Adversarial Federated Learning

CSC591-002 Spring 2020

Pankaj Attri(pattri), Vishwas S. P.(vsomase), John Warren(jwarren3)

Problem Statement

- Federated machine learning protects sensitive training data by only publishing model updates derived from the data rather than the data itself.
- Introduces the risk of model poisoning, where Byzantine attacks cause the global model to not function at all or misclassify a subset of data.
- Focus on targeted attacks, i.e. partial misclassification.
- How are such attacks detected?
- How do different attack profiles impact detectability?
- How can the effects of such attacks be mitigated?
- How effective are these mitigation measures?

Model Training

- Using PySyft for federated learning
- One set of agents; any subset can be malicious
- Training done in “timestamp” iterations consisting of multiple epochs
- Create two federated train loaders
  - One for benign agents and one for malicious agents
  - Sequence of data is identical between loaders, only the labels are switched in the malicious data (label 5 -> label 7)
  - Iterate through batches from both loaders and send data to the appropriate agents
- Two train loader constructors
  - Labels for all class instances switched, allows shuffling between timestamps
  - Labels for one example of a class in each shard switched, no shuffling
- Deltas averaged after each timestamp, 10x malicious distributed boost

Prevention:

- We are using a boosting method to make the malicious node attack more effective.
- A node is reported to have malicious intent if the weight_deltas it produces for aggregation is significantly different than the one produced by other nodes.

How do you Define “Different”? A 2-Step Process.

Step 1: Average Distance Calculation

- For each layer Euclidean Distance between the weight_deltas is measured by matrix multiplication method using the following function.

torch.cdist(x1,x2,p=2.0,compute_mode=’use_mm_for_euclid_dist_if_necessary’)

- X1 and x2 are the weight_deltas between two agents within a layer.
- p is the value for the p-norm distance to calculate between each vector pair ∈[0, ∞]
- Once we have the Euclidean_Distance we find the mean of this distance, which signifies on an average how different are the means between two agents in a single layer.

Step 2: Ranking System

- Score the agents in each layer on how close or far away they are from other agents.
- Punishment metric used,
  - Each agent ranks the other agents based on how far they are with respect to it.
  - Closer one gets the better rank than the one that's far.
  - Ranks are divided into three groups, top 30%, average 40%, and bottom 30%
  - Punishment grows from top to bottom by group (remains same within the group).
- Let us see an example to understand it better, Consider 3 agents [x1,x2,x3]

  cdist(x1,x2) = 10 and cdist(x1,x3) = 5

  cdist(x2,x1) = 10 and cdist(x2,x3) = 3

  cdist(x3,x1) = 5 and cdist(x3,x2) = 3

  Here we can see that x1 and x2 both give higher rank to x3.

  x2 gets the higher rank only by x3.

  x1 gets the lowest rank by both x2 and x3.

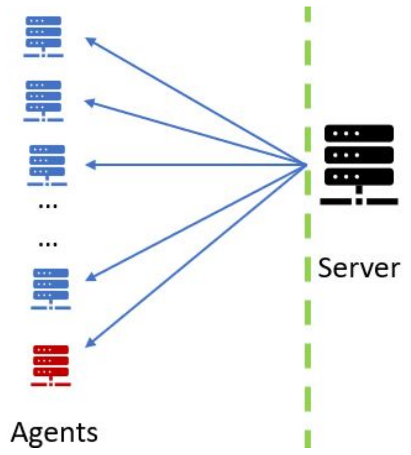
  Since x3 has the highest rank overall and falls in "top 30%" it gets least punished, x2 on the other hand fall in "average 40%" so gets slightly higher punishment and x1 for being the lowest rank gets heavily punished.

- Once the score is calculated for each agent across all the layers in a time-stamp.
    - Check if the score is less than a THRESHOLD = Mean+SD,

```
If Check is true,
    Tag agent as non-Malicious
Else,
    Tag Agent as Malicious
```

- mean+SD gave un understanding how big are the bigger score from the smaller ones, which helped to set the threshold.

