A Report of Summer Internship on Federated Learning and it's applications in healthcare

(Machine Intelligence Signals and Networking lab IIT Delhi)

Submitted by:

Vani Seth

B. Tech.

(Computer Science and Engineering)

Second-Year

Jaypee University of Engineering and Technology, Guna

CONTENTS

ACKNOWLEDGEMENT

I take this opportunity to express my sincere thanks and deep gratitude to all those people who extended their whole hearted co-operation and have helped me in completing this internship successfully. I'd like to express my sincere gratitude towards **Dr. Sandeep Kumar** sir for allowing me to work with MISN Group as a Summer Intern.

Special thanks to my mentor **Ms. Nikita Malik** for all the help and guidance extended to me by her in every stage during my internship. Their inspiring suggestions and timely guidance enabled me to perceive the various aspects of the project in a new light.

Finally, I would like to express my special thanks to my families and friends helping me in all aspects and appreciate me to spend my all time in the work place during my internship time.

Vani Seth

Jaypee University of Engineering and Technology, Guna

ABSTRACT

The report includes summaries of various research papers on federated learning and its applications in healthcare. The summaries describe what is federated learning and its importance. The summaries also cover the incentive mechanism that can be used in the federated setting.

The report also has the description of datasets that is used for research in papers and their implementation.

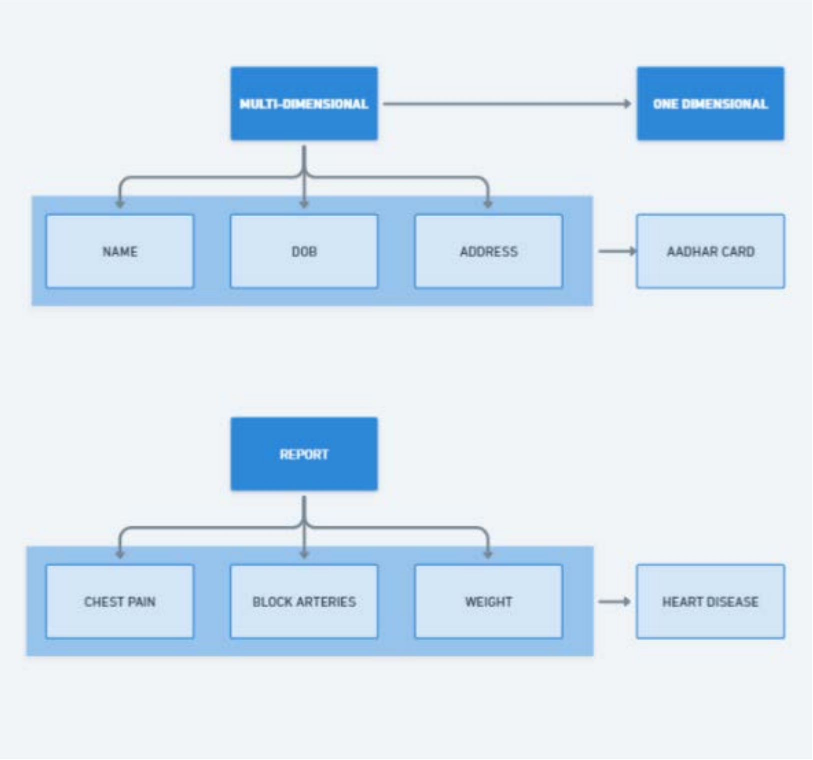
This report also includes the implementation of OpenFL paper for better understanding of how federated learning is used and implemented using the MNIST dataset in a real-life scenario.

At last, the report includes some relevant references that was used in different research papers.

