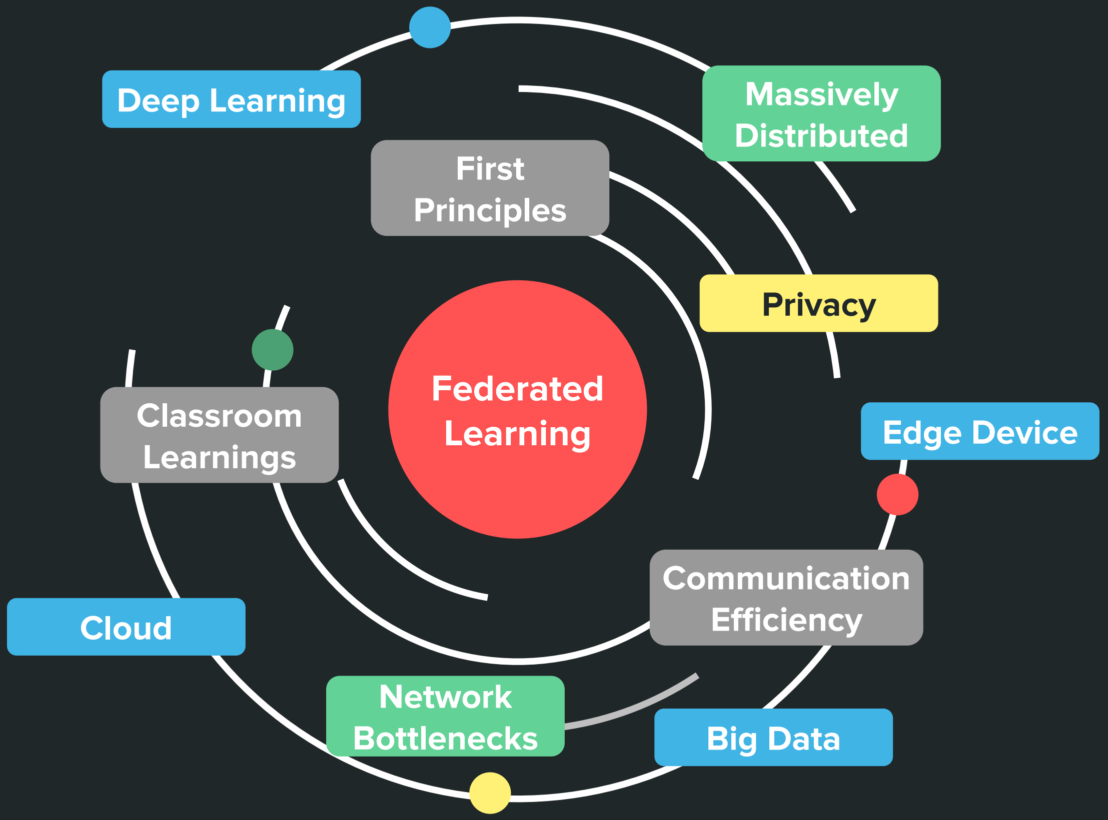
Intrusion Detection Using Federated Learning