Federated learning: Experimental Setup



We use PySyft to simulate federated learning environment. 10 virtual workers (or agents). Experimented with 4 different scenarios.

Benign/malicious agents split: 9/1, 8/2, 7/3

Training epochs per agent: 5. **Timestamps**: 12

Model details: CNN model with 2 Convolutional layers, each layer followed by max pooling and relu. Final activation function: log_softmax.

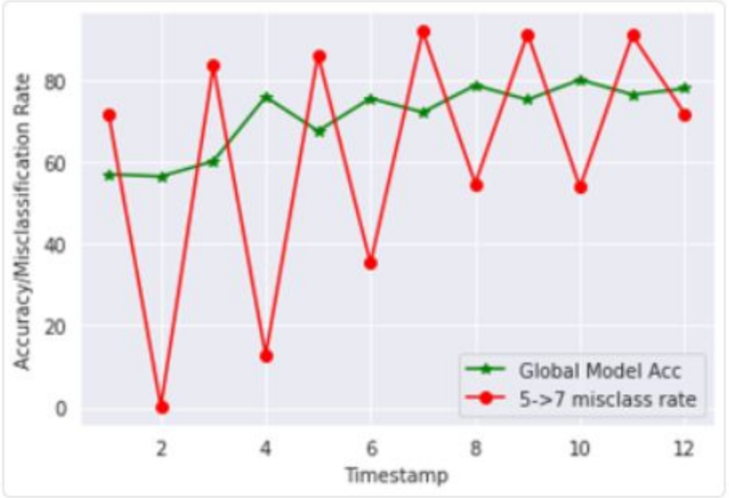
Weight delta boosting: Weight deltas of malicious agents are boosted by a factor of n, where n equals total number of examples at train dataset/total number of examples at malicious agent.

Datasets: Fashion-MINST split across all agents (with equal weights). Malicious agent's data modified for targeted attack. a). Change ALL Class 5 -> Class 7 (with shuffling in between timestamps). b). Change ONE SINGLE example from Class 5 -> Class 7 (no shuffling and all valid Class 5 examples removed)

Results: Scenario 1

Scenario:

- 1 malicious agent, 9 benign agents. All Class 5 examples being misclassified as Class 7 by the malicious agent.
- Boosting factor: 10
- Attack prevention **turned OFF**.
- Between timestamps the datasets are shuffled.
    - Malicious agent gets a new set of misclassified images and benign agents get a new set of *true* images.



Observations:

- Global model converging after 8 timestamps. Global model accuracy on Test Dataset hovers around 80%.
- Misclassification rate oscillates between odd/even timestamps. Reasoning: The weight deltas for malicious agent offsets effects of benign agent deltas in alternate timestamps.

$$Misclassification\ rate = \frac{number\ of\ misclassified\ Class\ 5 \rightarrow 7\ images}{total\ number\ of\ Class\ 5\ images} * 100$$

$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

NC STATE UNIVERSITY

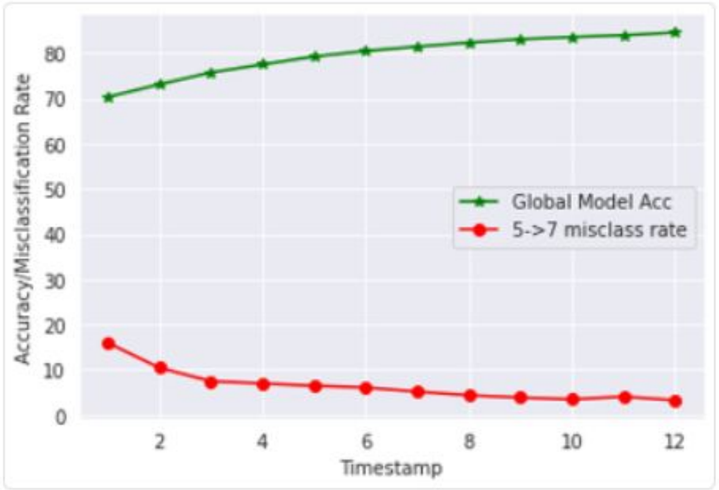
Results: Scenario 1

Scenario:

- 1 malicious agent, 9 benign agents. All Class 5 examples being misclassified as Class 7 by the malicious agent.
- Attack prevention **turned ON**. Boosting factor: 10
- Between timestamps the datasets are shuffled.
  - Malicious agent gets a new set of misclassified images and benign agents get a new set of *true* images.