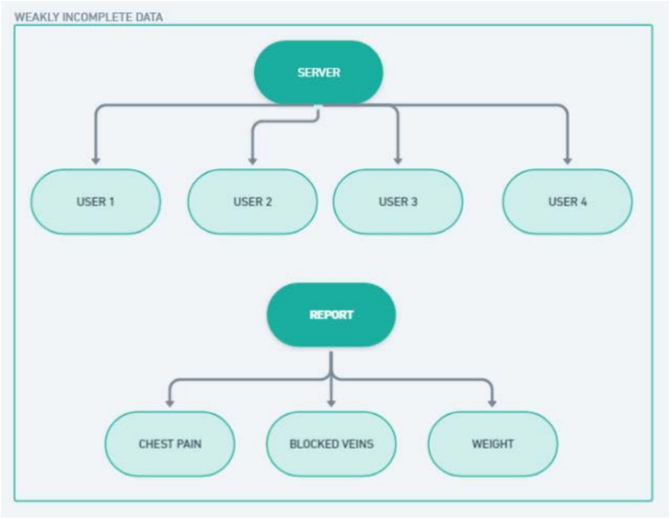
SUMMARIES OF RESEARCH PAPERS

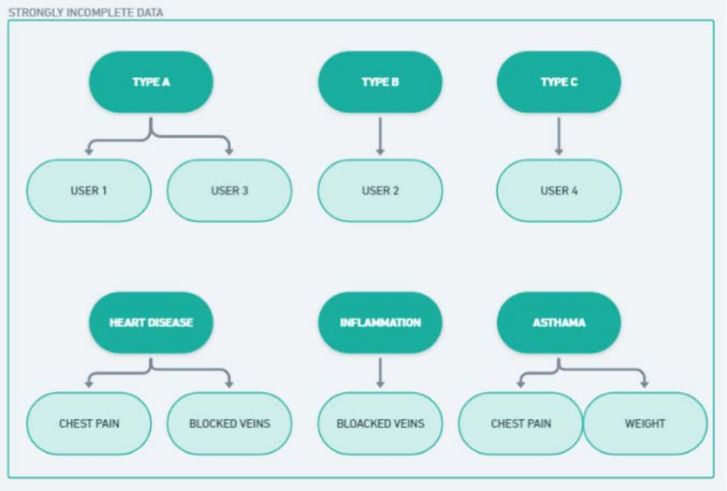
Optimal Contract Design for Efficient Federated Learning with Multi-Dimensional Private Information

Federated Learning is a process where the users train on their local data and combine the predictions, send it to the server and repeat the process to cooperatively train a global learning model. Users only need to send the most updated learning model parameters to the server without revealing their private data. This paper talks about how to incentivize users with multi-dimensional private information to participate and train the federated learning model. The authors used the results from related works [*1*] [*2*] [*3*] and found methods to test their contract design in the federated setting. In the related works incentive mechanisms were not devised so the authors proposed an incentive mechanism design with multi-dimensional private information. In this method incentives are provided according to the type and quantity of the private information the user is sharing along with that the model also considers different levels of asymmetry before providing the user with the incentive. The authors tried to find the optimal solution with the given incomplete private information. In this mechanism they summarized the user's multi-dimensional private information with a single dimension. The authors have designed incentive mechanisms with users for both IID (Independent and Identical Distribution) data and non-IID data. To propose the incentives mechanisms, they have considered three scenarios where they have weakly incomplete information scenario where the server knows the total number of users and the specific number of each user but does not know which user belong to which type, a strongly incomplete scenario where the server knows the total number of users and the distribution of the user types but do not the specific number of each user, and a complete information scenario where the server knows each user's type. Using these scenarios and the data types the authors were able to identify a way to summarize users' multi-dimensional private information with a one-dimensional metric as well as they were able to provide us with the effect of information asymmetry levels.



Multi-dimensional information to one-dimension





Weakly Incomplete Data

Strongly Incomplete Data

Learning based Incentive Mechanism for Federated Learning

In this paper the authors find an incentive-based mechanism for federated learning that motivates edge nodes to contribute to the model training. The authors have made a Deep Reinforcement Learning-based incentive mechanism to determine the optimal strategy. A major problem faced by federated learning is to incentivize people to join federated learning by contributing their computation power and data. For this a solution was proposed by other researchers to reward the participants according to their contributions. This solution though has some difficulties and is unfit in federated learning, one of the reasons for this solution being unfit in federated learning is that the relationship between the model accuracy and the amount of training data is nonlinear. The model accuracy depends on the model complexity and data quality and cannot be predicted in advance. Without the accurate predictions the previous used incentive mechanisms could not correctively reward participants, leading to financial loss or low participation rate (work can be seen in [*4*] [*5*]).