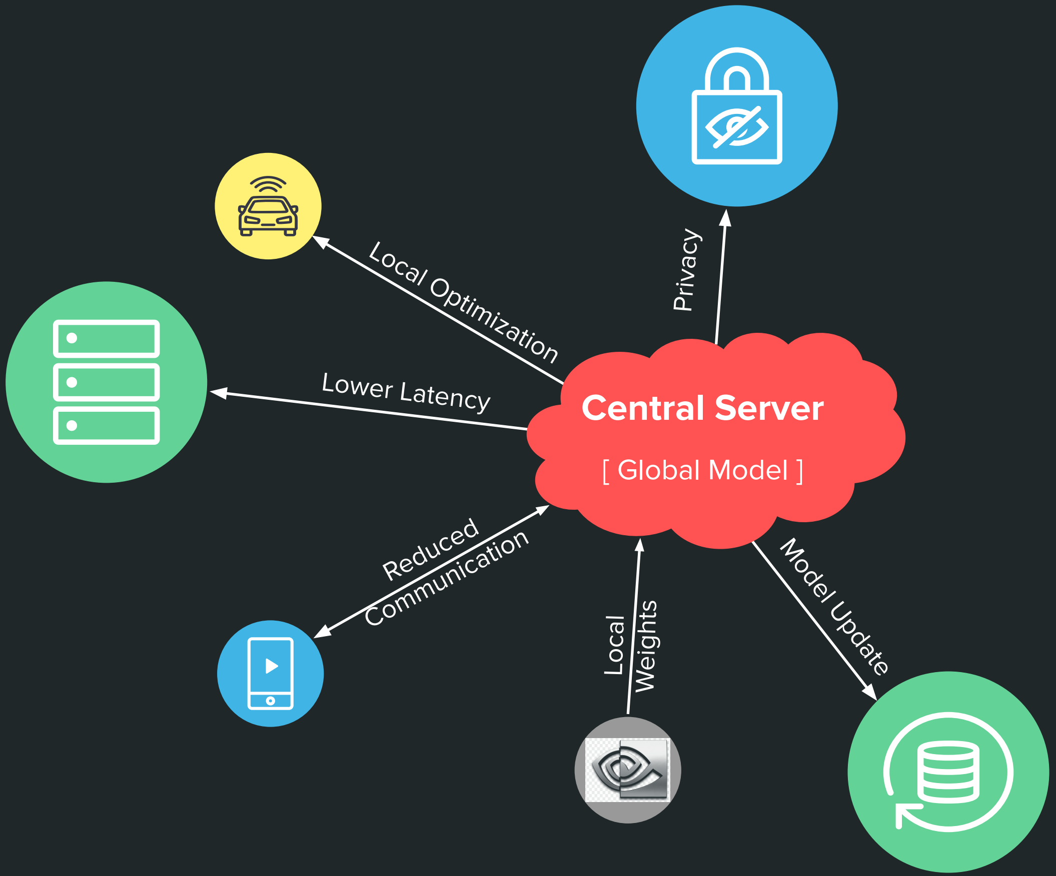
Dhileeban Kumaresan | Jocelyn Lu | Nitin Pillai | Riyaz Kasmani

12th April, 2021

Project Objective



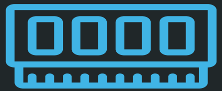
Why Federated Learning?



- ★ Privacy
- ★ Lower Latency
- ★ Decentralized Learning
- ★ Reduced Bottlenecks

Why Federated Learning?

Trainer 1

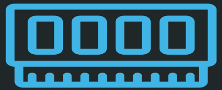


Benign

Benign

Botnet

Trainer 2

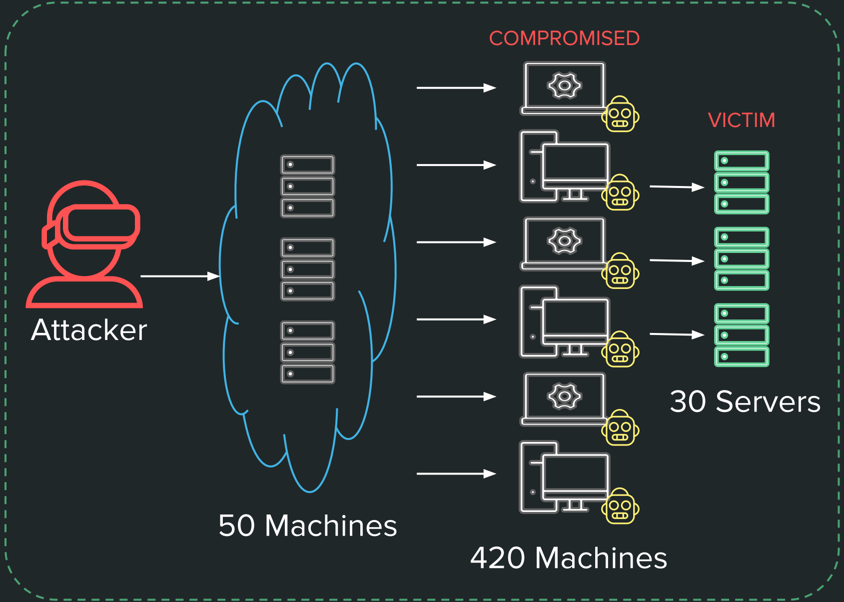


Benign

Benign

Brute-force

Data Characteristics



CSE-CIC-IDS2018** - Network Traffic Data

- ★ 16.2 M Instances / 10 Days
- ★ 17% Attack Traffic
- ★ 10 CSV files / 6.41 GB
- ★ HTTPS, HTTP, SMTP, POP3, IMAP, SSH, and FTP
- ★ Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and Inside Infiltration