Observations:

- Global model keeps improving rom timestamp 0. Goes upto 85% in 12 timestamps.
- Misclassification rate decreases with timestamps. This is because boosted weight deltas from malicious agent is ignored. Attack prevention works.

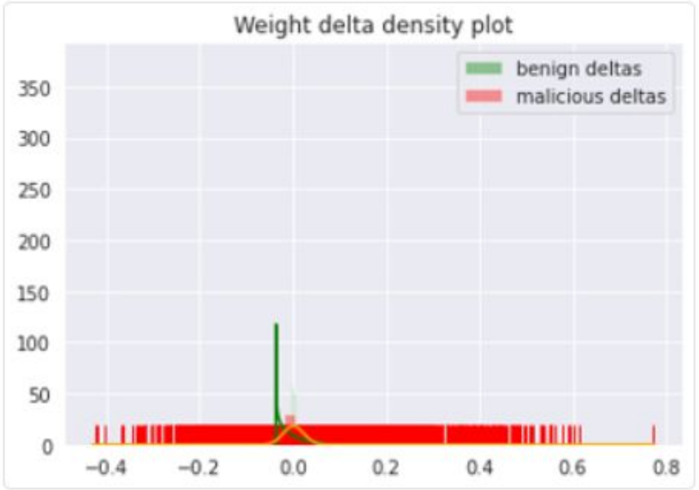


$$Misclassification\ rate = \frac{number\ of\ misclassified\ Class\ 5 \rightarrow 7\ images}{total\ number\ of\ Class\ 5\ images} * 100$$

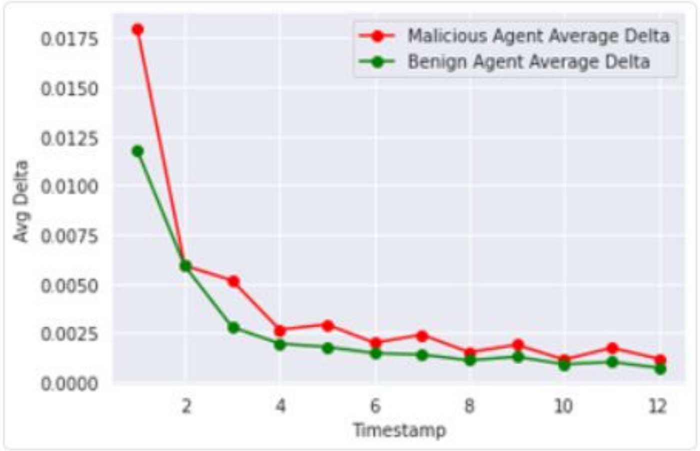
$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

NC STATE UNIVERSITY

Results: Scenario 1



Weight delta distribution of malicious and ONE benign agent



Average weight delta for benign and malicious agents

Observations:

- Benign agent weight delta distribution shown with *green* line. Malicious agent's weight delta shown with *orange* line.
- Spike seen with benign agent's weight delta. Malicious agent's deltas are more distributed. Reasoning: boosting factor of 10.

