pFedDef: Defending Grey-Box Attacks for Personalized Federated Learning

Summary:

In today's data-driven landscape, the delicate equilibrium between safeguarding user privacy and unleashing data potential stands as a paramount concern. Federated learning, which enables collaborative model training without necessitating data sharing, has emerged as a privacy-centric solution. Nonetheless, this decentralized approach brings forth security challenges, notably poisoning attacks where malicious entities inject corrupted data. Our research, initially spurred by test-time evasion attacks, investigates the intersection of adversarial training and poisoning attacks within federated learning, introducing Adversarial Robustness Unhardening (ARU). ARU is employed by a subset of adversaries to intentionally undermine model robustness during decentralized training, rendering models susceptible to a broader range of evasion attacks. We present extensive empirical experiments evaluating ARU's impact on adversarial training and existing robust aggregation defenses against poisoning and backdoor attacks. Our findings inform strategies for enhancing ARU to counter current defensive measures and highlight the limitations of existing defenses, offering insights into bolstering defenses against ARU.

Requirements

To install requirements:

```
pip install -r requirements.txt
```

All experiments require NVIDIA GPU and CUDA library. All experiments of this paper are run on the Amazon EC2 instance G4DN.XLARGE.

Data Sets

The CIFAR-10 and CIFAR-100 data sets are readily available to set up with the existing infrastructure of FedEM. The CelebA data set can be found and downloaded at LEAF, while the MovieLens data set is found at MovieLens.

Original Repository

The code from this repository has been heavily adapted from: Federated Multi-Task Learning under a Mixture of Distributions. The code is found at https://github.com/omarfoq/FedEM.

The following components of their work has been used as a basis for our work.

- data folder with the data set downloading and data splitting
- learner folder with learner and learners_ensemble classes
- 'client.py', 'aggregator.py' classes

Training

The model weights can be trained given varying methods of adversarial training and aggregation method by running the .py files in the 'run_experiments_collection' folder. Move the .py file to the root directory and follow the instructions to run each experiment.

The scripts have been written in such a way that an individual script can be run for consecutive training of related neural networks. All experiments require NVIDIA GPU and CUDA library, and has been run on AWS EC2 g4dn.xlarge instance.

Evaluation

The Notebooks in the Evaluations folder provides jupyter could be used for evaluation.