EMITTER International Journal of Engineering Technology
Vol. 11, No. 1, June 2023, pp. 1~20
DOI : 10.24003/emitter.v11i1.742

Federated Learning Framework for IID and Non-IID datasets of Medical Images

Kavitha Srinivasasn[1], Sainath Prasanna[1], Rohit Midha[1], Shraddhaa Mohan[1]

[1]Department of CSE, Sri Sivasubramaniya Nadar College of Engineering, Chennai, Tamilnadu, India
Corresponding Author: kavithas@ssn.edu.in

Received August 24, 2022; Revised November 8, 2023; Accepted December 29, 2023

Abstract

Advances have been made in the field of Machine Learning showing that it is an effective tool that can be used for solving real world problems. This success is hugely attributed to the availability of accessible data which is not the case for many fields such as healthcare, a primary reason being the issue of privacy. Federated Learning (FL) is a technique that can be used to overcome the limitation of availability of data at a central location and allows for training machine learning models on private data or data that cannot be directly accessed. It allows the use of data to be decoupled from the governance (or control) over data. In this paper, we present an easy-to-use framework that provides a complete pipeline to let researchers and end users train any model on image data from various sources in a federated manner. We also show a comparison in results between models trained in a federated fashion and models trained in a centralized fashion for Independent and Identically Distributed (IID) and non IID datasets. The Intracranial Brain Hemorrhage dataset and the Pneumonia Detection dataset provided by the Radiological Society of North America (RSNA) are used for validating the FL framework and comparative analysis.

Keywords: Federated Learning, Federated Learning framework, Classification task, Object detection task, Medical datasets.

1. INTRODUCTION

Recent advances made in machine learning have shown that it is an efficient tool that can be used successfully in segmentation, image analysis, and even reconstruction of images from sensor data. Deep neural networks have shown the most promise in image segmentation, analysis, and image generation (done by GANs). These advances in Computer vision have been made due to the availability of extremely large repositories of data. For example, the popular image database ImageNet [1] has over 14 million images of common items and objects. This is where a limitation is placed in fields such as healthcare where there is not enough public data available.

Medical imaging researchers have addressed this by either gathering or producing large, high-quality datasets by themselves such as the UK Biobank [2], but even the Biobank has very little data when compared to huge datasets such as ImageNet.

The reason for the lack of publicly available data in many fields is not the fact that data is not collected as often. Rather it is because the data collected (for example, by hospitals, clinicians and researchers) is extremely sensitive, confidential and often contains private information. Understandably, access to this data is highly constrained, even within institutions, due to important client data privacy regulations. To overcome the data scarcity and privacy in maintaining healthcare related data, Federated Leaning (FL) is introduced in domain of machine learning.

In [3], Shaheen et al., discussed the applications of Federated Learning (FL) in different domains and the taxonomy of FL with research trends for better understanding. Ng et al., listed the four challenges in implementing FL with the available resources related to both hardware and software environments in 2021 [4]. FL is prominently useful in developing applications using medical data or images, where data privacy is mandatory [5, 6]. The implementation of FL is carried out using deep neural network techniques such as Convolutional Neural Network, ResNet, EfficientNet and ResNeXt for feature extraction, classification, prediction and detection [7, 8]. At present, FL is combined with upcoming technologies such as Explainable AI and Blockchain for industry 5.0 applications [9].

This research explains a complete pipeline of federated learning framework for the researchers and end users to train any model on image data from various sources in a federated manner. In Sect. 2, the related works of Federated Learning are explained. The originality of this research paper is briefly listed in Sect. 3. The proposed methodology is explained with design and configuration in Sect. 4. In the result Section, comparisons between centrally trained models and federated models are discussed to showcase the efficacy of federated learning with IID and Non-IID datasets of medical images. The datasets used are already available such as Intracranial Brain Hemorrhage dataset and the Pneumonia Detection dataset to simulate federated learning for the purpose of experimentation. Finally, the paper is summarized with conclusion.

2. RELATED WORKS

Federated Learning is a way to train AI models and it is capable to work on heterogeneous dataset. It has its applications in healthcare, insurance and industrial sectors as transforming healthcare industry for better patient care, identifying fraudulent activities and improving Siri's voice recognition respectively. In the healthcare sector, federated learning allows the individual hospitals to benefit from the rich datasets of multiple non-affiliated hospitals without centralizing the data in one place. The Sheller et al. evidenced this in 2020 using data-private collaborative learning approach

for multi-institutional data [10]. The data of collaborative learning are independent and identically distributed (IID) so that multiple collaborators train a machine learning model at the same time (i.e., each on their own data, in parallel) and then send their model updates to a central server to be aggregated into a consensus model. The aggregation server then sends the consensus model to all collaborating institutions for further training.