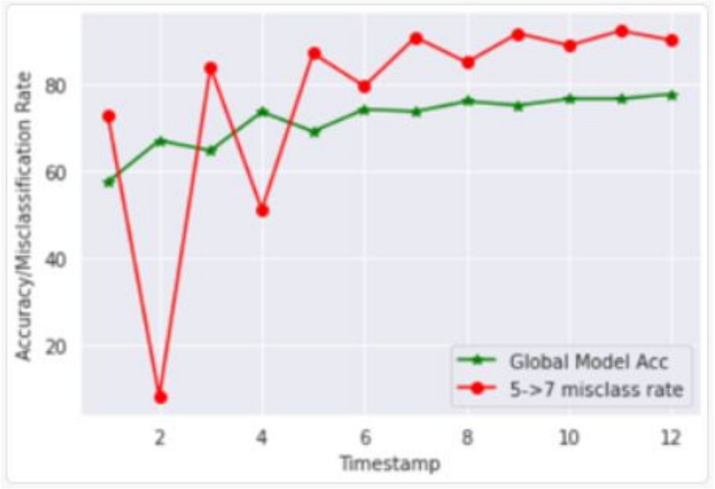
Results: Scenario 2

Scenario:

- 2 malicious agents, 8 benign agents. All Class 5 examples being misclassified as Class 7 by the malicious agent.
- Boosting factor: 5 for each malicious agent.
- Attack prevention **turned OFF**.
- Between timestamps the datasets are shuffled.
  - Malicious agent gets a new set of misclassified images and benign agents get a new set of *true* images.

Observations:

- Global model converging after 8 timestamps. Global model accuracy on Test Dataset hovers around 78-80%.
- Misclassification rate oscillates between odd/even timestamps but stabilizes and stays above 80% after 5 timestamps.



$$Misclassification\ rate = \frac{number\ of\ misclassified\ Class\ 5 \to 7\ images}{total\ number\ of\ Class\ 5\ images} * 100$$

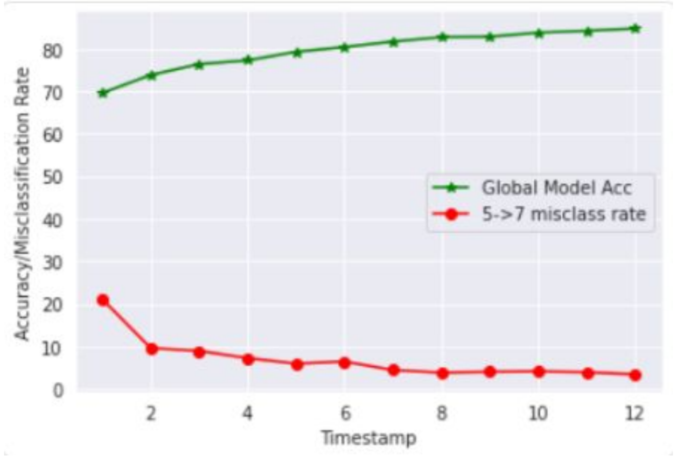
$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

NC STATE UNIVERSITY

Results: Scenario 2

Scenario:

- 2 malicious agents, 8 benign agents. All Class 5 examples being misclassified as Class 7 by the malicious agent.
- Attack prevention **turned ON**. Boosting factor 5 (per mal agent)
- Between timestamps the datasets are shuffled.
    - Malicious agent gets a new set of misclassified images and benign agents get a new set of *true* images.



$$Misclassification\ rate = \frac{number\ of\ misclassified\ Class\ 5 \rightarrow 7\ images}{total\ number\ of\ Class\ 5\ images} * 100$$

$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

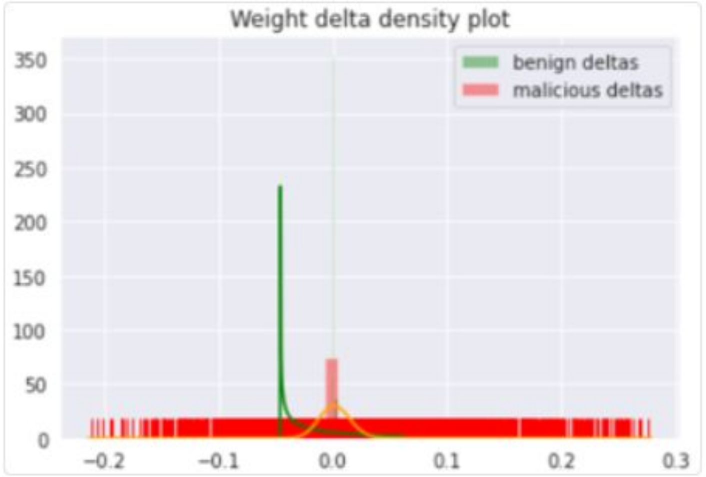
Observations:

- Global model accuracy keeps improving and goes upto 85% in 12 timestamps.
- Misclassification rate decreases continuously and reaches less than 5% after 12 timestamps. Attack prevention works.