The authors have proposed a new incentive mechanism that integrates mode updating using fresh data for federated learning in IoT applications which usually includes a parameter server which resides in the cloud and some edge nodes which is in charge of some IoT device. The parameter server aims to minimize the total reward, while each edge node maximizes the revenue which is the difference of the reward received from the parameter server and the cost of data collection, model training. The author has then proposed a Deep Reinforcement Learning based incentive mechanism without any prior information. For this they have introduced a basic learning mechanism of applying Deep Reinforcement Learning into the decentralized incentive mechanism design problem. In this the model learns a general action decision from past experience based in the current state and the reward. The authors have proposed that the incentive mechanism can motivate the edge nodes to participate in the federated learning training. The deep reinforcement learning based incentive mechanism as used by the authors can learn the optimal strategies for the parameter server and edge nodes. On applying this mechanism, the authors were able to observe that the parameter server decreases its payment as the training cost increases. If the training cost is less than the server will be able to incentivize each node better. We also observed that the participation level of each node decreases as the training cost increases. Another observation that is made is that when the parameter server increases its payment to incentivize mode edge nodes, it leads to competition between the nodes so for they came up a solution that each edge node receives less reward from the parameter server. This paper was thus results in providing a better incentive mechanism using Deep Reinforcement Learning.

OpenFL: An open-source framework for Federated Learning

This paper talks about OpenFL (Open Federated Learning) which is an open-source framework for training Machine Learning algorithms. It is a production-ready Federated Learning package that allows the developers to train ML models in the nodes of remote data. The basic premise behind Federated Learning is that the AI model moves to meet the data, instead of the data moving to meet the model. A global model is sent to different users for training their local data. An aggregator node combines model updates to generate new global model that is sent back to the local users for further training. The goal of federated learning is to allow greater access to larger and more diverse datasets without violating privacy laws. The authors developed OpenFL to train ML models on the nodes of remote data owners. The models are trained on hardware at the collaborator node. The data used to train the model remains at the collaborator node at all times; only the model weight updates and metrices are shared with the model owner via the aggregator. The federation is a star topology with the collaborators and aggregators. OpenFL is installed on all the nodes of the federation and every member of the federation has a valid PKI certificate. To work with the federated system the authors demand to run an instance of a federated workload so that the workspace is distributed to all the federation members.

> How is OpenFL different from other federated learning frameworks?
>
> The main issue in FL is that the collaborators wish to protect their data and would like to ensure that their data cannot be extracted by the global model. [6] The authors designed OpenFL in such a way that is prioritizes the user's data security as well. To do so they introduced concepts like narrow interfaces, code reuse etc. For OpenFL to work every node should have OpenFL installed. OpenFL provides a PKI certificate to all the nodes to ensure security.

DATASETS USED IN THE RESEARCH PAPERS:

1. *CIFAR-10 dataset*: The CIFAR-10 is an object detection dataset that consists of 60,000 32x32 colour images in 10 classes, with 6000 images per class. There are 50,000 training images and 10,000 test images in the official data.

2. *MNIST dataset*: MNIST is a dataset that consists of handwritten digits. It is a dataset of 60,000 small square 28×28-pixel grayscale images of handwritten single digits between 0 and 9.

3. *KVASIR dataset:* The Kvasir dataset is based on images obtained from the GI tract via an endoscopy procedure. The dataset is composed of images that are annotated and verified by medical doctors, and captures 8 different classes.

4. *SIDER dataset:* SIDER dataset contains the information on marketed medicines and their recorded adverse drug reactions.

5. *PROTIENS dataset:* PROTIENS dataset consists of protein molecules and the goal is to classify whether a protein is an enzyme or not.

6. *REMBRANDT dataset:* REMBRANDT is a brain cancer biomedical dataset which consists of 110,020 pre-surgical MR images from 130 brain tumour patients. This dataset is used to classify if a patient has brain tumour or not.