Non IID | Unbalanced | Massively Distributed

** - Communications Security Establishment (CSE) and Canadian Institute of CyberSecurity (CIC)

Data Cleansing

Step #1
Load Data

10 CSV Files /
6.41 GB

Step #2
Distinguish
Attributes

79 Independent
Features, 1 Label**

Step #3
Assess &
Cleanse Data

Approx. 20K
Rows Dropped

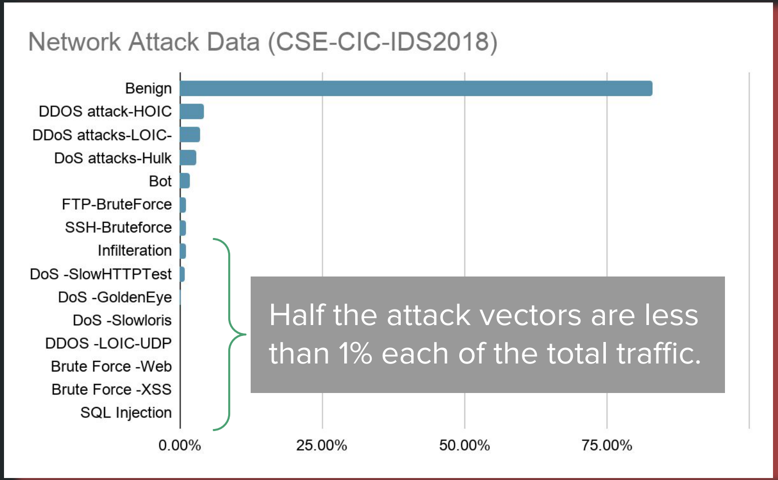
DROPPED ROWS / COLUMNS

- ★ Infinity / NaN Values
- ★ Repeat Headers
- ★ 4 Extraneous Attributes
- ★ Timestamp

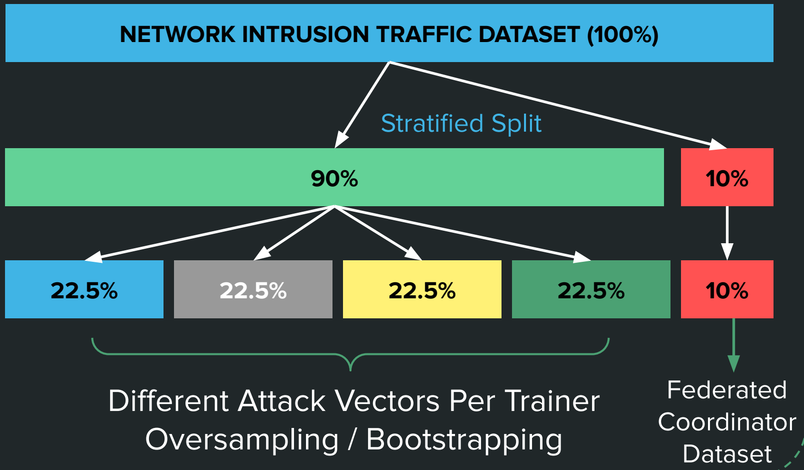
** - Once CSV file had 83 independent features i.e. 4 extra attributes