Results: Scenario 2

Observations:

- Benign agent weight delta distribution shown with *green* line. Malicious agent's weight delta shown with *orange* line.
- Spike seen with benign agent's weight delta. Malicious agent's deltas are more distributed. Reasoning: boosting factor of 5 (per malicious agent).



$Weight\ delta\ distribution\ of\ malicious\ and$

$ONE\ benign\ agent$

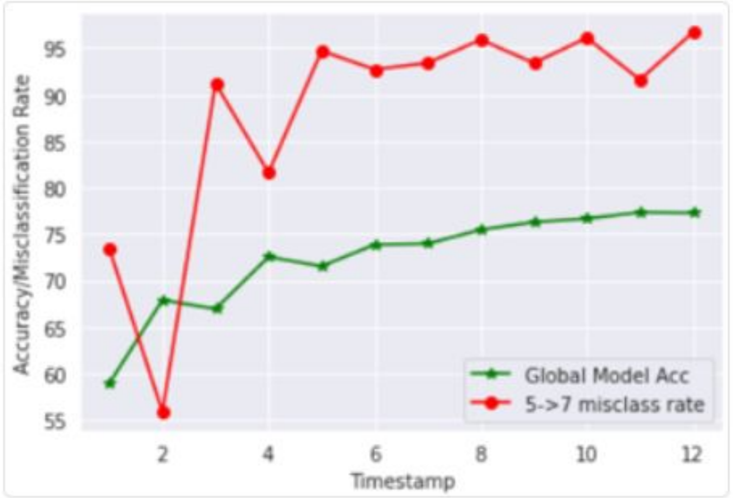
Results: Scenario 3

Scenario:

- 3 malicious agents, 7 benign agents. All Class 5 examples being misclassified as Class 7 by the malicious agent.
- Boosting factor: 3 for each malicious agent.
- Attack prevention **turned OFF**.
- Between timestamps the datasets are shuffled.
  - Malicious agent gets a new set of misclassified images and benign agents get a new set of *true* images.

Observations:

- Global model accuracy keeps increasing and reaches 75% after 12 timestamps. Overall its less than other setups.
- Misclassification rate oscillates initially then stabilizes and stays in high 90's.



$$Misclassification\ rate = \frac{number\ of\ misclassified\ Class\ 5 \rightarrow 7\ images}{total\ number\ of\ Class\ 5\ images} * 100$$

$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

NC STATE UNIVERSITY

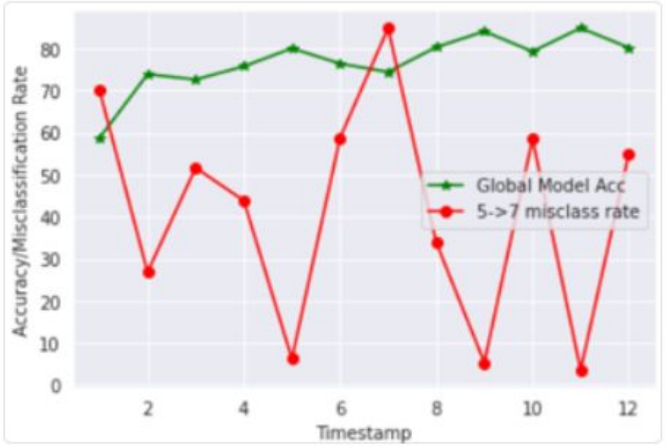
Results: Scenario 3

Scenario:

- 3 malicious agents, 7 benign agents. All Class 5 examples being misclassified as Class 7 by the malicious agent.
- Attack prevention **turned ON**. Boosting factor 3 (per mal agent)
- Between timestamps the datasets are shuffled.