IMPLEMENTATION OF PAPER

OpenFL: An open-source framework for Federated Learning

Dataset used: MNIST dataset

Link for the implementation on MNIST dataset using OpenFL:

https://colab.research.google.com/drive/1HBQk7GZU_wW0_TCqWt1ODEk1bDIFETIX?usp=sharing

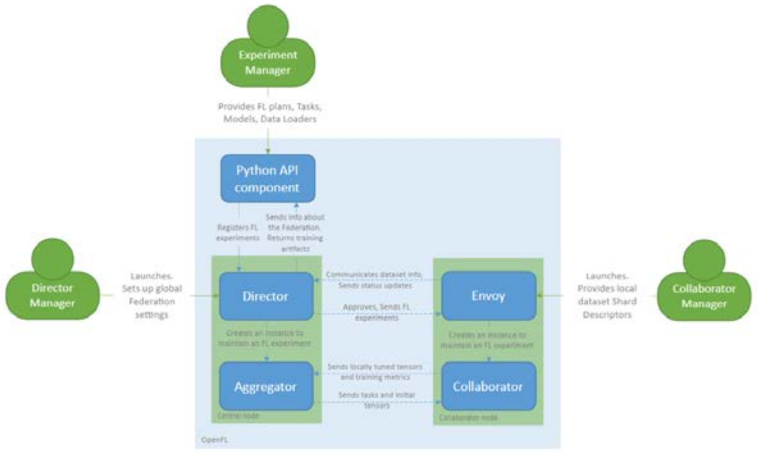
How does the model work?

The model used OpenFL and its libraries to train a federated learning model on the MNIST dataset. Along with OpenFL tensorflow and keras are also being used. A workspace is also created for better visualization and implementation.

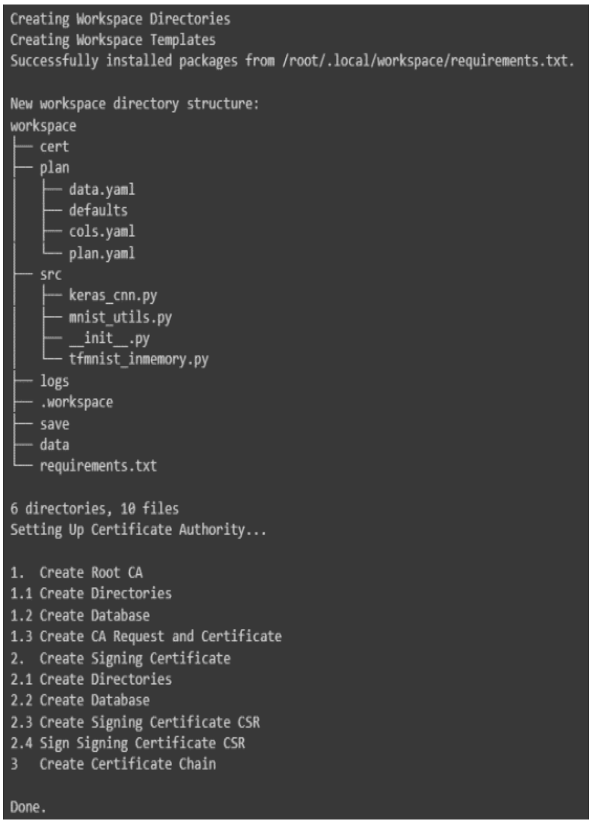
To achieve high accuracy, we use CNN (Convolutional Neural Networks) with a function model. The code creates two layers which consists of an input layer and dense layers. The dense layers are used for the calculation of the activation function. We have used a softmax activation function in this model.

A federated model is then created using the FederatedModel object. It provides built in federated training and validation functions. To test the model a setup of 2 collaborators are made. MNIST data is divided for each of the collaborator for training. The model is then trained and evaluated.

We test the unbalanced split of data on MNIST dataset when different aggregation algorithms for OpenFL are used.

