Federated AI

Smart Contract driven Fair and transparent reward mechanism

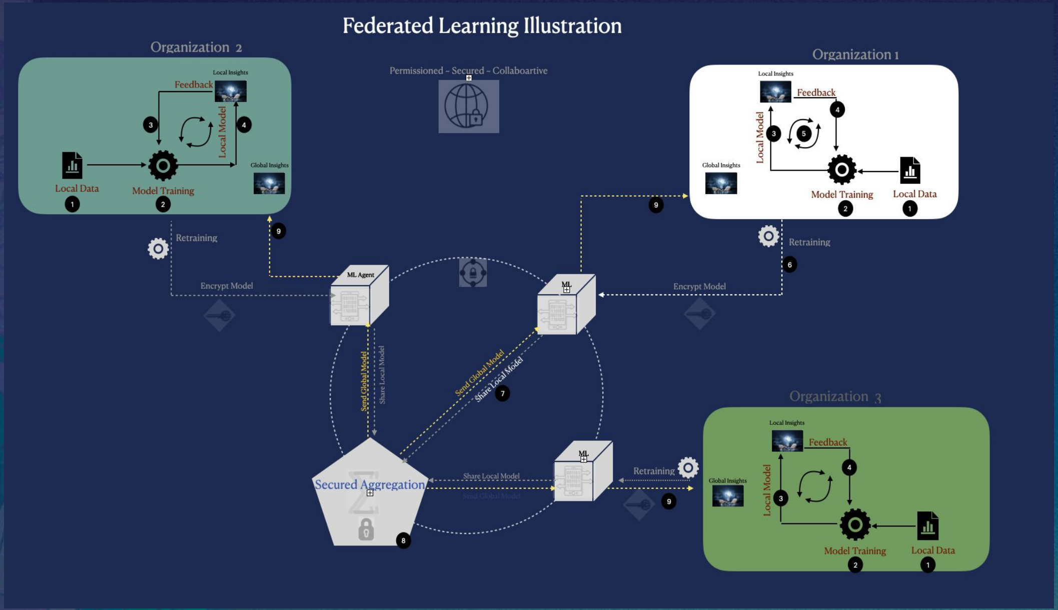
Speaker (s)
Sudhir Upadhyay and Monik R Behera

Section 1

Challenge

Lack of a fair, verifiable, and proportionate reward mechanism in Federated AI

https://arxiv.org/pdf/2107.10243



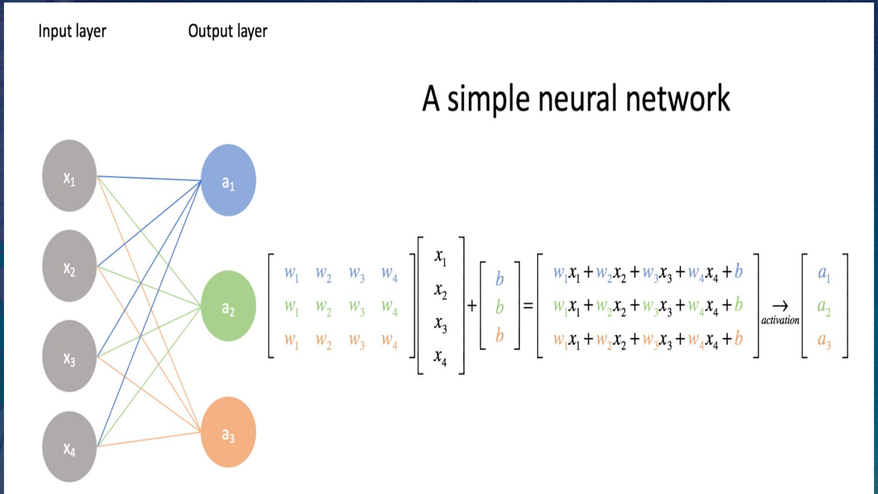
Federated AI - A Primer

- Enables multiple parties to contribute towards a global machine learning model.
- Contributors only share their locally trained model, not the data.

Solution

- Smart contract-based contribution analysis framework for federated learning
- Design and develop techniques for Federated contribution
- Enhance Security of private messaging between contributor and the aggregator using dynamic key generation for each FL round
- (Future) Leverage ethereum to build the economics of on-chain reward (and penalty)

Visual Representation



Source: https://www.jeremyjordan.me/intro-to-neural-networks/

Machine Learning Model

Machine learning models have **weight matrices**, calculated from loss function and gradient descent approach (standard ML terminology)

Mathematical Illustration

$$\gamma^k := \|\beta^k\|_2$$

$$\beta^k := \langle \|\delta_1^k\|_{\mathbf{F}}, \quad \|\delta_2^k\|_{\mathbf{F}}, \quad \dots \quad , \|\delta_L^k\|_{\mathbf{F}} \rangle$$

$$\delta_l^k := \mathbf{w}_{\mathbf{l,t}}^{\mathbf{global}} - \mathbf{w}_{\mathbf{l,t+1}}^{\mathbf{k}}$$