Observations:

- Global model accuracy keeps improving and goes upto 80% in 12 timestamps.
- Even with attack prevention turned on the attack detection algorithm is not identifying malicious weight deltas for all timeframes. E.g. in timestamp 7 the misclassification rate goes beyond 80% and it should be less than 10% when compared with other scenarios



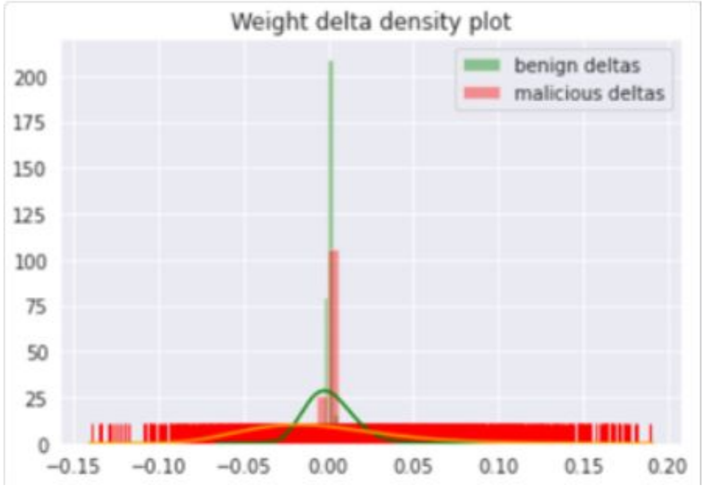
$$Misclassification\ rate = \frac{number\ of\ misclassified\ Class\ 5 \rightarrow 7\ images}{total\ number\ of\ Class\ 5\ images} * 100$$

$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

Results: Scenario 3

Observations:

- Benign agent weight delta distribution shown with *green* line. Malicious agent's weight delta shown with *orange* line.
- Spike seen with benign agent's weight delta. Malicious agent's deltas are more distributed. Reasoning: boosting factor of 5 (per malicious agent).



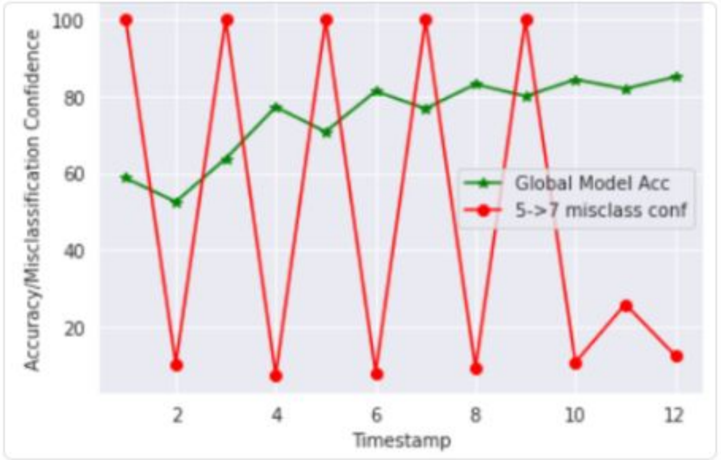
Weight delta distribution of malicious and ONE benign agent

NC STATE UNIVERSITY

Results: Scenario 4

Scenario:

- 1 malicious agents, 10 benign agents. ONE SINGLE example targeted to be misclassified.
- Boosting factor: 10 for malicious agent.
- Attack prevention **turned OFF**.
- Between timestamps the datasets are NOT shuffled.
    - The malicious agent keeps sending updated weight deltas for the same targeted image.



$$Misclassification\ rate(confidence) = \begin{cases} 100, if\ global\ model\ predicts\ Class\ 5 \rightarrow 7 \\ Probability\ of\ predicting\ class\ 7 * 100 \end{cases}$$