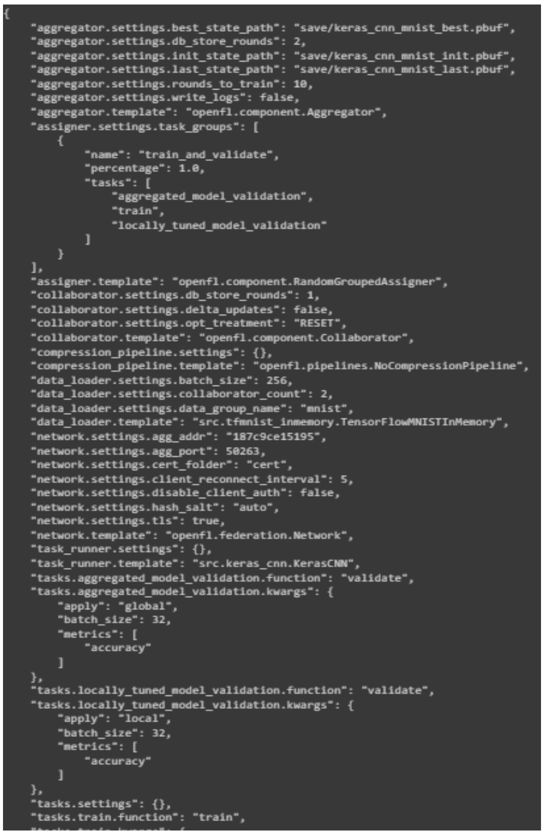

Experiment Manager
Provides FL plans, Tasks, Models, Data Loaders
Python API component
Registers FL experiments
Sends info about the Federation. Returns training artifacts
Director Manager
Launches. Sets up global Federation settings
Director
Communicates dataset info. Sends status updates
Approves, Sends FL experiments
Envoy
Launches. Provides local dataset Shard Descriptors
Collaborator Manager
Creates an instance to maintain an FL experiment
Creates an instance to maintain an FL experiment
Aggregator
Sends locally tuned tensors and training metrics
Sends tasks and initial tensors
Collaborator
Central node
Collaborator node
OpenFL

Overview of the Director-Based Workflow



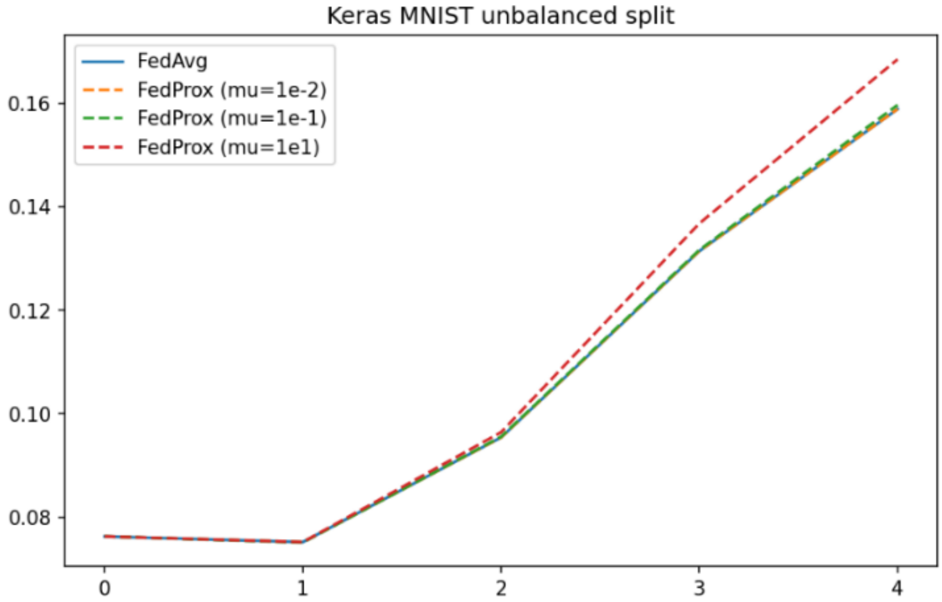
```
Creating Workspace Directories
Creating Workspace Templates
Successfully installed packages from /root/.local/workspace/requirements.txt.

New workspace directory structure:
workspace
├── cert
├── plan
│   ├── data.yaml
│   ├── defaults
│   ├── cols.yaml
│   └── plan.yaml
├── src
│   ├── keras_cnn.py
│   ├── mnist_utils.py
│   ├── __init__.py
│   └── tfmnist_inmemory.py
├── logs
├── .workspace
├── save
├── data
└── requirements.txt

6 directories, 10 files
Setting Up Certificate Authority...

1.  Create Root CA
1.1 Create Directories
1.2 Create Database
1.3 Create CA Request and Certificate
2.  Create Signing Certificate
2.1 Create Directories
2.2 Create Database
2.3 Create Signing Certificate CSR
2.4 Sign Signing Certificate CSR
3   Create Certificate Chain

Done.
```