In 2020, Rieke et al., stated that sharing medical data for training the dataset has security issues and it can be resolved by the concept of federated learning [11]. Here the training is given by the training algorithms collaboratively without exchanging the data itself i.e., without moving patient data beyond the firewalls of the institutions in which they reside. Instead, the ML process occurs locally at each participating institution and only model characteristics (e.g., parameters, gradients) are transferred.

It's also becoming increasingly important to maintain data privacy: true anonymization of data is difficult to achieve because it's unclear what kind of information machine learning can extract from seemingly innocuous data. For example, it's possible to predict the age [12] and sex [13] of a patient from medical images, and we've seen that in some cases, multiple anonymized datasets can be combined to deanonymize them [14].

These privacy concerns are important and necessary, but this has limited researchers from fully maximizing the benefits of artificial intelligence in research. One recent development that helps overcome this limitation is Federated Learning (FL), introduced by Google in 2017 [15]. Federated Learning allows us to train a model on data from multiple institutions, organizations, and people without sharing the data. Therefore, in this study FL was chosen to address the issues in the research of medical imaging such as public data scarcity due to ethical reasons and privacy preserving of data.

3. ORIGINALITY

The uniqueness of the proposed work is: Open Source Federated Learning (FL) Framework, Supports multiple FL algorithms, multiple Pytorch models for both Image Classification and Object Detection tasks, Easily configurable and takes minutes to set up the runnable environment with Clear and informative User Interface developed using React based web app.

4. SYSTEM DESIGN

The design and configuration are explained with two subsections namely, framework architecture and its design with deployment.

4.1 Framework Architecture

The Federated Learning (FL) framework was built using the client-server architecture. In this approach the centralized server acts as the master and various clients are units where the data is stored. The centralized server

coordinates with the clients and facilitates training across the clients as shown in Figure 1.