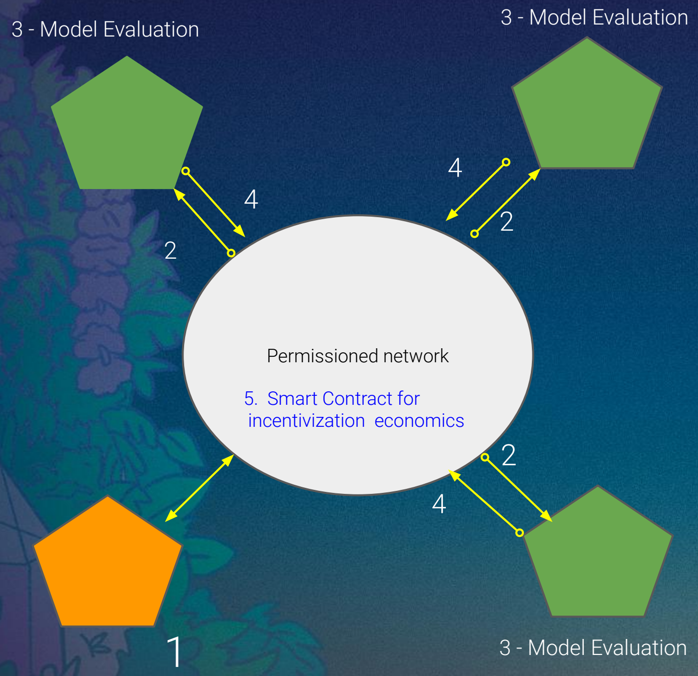
$$\gamma_{rel}^k := \frac{\gamma^k}{\sum_{k=1}^{K} \gamma^k}$$

where γ^k is the absolute federated contribution of client k, $\|.\|_p$ represents p^{th} norm, $\|.\|_F$ represents Frobenius norm[32], L represents the final layer's weight matrix of a generic machine learning model. δ_l^k represents difference of model weight parameter matrix for l^{th} layer of k^{th} client. $\mathbf{w}_{\mathbf{l,t+1}}^{\mathbf{k}}$ represents model weight for l^{th} layer of k^{th} client at $t+1^{th}$ iteration. $\mathbf{w}_{\mathbf{l,t}}^{\mathbf{global}}$ is the model weight for l^{th} layer of global model at t^{th} iteration. γ_{rel}^k is relative federated contribution of client k.

Federated Contribution

1. Federated computation is a scalar quantity, which depicts the deviation, or divergence of two machine learning models.
2. Leveraging Frobenius norm that calculates the deviation between two weight matrices (current and previous model)

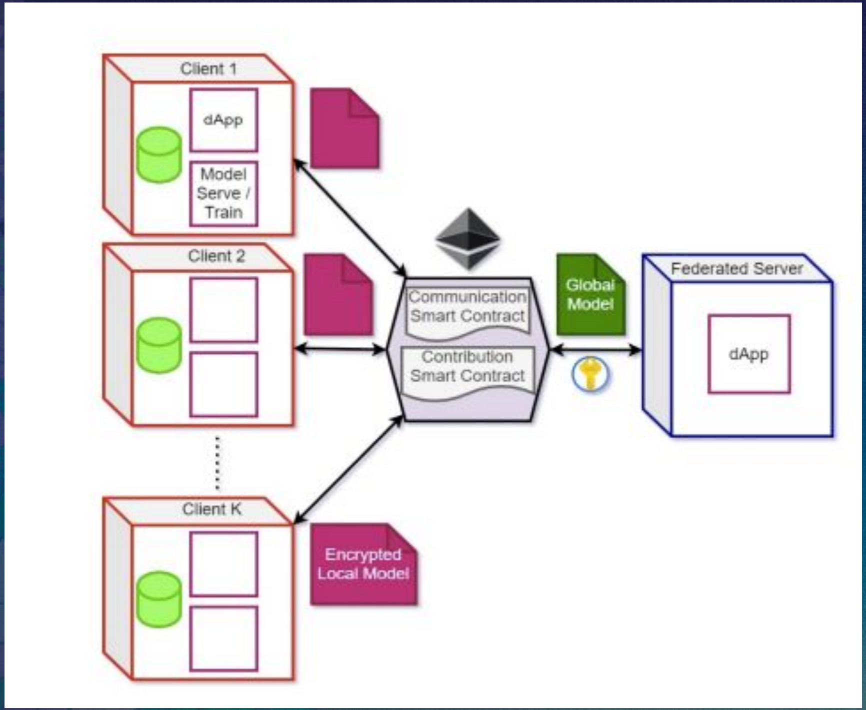
Visual Illustration



Federated Contribution

1. After each round of aggregation, Federated Contribution is recorded on chain
2. Each of the clients receive that contribution
3. Every client does the evaluation against the aggregated model
4. Each client then submits their evaluation back to chain.
5. Smart contract **can** combine the results of federated contribution and each client's evaluation results to generate the incentive mechanism

