Enhancing IoT Anomaly Detection Performance for Federated Learning

Brett Weinger[1], Alex Sim (advisor)[2], John Wu (advisor)[2], Jinoh Kim (advisor)[2]
[1]Stony Brook University, [2]Lawrence Berkeley National Laboratory

ABSTRACT

As security threats to IoT devices become more common, machine learning provides a mechanism to discover and mitigate possible intrusions. Federated learning is a decentralized form of machine learning where training happens on-device, allowing participating IoT devices to maintain privacy of data while contributing to an optimal aggregate model. In this work, we study supervised federated learning for recently published IoT sensor readings and experiment with data augmentation to overcome the class imbalance issue and achieve higher performance metrics, ultimately leading to more frequent and reliable anomaly detection.

BACKGROUND

- The UNSW TON_IoT datasets contain data from IoT sensors intended for cybersecurity AI applications.
- The dataset includes sensory readings for various IoT devices under normal circumstances as well as hacking attacks (i.e. DDoS, backdoor, poisoning).
- Federated learning (FL) is applied to avoid transmitting sensitive data.

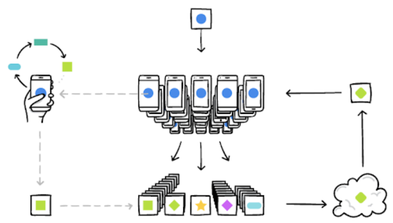

Fig 1: Federated learning framework for cloud devices
Source: Google AI Blog

- FL uses a client-server architecture to collaboratively train a model over a series of rounds.
- Steps of a given training round:

  - Server sends current model to each client
  - Client trains model and computes gradient update using its own data
  - Updates are sent back and aggregated on server (*FedAvg* algorithm)

- After converging, a learned model can allow clients to identify types attacks that they have not been directly exposed to.

RESEARCH QUESTION

How can data augmentation be used to improve naïve federated learning for IoT anomaly detection with imbalanced class datasets?

METHODS

- Supervised binary classifier neural networks are trained using fixed parameters for each trial (2 hidden layers, 128 batch size, 10 local epochs, 0.1 initial learning rate).
- IoT Modbus features binned using *bin_size* of 1,000 to increase feature space and create sparse input data.
- Naïve FL first performed using 5, 50, and 100 nodes to establish baseline metrics.
- Trials are then conducted using statistical and nearest neighbor augmentation strategies:

  - Random oversampling (RAND)
  - Synthetic Minority Oversampling Technique (SMOTE)
  - Adaptive Synthetic Sampling Approach (ADASYN)

Naïve Models

Training naïve federated models without any augmentation results in F1 score stalling at zero for initial rounds:

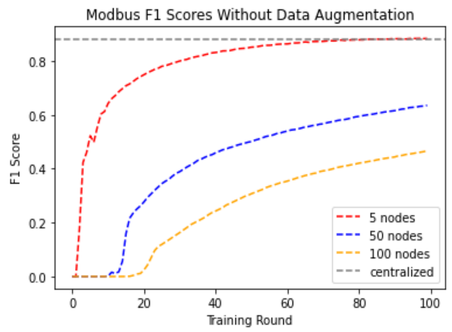

Fig 2: F1 scores for naïve FL Modbus classification

Augmentation is used to raise initial F1 score and compare performance after further training.

RESULTS

- An F1 score threshold of 70.00% is used as an acceptable performance level to compare data augmentation strategies and baseline models:

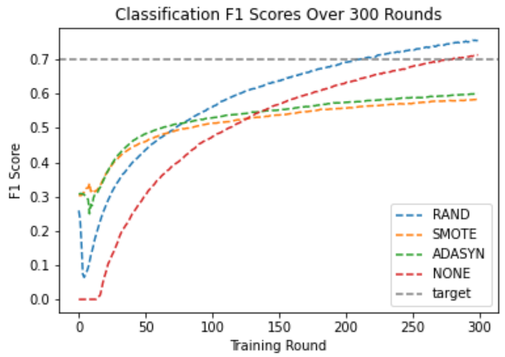

Fig 3: F1 scores using various augmentation strategies (100 nodes)

- Baseline model took 280 rounds to reach threshold, RAND approach took 211 rounds (24.6% decrease).
- Trials conducted for 5 and 50 nodes conducted as well, but advantage of oversampling most noticeable when learning is increasingly distributed.
- Data for 100 nodes and 100 rounds:

Method	Accuracy	F1 Score	Avg Round Time
RAND	**83.49%**	**55.96%**	37.09 s
SMOTE	72.22%	52.09%	36.87 s
ADASYN	73.34%	52.77%	36.63 s
NONE	81.87%	46.64%	**34.68 s**

Table 1: Comparing short-term augmentation method results. RAND remains advantageous with minor computational overhead.

- Additionally, *device heterogeneity* is investigated where amount of data is nonuniform across clients:

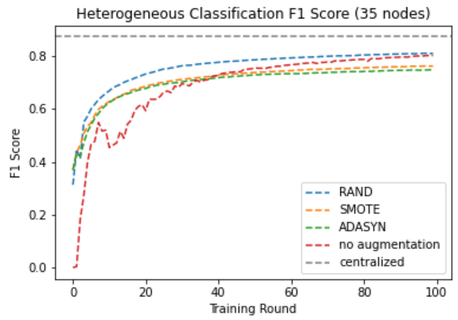

Fig 4: F1 scores for heterogeneous data quantities obtained by Gaussian sampling. Due to weighted federated averaging, these models outperform their homogeneously-trained counterparts.

CONCLUSION

- Randomly augmenting minority class data samples can provide performance boosts to FL systems where minimizing number of training rounds is a priority.
- Machine learning augmentation methods either degrade long-term performance (SMOTE, ADASYN) or add infeasible computational costs for mobile devices (i.e. GANs, which are not used in this study).
- With highly decentralized (large number of nodes) and homogeneous (no node has most of data) training, advantage of augmentation is most significant.

FURTHER READING

Scan the above code to view a slide deck containing further information about this work.

ACKNOWLEDGMENTS

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internship (SULI) program. This work was also supported by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231, and also used resources of the National Energy Research Scientific Computing Center (NERSC).