EDA & Test Data Setup

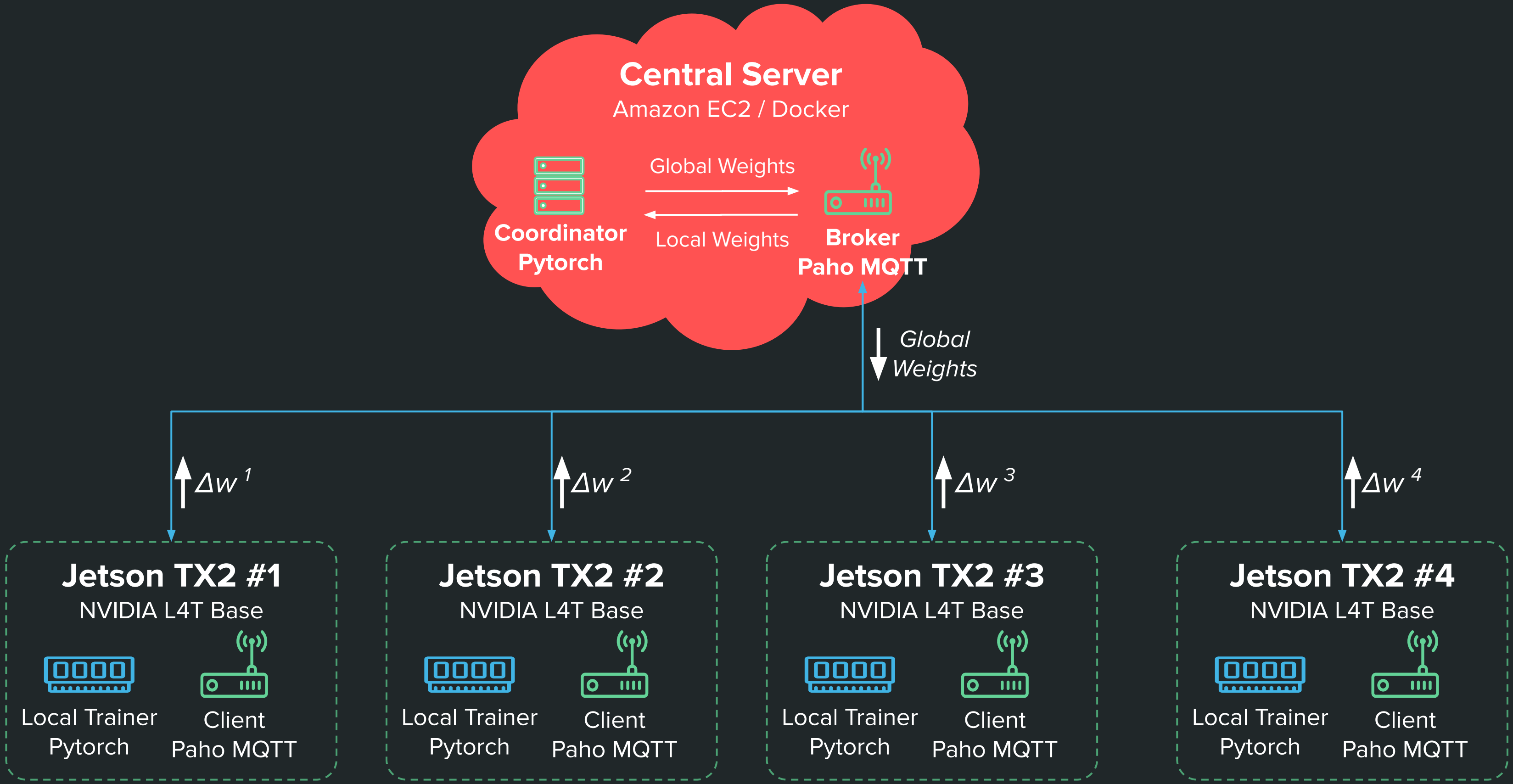
NETWORK INTRUSION TRAFFIC DATASET



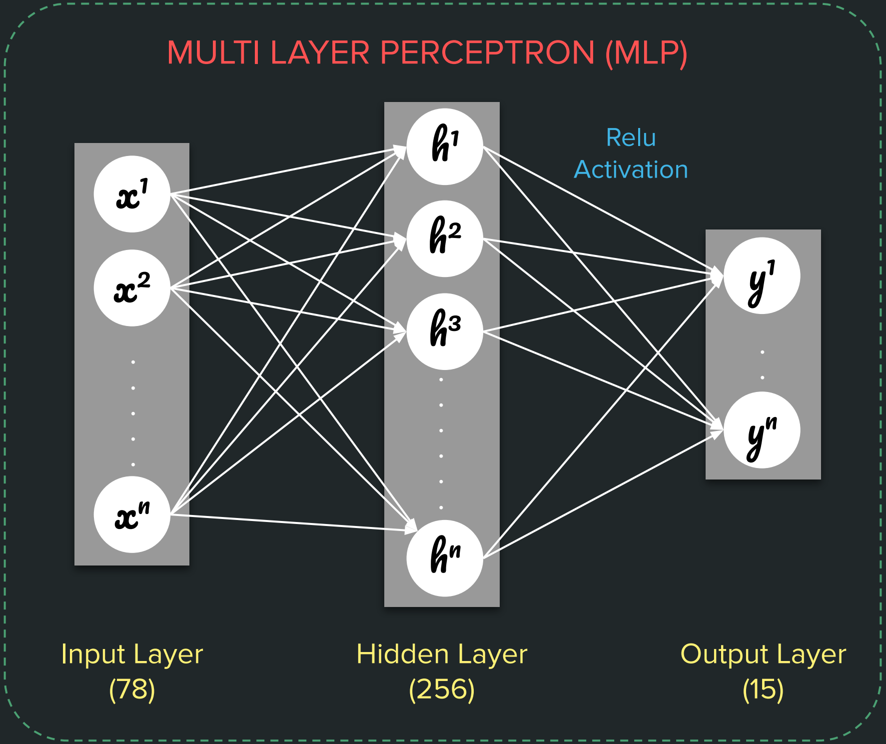
TRAINING / TEST DATA SETUP



System Architecture



Model Architecture



- ★ Batch Size: 5000
- ★ Epochs: 5
- ★ Learning Rate: 0.001
- ★ Optimizer: Adam

Federated Averaging Algorithm

Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

Server executes:

  initialize w_0

  for each round $t = 1, 2, \dots$ **do**

    $m \leftarrow \max(C \cdot K, 1)$

    $S_t \leftarrow$ (random set of m clients)

    for each client $k \in S_t$ **in parallel do**

