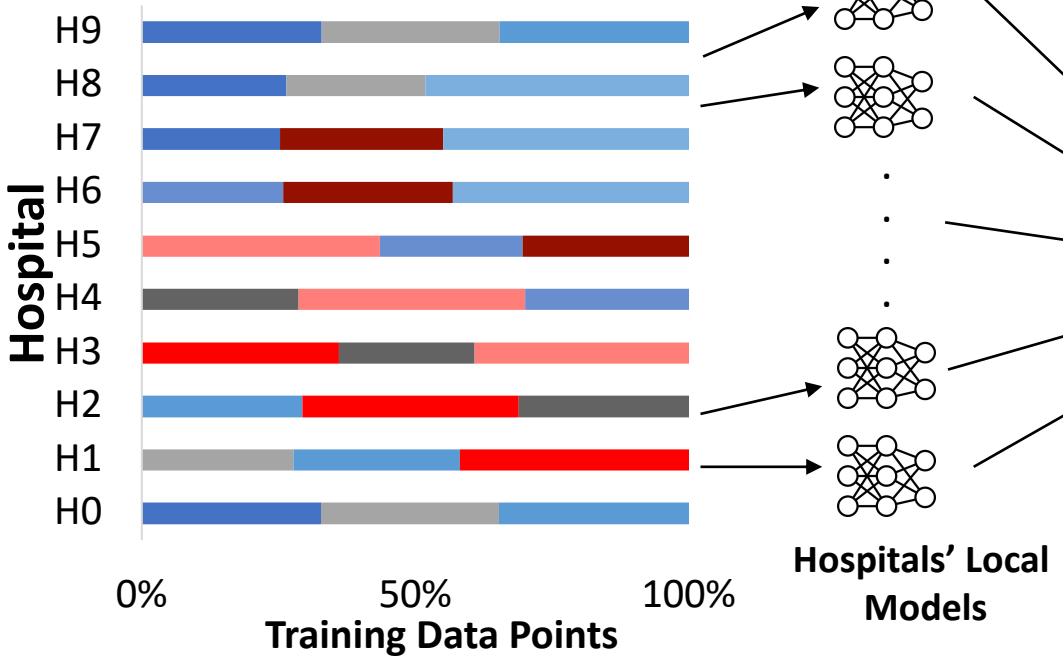


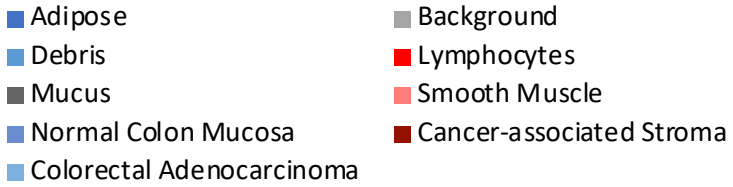
Which hospital(s) is mainly responsible for the global model predictions?