Blockchain for Recording Contributions



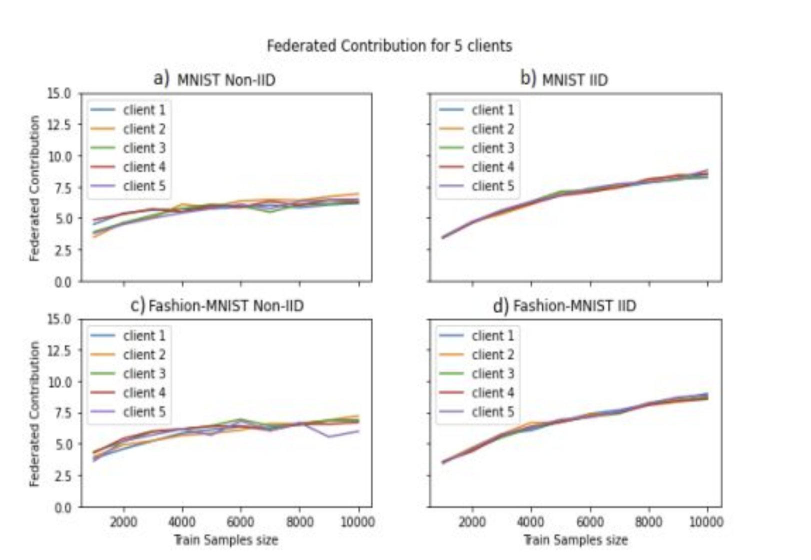

```
// Communication : Interface
pragma solidity >=0.8.0 <0.9.0;
contract Communication {
  event BCEvent(
    uint256 timestamp,
    bool is_encrypted,
    bytes event_type,
    bytes body
  );
  function publish(
    uint256 timestamp,
    bool is_encrypted,
    bytes memory event_type,
    bytes memory body
  ) public returns(uint ack) { }
}

// Contribution : Interface
pragma solidity >=0.8.0 <0.9.0;
contract Contribution {
  uint len = 5; //5 federated clients
  uint[] memory _clients = new uint[](5);
  function set_contribution(
    uint client_id,
    uint relative_contribution
  ) public returns(uint ack) {
  //only owner(federated server)
  //modifies state of _clients
  }
  function get_contributions()
  public view returns (uint memory) { } }
```

Questions ?

And some answers… may be -:)😀

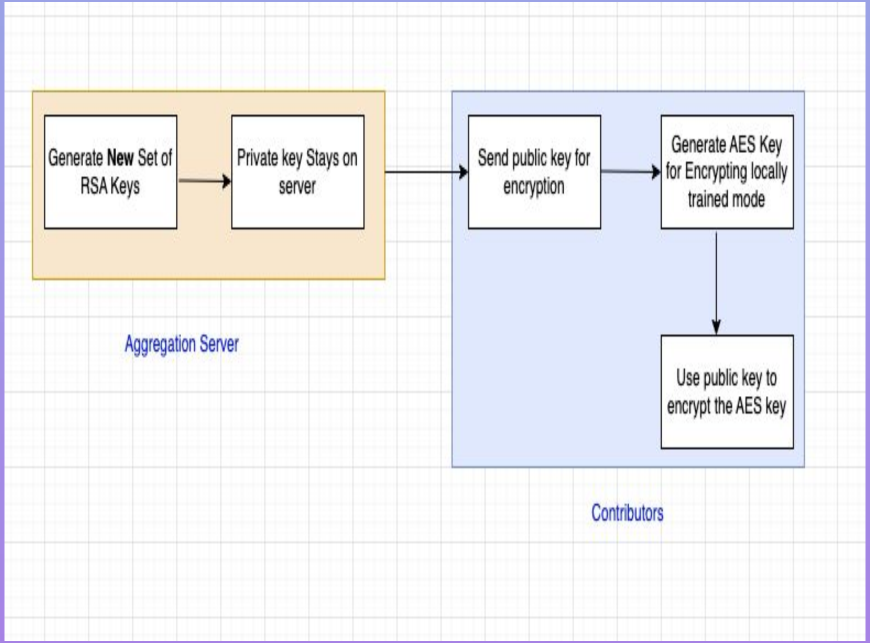
Appendix

Results using Public Dataset



- Federated contribution being increasing with increase in training data sample size.
- Increase in federated contribution value with increase in training size
- This essentially validates our hypothesis of federated contribution being dependent on number of weight updates, which is directly proportional to higher training data (or training iterations)

Each Federated Learning Cycle



Enhanced Security Implementation

1. Leverage symmetric encryption using RSA cryptography
2. Each new federated learning round is published as blockchain event, federated aggregation server generates a new set of RSA key pair.
3. The private key of the pair stays with the server
4. The public key is sent across the network to all the participants.
5. Participant will generate an AES key for encrypting local train model, and then use the public key received from federated server to encrypt the AES key[28].
6. With RSA keys revolving for every new federated learning round, this decreases the chances of compromising model information over the blockchain, for any given participant on the network

Why on Ethereum ?

1. (Potentially) Perform Aggregation on L2
2. Record contributions on L1
3. Record the results of model evaluation on L1
4. Leverage Ethereum token economics for Reward Model