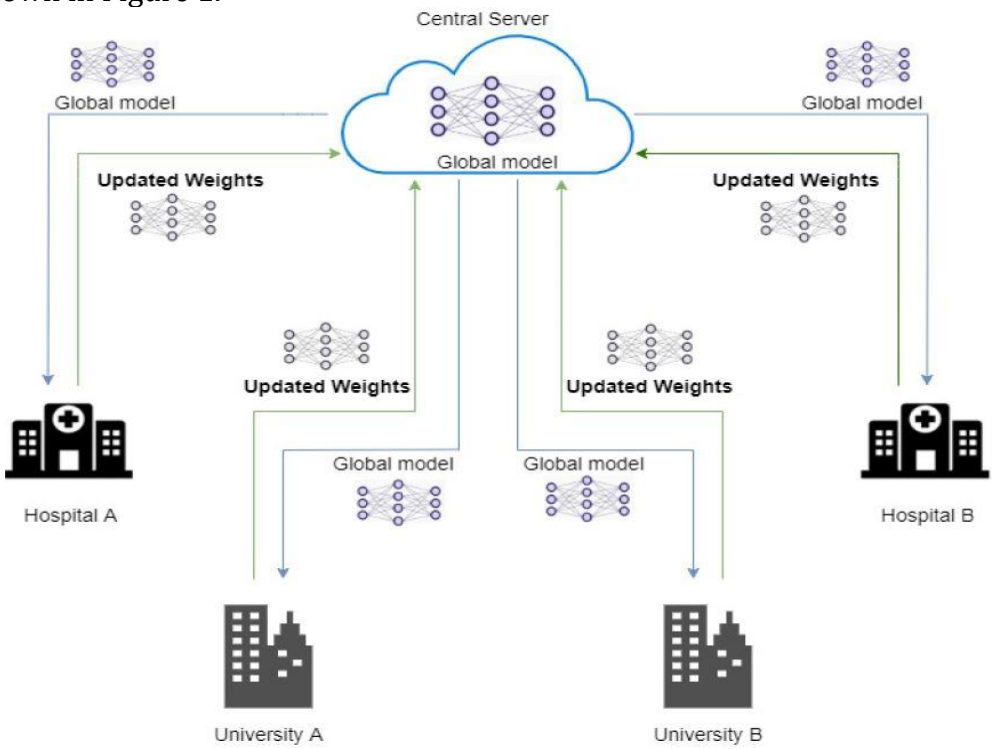


Figure 1. Client Server Federated Learning Architecture

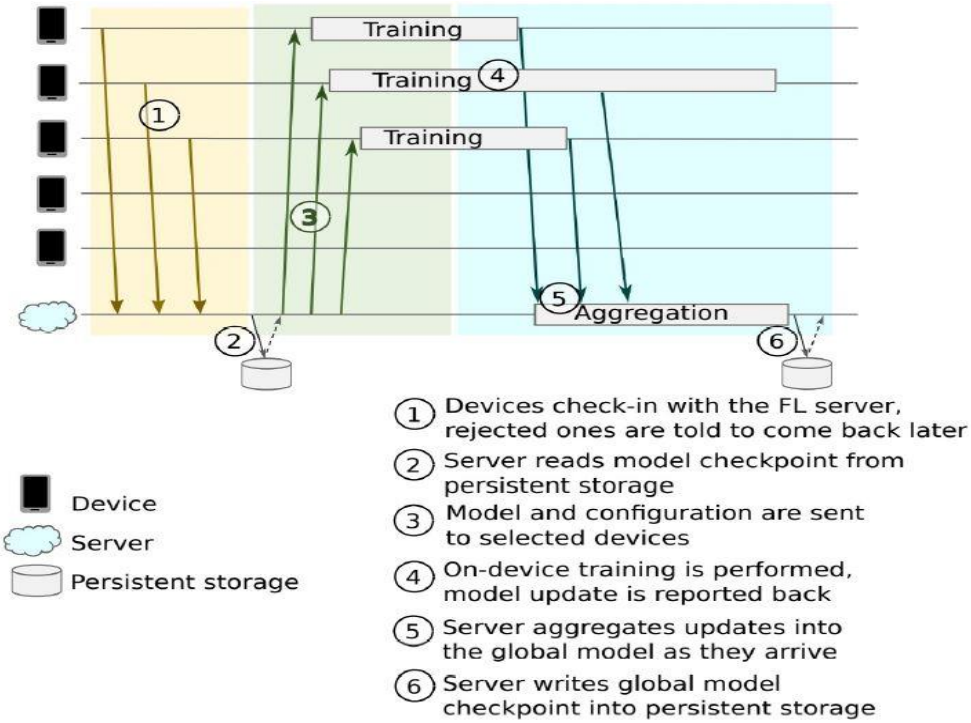


Figure 2. Workflow Diagram of 1 Round

At the start of an FL round, the central server which hosts the global model sends the model to all the clients. These clients train on their data following certain configurations. After training, the clients send their updated models back to the central server. The central server on receiving client models executes the chosen algorithm and combines all the models to form the new global model which is subsequently sent to all clients at the start of the next FL round. The workflow diagram of "one round" is given in Figure 2.

The sequence of steps that constitute one FL round are: Server sends a global model, Clients train on local data and sends an updated models to the server and Server aggregates all received models to form the new global model.

The FL framework can be split into two main processes such as: Client Server Connection Establishment and FL Rounds.

4.1.1 Client Server Connection Establishment

This process deals with registering a new client and generating a unique token for the client in the designed framework. Initially, the client makes a HTTP request to the server endpoint /register_client to connect to it, as shown in Figure 3. The server responds by sending the client a unique hexadecimal token that is used for authentication. The token is also used to keep track of the state of the client throughout the working of the system. The client receives the token and responds with a received message and the token. The server checks if the client has sent the correct token, authenticates and registers the client.