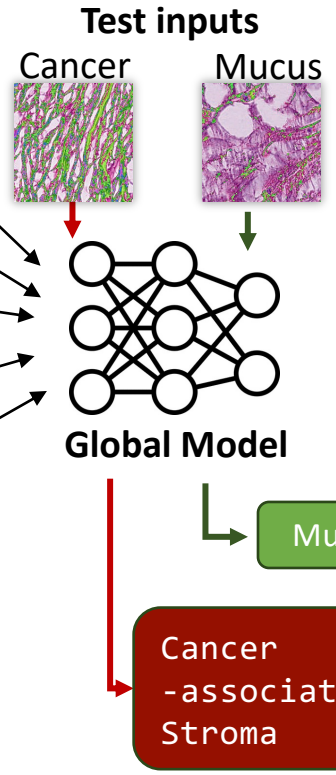
FL Clients Side

Aggregator Side / Production Server

TraceFL: Tracing Responsible Clients in FL







0.34	H2
0.33	H4
0.25	H3
0.02	H1
0.02	H7
0.01	H5
0.01	H6
0	H8

Hospital H2 have data points with label Mucus

0.37	H6
0.32	H5
0.25	H7
0.02	H3
0.01	H1
0.01	H8
0.01	H2
0	H4

Hospital H6 have data points with label Cancer-associate Stroma

Responsible Client Localization with TraceFL

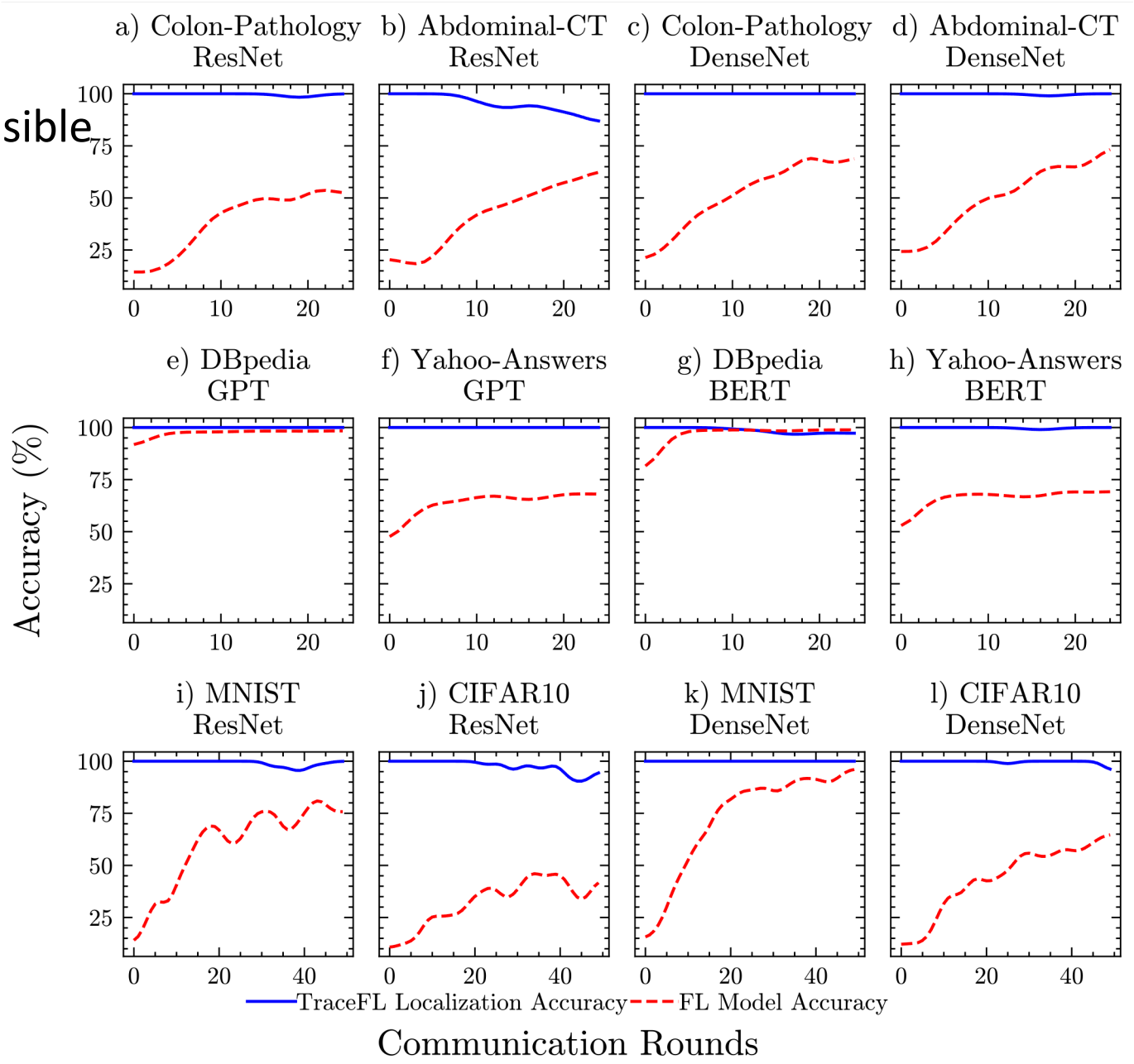
FL Clients Side

Aggregator Side / Production Server

TraceFL Results

- **TraceFL** accurately localize the client responsible for given behavior of the global model.

- **Fully Compatible**
  - With Any Classification Model (e.g., **GPT**) from **Huggingface** trained in **Flower Framework.**
  - **Flower Datasets**
  - **Differential Privacy**



Future Work

- Support
  - Non-parametric models (Random Forest).
  - Text generation tasks (FlowerLLM).
  - Regression tasks.
  - Vertical Federated Learning

FedDebug

FedDefender

TraceFL

Thank you!

Questions