      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$

    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate(k, w): *// Run on client k*

  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B)

  for each local epoch i from 1 to E **do**

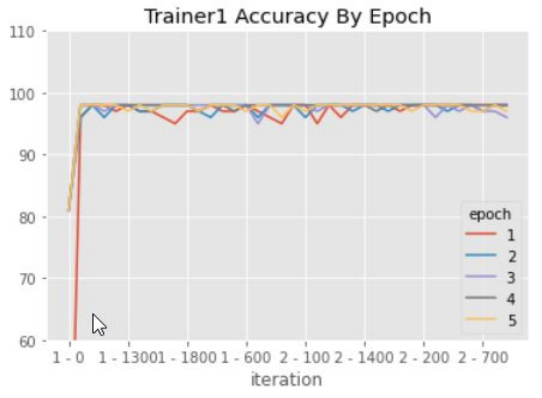
    for batch $b \in \mathcal{B}$ **do**

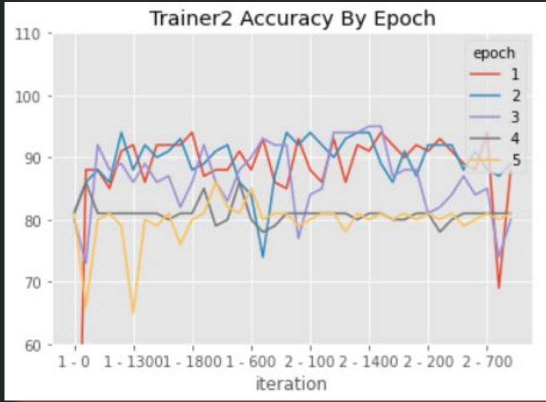
      $w \leftarrow w - \eta \nabla \ell(w; b)$

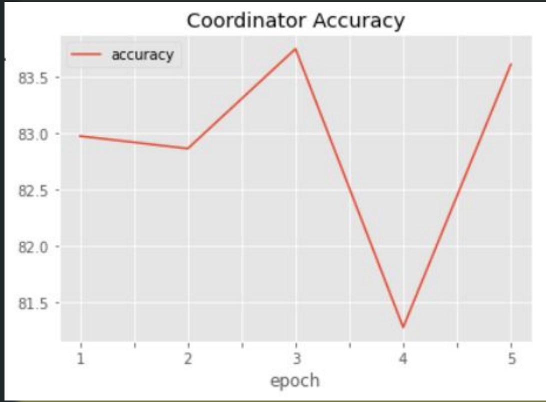
  return w to server

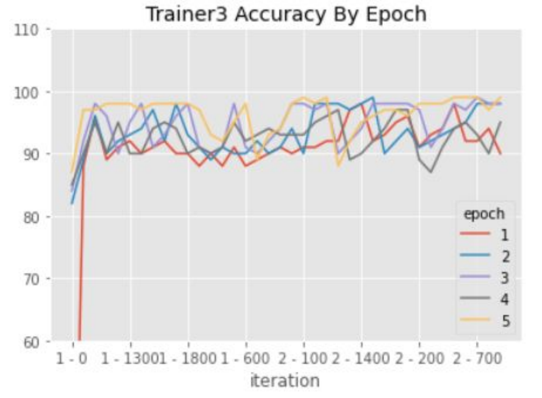
Federated Learning in Action (Demo)

