Blockchain in Federated Learning (Phase 3,4)

Team Member 1:

First Name: Arjun Ramesh

Last Name: Kaushik

Email: kaushik3@buffalo.edu

Person Number: 50413327

Approved by: Prof. Bina Ramamurthy and Chen Xu

Reference Papers:

1. https://drive.google.com/file/d/1tbIfXJRVTmbeQYT6QmckV_Yq_u1rkow4/view?usp=sharing
2. https://drive.google.com/file/d/1BuzRDZioCtsEgFR9L0FLIYoCveJlaYY6/view?usp=sharing

Issue(s) addressed: Improving privacy in a world driven by increasing needs for privacy

Abstract:

Federated learning is a type of Machine Learning which works like a decentralised system. Here, learning happens locally in the devices of clients; the gradients from client devices are aggregated in a server and then each client is updated with a new ML model from the server called “global model”. The idea of Federated learning is to preserve privacy of client data. But it doesn’t prevent the clients from using malicious data to train, which might degrade the performance of the global model. This is where we bring in Blockchain Technology. Before updating the server model, we validate, using smart contracts, that the global model is performing better with the aggregation of locally trained gradients. And then, add the global model (aggregated gradients) to the blockchain network.

Contract Diagram:

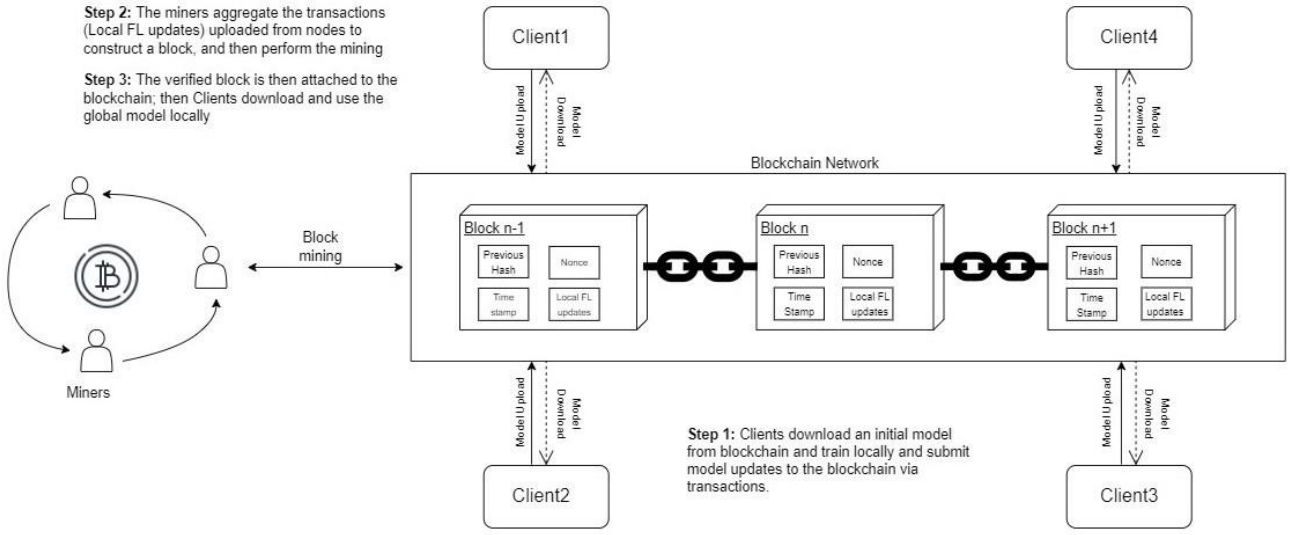
FedLearning

string server;
uint voterCount = 0;
uint clientCount = 0;
mapping(address => string) clientWeights;

function sendWeights(address x, string memory y) public
function setServer(string memory serverHash) public returns (bool)
function getServer() public view returns (string memory)
function getWeights(address a) public view returns (string memory)

Flockie
mapping(address => bool) public isRegistered;
mapping(address => bool) public hasVoted;
uint voterCount = 0;
uint update = 0;
uint noUpdate = 0;
function mintNFT(address recipient, string memory tokenURI) public returns (uint256)
function vote(address voter, uint acc) public payable
function getVoteUpdate() public view returns(bool)

Working Model:



Instructions to start the DApp

1. Start the server by running the python file in *api/main.py*
2. Run *npm install* in the project folder to install requisite dependencies
3. Deploy smart contract to the test network on Ganache and replace the contract address and *abi* values in *src/components/sc_config.js*
4. Run *npm start* to run the application

Token Design

A NFT token has been designed that gives the authority to vote and approve the new Machine Learning model. The Name of the token is “Flockie” and the symbol is “FLK”. *TokenURI* while calling the mintNFT() function can be found under src/components/sc_config.js as FLK_tiger/FLK_wolf/FLK_elephant.