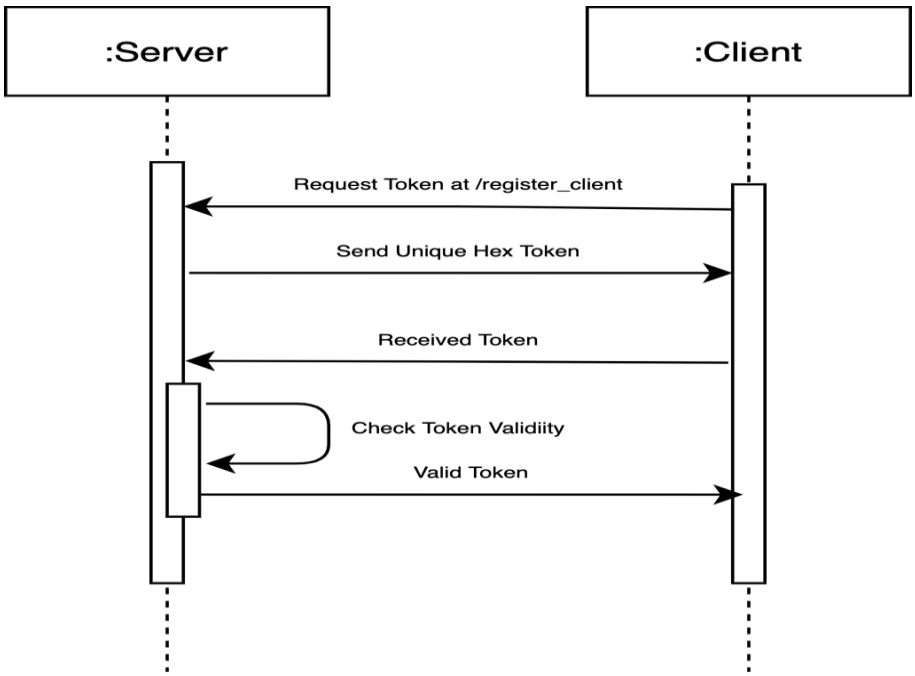


Figure 3. Client Connection to the Server

4.1.2 FL Rounds

This process deals with the actual federated learning process post registration and authentication of the client. Once the client has connected to the server, it first gets the global model from the server by making a HTTP request to the /get_model endpoint. For the request made the client attaches the allocated token and the server uses this for client validation and registering the state of the client. If the client isn't valid, the server sends a *INVALID_CLIENT* message to the client.

Once the client is authenticated, the server accesses the persistent storage and sends the model to the client along with the file size. The client receives the model and uses the file size to make a simple check on whether the file was transmitted correctly. It saves the model (global model) and tells the server that the received model is valid through the /model_received endpoint. Post this, the client starts performing local training.

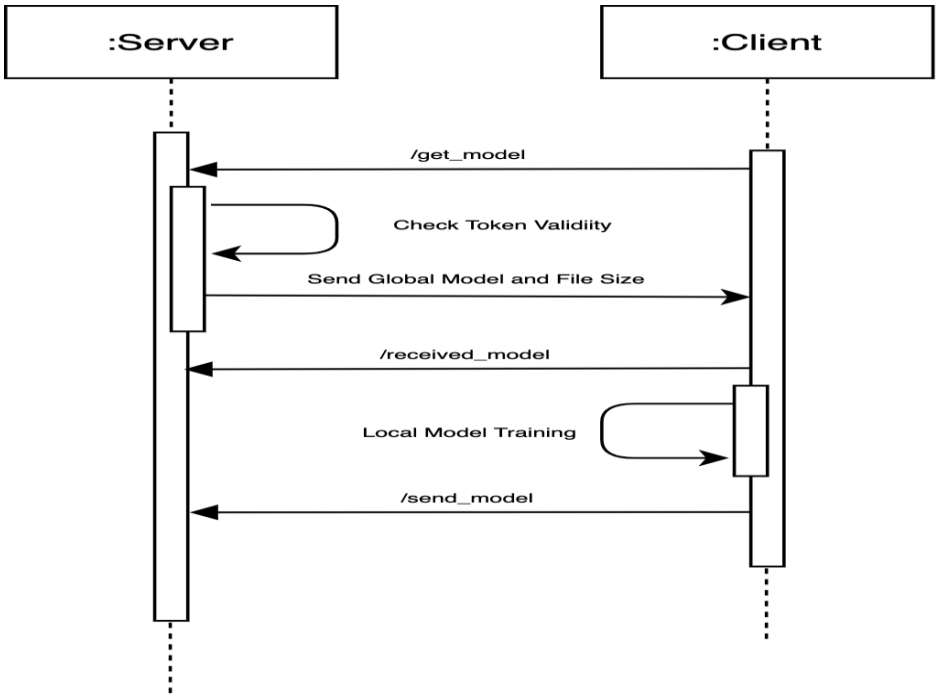


Figure 4. One Client FL Round

After a fixed number of epochs, the client sends the model to the server using the /send_model endpoint. The server waits for all the connected clients to send their models before performing aggregation. Post aggregation of the models, the server updates the global model and sends it to the clients to repeat the process over again for a fixed number of rounds.

4.2. Framework Design

The FL framework design elaborates the creation of configuration files with its parameters, user interface design, importance of heartbeat messages, identification of dead clients, termination, and validation.