$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

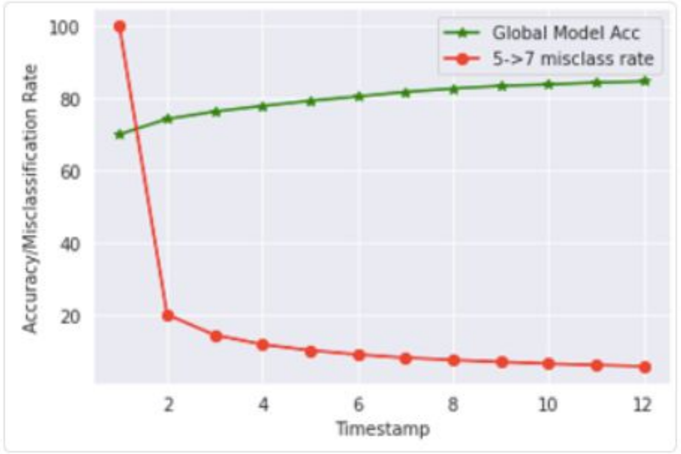
Observations:

- The malicious agent is able to misclassify the targeted image by global model half of the time. In the other half updates from bening agents prevents misclassification.
- Global model accuracy on test data keeps increasing and reaches 85%. It starts to converge around timestamp 8.

Results: Scenario 4

Scenario:

- 1 malicious agents, 10 benign agents. ONE SINGLE example targeted to be misclassified.
- Boosting factor: 10 for malicious agent.
- Attack prevention **turned ON**.
- Between timestamps the datasets are NOT shuffled.
  - The malicious agent keeps sending updated weight deltas for the same targeted image.



Observations:

- Attack prevention works and global model misclassification confidence keeps decreasing on the SINGLE targeted image.
- Global model accuracy on test data set keeps increasing and reaches 85% in 12 timestamps.

$$Misclassification\ rate(confidence) = \begin{cases} 100, if\ global\ model\ predicts\ Class\ 5 \rightarrow 7 \\ Probability\ of\ predicting\ class\ 7 * 100 \end{cases}$$

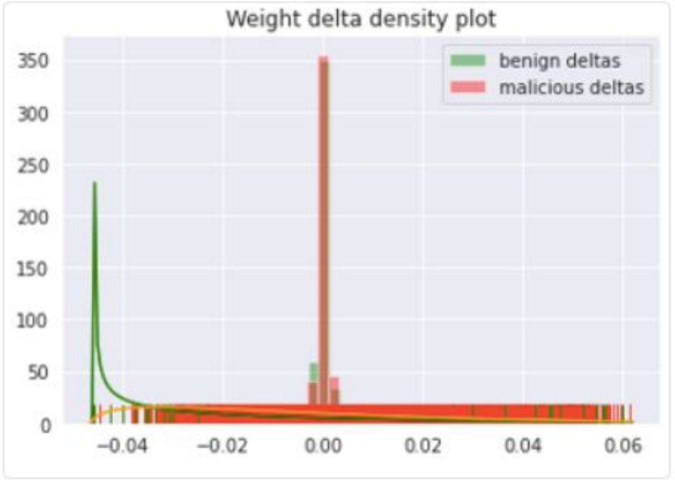
$$Gloabl\ model\ accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ images\ in\ test\ dataset} * 100$$

NC STATE UNIVERSITY

Results: Scenario 4

Observations:

- Benign agent weight delta distribution shown with *green* line. Malicious agent's weight delta shown with *orange* line.
- Weight delta distribution is quite different for malicious and (one random) benign agent.



Weight delta distribution of malicious and ONE benign agent

NC STATE UNIVERSITY

Conclusions

- We were able to simulate federated learning with attack prevention by using PySyft’s libraries.
- We were able to successfully meddle with the global model and force it to misclassify a whole class of examples and a single example while still maintaining good accuracy levels of the global model on the test dataset that is not seen by any agent.
- We were also able to detect malicious updates and prevent them from affecting global model. Attack prevention ensured that the global model has a higher image classification accuracy.

Future Work:

- Execute all scenarios with more timestamps and assess performance.
- Experiment with other datasets and models.

NC STATE UNIVERSITY