Workspace creation



```
{
    "aggregator.settings.best_state_path": "save/keras_cnn_mnist_best.pbuf",
    "aggregator.settings.db_store_rounds": 2,
    "aggregator.settings.init_state_path": "save/keras_cnn_mnist_init.pbuf",
    "aggregator.settings.last_state_path": "save/keras_cnn_mnist_last.pbuf",
    "aggregator.settings.rounds_to_train": 10,
    "aggregator.settings.write_logs": false,
    "aggregator.template": "openfl.component.Aggregator",
    "assigner.settings.task_groups": [
        {
            "name": "train_and_validate",
            "percentage": 1.0,
            "tasks": [
                "aggregated_model_validation",
                "train",
                "locally_tuned_model_validation"
            ]
        }
    ],
    "assigner.template": "openfl.component.RandomGroupedAssigner",
    "collaborator.settings.db_store_rounds": 1,
    "collaborator.settings.delta_updates": false,
    "collaborator.settings.opt_treatment": "RESET",
    "collaborator.template": "openfl.component.Collaborator",
    "compression_pipeline.settings": {},
    "compression_pipeline.template": "openfl.pipelines.NoCompressionPipeline",
    "data_loader.settings.batch_size": 256,
    "data_loader.settings.collaborator_count": 2,
    "data_loader.settings.data_group_name": "mnist",
    "data_loader.template": "src.tfmnist_inmemory.TensorFlowMNISTInMemory",
    "network.settings.agg_addr": "187c9ce15195",
    "network.settings.agg_port": 50263,
    "network.settings.cert_folder": "cert",
    "network.settings.client_reconnect_interval": 5,
    "network.settings.disable_client_auth": false,
    "network.settings.hash_salt": "auto",
    "network.settings.tls": true,
    "network.template": "openfl.federation.Network",
    "task_runner.settings": {},
    "task_runner.template": "src.keras_cnn.KerasCNN",
    "tasks.aggregated_model_validation.function": "validate",
    "tasks.aggregated_model_validation.kwargs": {
        "apply": "global",
        "batch_size": 32,
        "metrics": [
            "accuracy"
        ]
    },
    "tasks.locally_tuned_model_validation.function": "validate",
    "tasks.locally_tuned_model_validation.kwargs": {
        "apply": "local",
        "batch_size": 32,
        "metrics": [
            "accuracy"
        ]
    },
    "tasks.settings": {},
    "tasks.train.function": "train",
    "tasks.train.kwargs": {
```

Shows the current values of the plan



The plot shows the difference between balanced and unbalanced split of the Keras MNIST dataset when different aggreagtion algoirthms for OpenFL is used.

REFERENCES:

[1] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-IID data,” 2018, arXiv:1806.00582. [Online]. Available: http://arxiv.org/abs/1806.00582

[2] J. Ren, G. Yu, and G. Ding, “Accelerating DNN training in wireless federated edge learning systems,” 2019, arXiv:1905.09712. [Online]. Available: http://arxiv.org/abs/1905.097124

[3] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, “Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory,” IEEE Internet Things J., vol. 6, no. 6, pp. 10700–10714, Dec. 2019.

[4] Y. Peng, Y. Bao, Y. Chen, C. Wu, and C. Guo, “Optimus: an efficient dynamic resource scheduler for deep learning clusters,” in Proc. of ACM EuroSys, 2018, pp. 1–14.4

[5] Y. Zhan, S. Guo, P. Li, K. Wang, and Y. Xia, “Big data analytics by crowdlearning: Architecture and mechanism design,” IEEE Network, 2019.

[6] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., \Advances and open problems in federated learning," arXiv preprint arXiv:1912.04977, 2019.