4.2.1 Configuration Files

The framework is designed in a way that the user can configure everything through a configuration file. Both the server and the client work on the basis of YAML configuration files are predefined and can be modified by the user with ease.

```yaml
# CONFIGURATION FILE FOR THE SERVER
name: BETA_TEST_V0.1
model_path: GLOBAL/global_model.pth
client_updates_path: CLIENT_UPDATES/
mode: "classification"
HOST: "http://192.0.122.14"
PORT: 9865

#DEEP LEARNING CONFIG
arch: "resnet18"
n_classes: 6

# FL PARAMS
rounds: 15
max_clients: 5
aggregation: "FedAvg"
```

Whereas a typical client configuration files would look like:

```yaml
# CONFIGURATION FILE FOR CLIENT
mode: "classification" #classification, detection
send_after_epoch: 1
weight_update: "normal" #uga, normal

# MODEL
model_folder: CLIENT_MODEL
model_name: client.pth
arch: "resnet18"

# IP
HOST: "http://192.0.122.14"
PORT: "9865"

# DATASET
labels:
  [
    "epidural",
    "intraparenchymal",
    "intraventricular",
    "subarachnoid",
    "subdural",
    "any",
  ]
n_classes: 6 #should match len(labels)

# TRAINING
train_csv_file: "FL/classification/client1/client1_train.csv"
train_image_dir: "FL/data/proc/"
train_batch_size: 64

# PARAMS
epochs: 15
lr: 0.00002
device: "cuda:0" #cpu, cuda
optimizer: "Adam" #Adam, SGD

# TESTING
test_csv_file: "FL/classification/client1/client1_val.csv"
test_image_dir: "FL/data/proc"
test_batch_size: 1
```

Configuration files were used as they are easy to understand and so that even people who aren't as experienced with coding can use the system. This would allow everyone to use the system without having to spend a long time learning how to use the framework.

4.2.2 User Interface

The framework provides for a Web User Interface (UI) which allows the user to keep track of the server and client. For the server, the UI shows the current FL Round the server is in. For each client, connected to the server, the UI shows the client's unique token, it's status (Active / Inactive) and the current epoch the client is in.

4.2.3 Heartbeat

For the server to know the status of each client during the local updating process, the client sends heartbeat messages at a regular interval to the server. By default, the heartbeat messages are set to notify the server every 5 seconds but this is something that the user can configure based on need.

Each heartbeat consists of the clients assigned token for authorization on the server side, along with other useful information like client status, current epoch, exception raised on client side, etc. The information provided by the heartbeat is used by the accompanying user interface to display the health and status information of all nodes.

4.2.4 Dead Clients

The framework provides a way to resume training in case an exception arises. The server saves the information of when the client disconnected and when the client reconnects this information is used to resume the client from where it left off. Users can resume the client by running the main client python script file.

4.2.5 Algorithm used for FL design

The framework currently supports 2 tasks, namely:

- Classification Task
- Object Detection Task

For the two tasks, the framework contains high quality deployments of popular CNN architectures. For the same, the project makes use of a supporting framework, Pytorch Image Models [16].

Few of the supported architectures are as listed below:

- EfficientNet [17] and it's corresponding architectures (B0-B7, etc).
- ResNet [18] and it's corresponding architectures (18, 34, 50, 101, 152).
- ResNeXt [19] and it's variants.

The framework also implements the FedAvg [15] for averaging all the client models on the server side. Further, there is an option for the user to

choose whether the weight updates on the client side happen in the normal fashion or using Unbiased Gradient Aggregation [20]. There is also support for the Augmentations [21] library, that lets the user define custom augmentations. Internally, ResNet followed by FasterRCNN architecture is used for FL design implementation in which, the last layer of the ResNet is substituted with FasterRCNN. The relationship between the model and the different layers of ResNet is shown in Figure 5 for better understanding.