Experimental Results

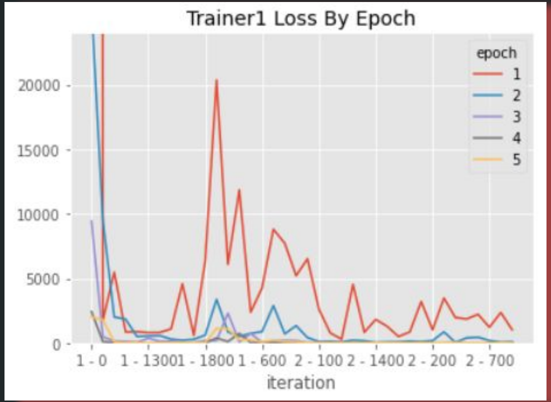


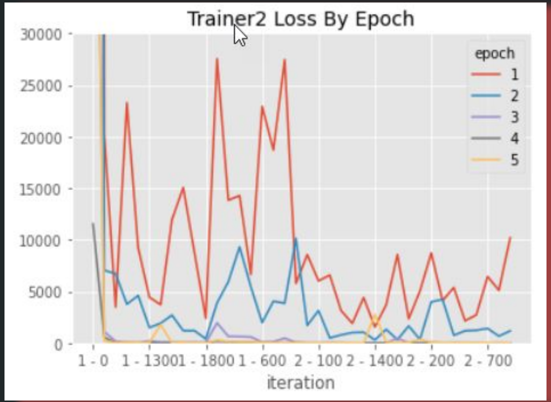


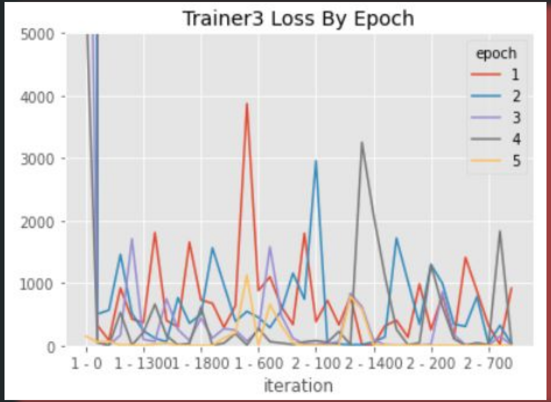


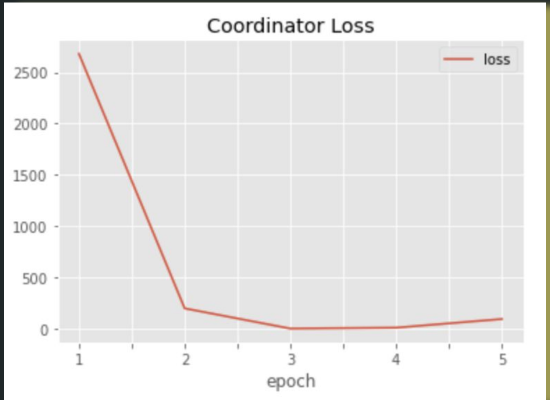


Experimental Results









Learnings

★ Random Independent vs Common Weights Initialization

★ More rounds of weights updates for non-iid data

★ Another hyperparameter to tune - when to pass back the weights

Future Work

★ Asynchronous Setup / Fault Tolerance

★ Share information about the distribution of data the trainers have

★ Weighted average instead of simple average

Thank you!

References

- Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv:1602.05629v3
- Deep Learning Algorithms for Cybersecurity Applications. ScienceDirect
- Case Study on Using Deep Learning for Network Intrusion Detection. arXiv:1910.02203v1
- Autonomous Intrusion Detection System Using an Ensemble of Advanced Learners. arXiv:2001.11936v2
- Survey and Analysis of Intrusion Detection Models based on CSE-CIC-IDS2018. J Big Data 7, 104 (2020)
- AI-IDS: Application of Deep Learning to Real-time Web Intrusion Detection. doi: 10.1109/ACCESS.2020.2986882
- Using Deep Learning Techniques for Network Intrusion Detection. doi: 10.1109/ICIoT48696.2020.9089524
- Hybrid Model for Intrusion Detection Systems. arXiv:2003.08585v1
- Machine Learning and Deep Learning Methods for Intrusion Detection Systems. doi: 10.3390/app9204396
- Deep Learning Approach for Intelligent Intrusion Detection System. doi: 10.1109/ACCESS.2019.2895334