Model: "model"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 32, 32, 3)]	0	[]
conv2d (Conv2D)	(None, 32, 32, 16)	448	['input_1[0][0]']
batch_normalization (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d[0][0]']
activation (Activation)	(None, 32, 32, 16)	0	['batch_normalization[0][0]']
conv2d_1 (Conv2D)	(None, 32, 32, 16)	2320	['activation[0][0]']
batch_normalization_1 (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d_1[0][0]']
activation_1 (Activation)	(None, 32, 32, 16)	0	['batch_normalization_1[0][0]']
conv2d_2 (Conv2D)	(None, 32, 32, 16)	2320	['activation_1[0][0]']
batch_normalization_2 (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d_2[0][0]']
add (Add)	(None, 32, 32, 16)	0	['batch_normalization_2[0][0]', 'activation[0][0]']
activation_2 (Activation)	(None, 32, 32, 16)	0	['add[0][0]']
conv2d_3 (Conv2D)	(None, 32, 32, 16)	2320	['activation_2[0][0]']
batch_normalization_3 (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d_3[0][0]']
activation_3 (Activation)	(None, 32, 32, 16)	0	['batch_normalization_3[0][0]']
conv2d_4 (Conv2D)	(None, 32, 32, 16)	2320	['activation_3[0][0]']
batch_normalization_4 (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d_4[0][0]']
add_1 (Add)	(None, 32, 32, 16)	0	['batch_normalization_4[0][0]', 'activation_2[0][0]']
activation_4 (Activation)	(None, 32, 32, 16)	0	['add_1[0][0]']
conv2d_5 (Conv2D)	(None, 32, 32, 16)	2320	['activation_4[0][0]']
batch_normalization_5 (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d_5[0][0]']
activation_5 (Activation)	(None, 32, 32, 16)	0	['batch_normalization_5[0][0]']
conv2d_6 (Conv2D)	(None, 32, 32, 16)	2320	['activation_5[0][0]']
batch_normalization_6 (BatchNormalization)	(None, 32, 32, 16)	64	['conv2d_6[0][0]']
add_2 (Add)	(None, 32, 32, 16)	0	['batch_normalization_6[0][0]', 'activation_4[0][0]']
activation_6 (Activation)	(None, 32, 32, 16)	0	['add_2[0][0]']
global_average_pooling2d (GlobalAveragePooling2D)	(None, 16)	0	['activation_6[0][0]']
flatten (Flatten)	(None, 16)	0	['global_average_pooling2d[0][0]']
dense (Dense)	(None, 10)	170	['flatten[0][0]']

Figure 5. Summary of ResNet architecture used in FL design

4.2.6 Termination and Validation of FL Rounds

The federated learning process continues for a chosen number of FL rounds or until certain conditions are met such as an accuracy threshold. Validation of the FL experiment can be conducted after the termination condition has been reached. The federated model can be validated against a global dataset consisting of a small amount of data from all organizations for the purpose of validation. If this is not possible, each organization can validate the federated model on their own local validation sets.

5. EXPERIMENTS AND RESULTS

This section explains the experiments done using the framework to benchmark the various implemented models for image classification and object detection datasets. The models are developed with centralized and federated way (IID, non IID data splits) and analyzed.

5.1 Classification Results

The image classification dataset, FL configuration set-up and results of centralized and decentralized are obtained and analyzed in the following subsections**.**

5.1.1 Dataset

For testing the framework and for establishing benchmarks we make use of a sample medical dataset. Finding medical imaging data is not easy, hence we make use of a public dataset for establishing the benchmark. In specific, we use a dataset provided by the Radiological Society of North America (RSNA) in collaboration with members of the American Society of Neuroradiology and MD.ai for predicting of intracranial brain hemorrhage.

The data consists of **752,803** images and all provided images are in *DICOM (.dcm)* format. Each image contains the following metadata: *PatientID, StudyInstanceUID, SOPInstanceUID, PhotometricInterpretation, SeriesInstanceUID, Modality* and other features. The task is to predict whether a hemorrhage exists in each of the given images, and the type of hemorrhage.

5.1.2 Setup

In realistic split, we assumed that each patient would get scans done only at a single hospital. Hence using this assumption, the data is split into 5 folds, where each patient can belong only to 1-fold. Of these 5 folds, 4 folds were used for training, and one was used for testing. In the centralized implementation, all the 4 training folds were used for training with all the data on a central node. However, for the FL model, the 4 data folds were split across 2 nodes for training, both uniformly and non-uniformly.

For benchmarking we differentiate between two types of datasets, Independent and Identically Distributed Data (IID) and Non IID. Identically Distributed means all items in the sample are taken from the same probability distribution. Independent means that the sample items are all

independent events. However, in practice, the data on each client device is not IID and it would be wrong to assume the same. For example, devices within the same geolocation are likely to have correlated data thus leading to violation of independence. Due to devices being tied to geolocations, the distribution of labels varies across partitions thus leading to violation of the identical property. We considered both IID and non IID data and made the following splits for 2 clients.

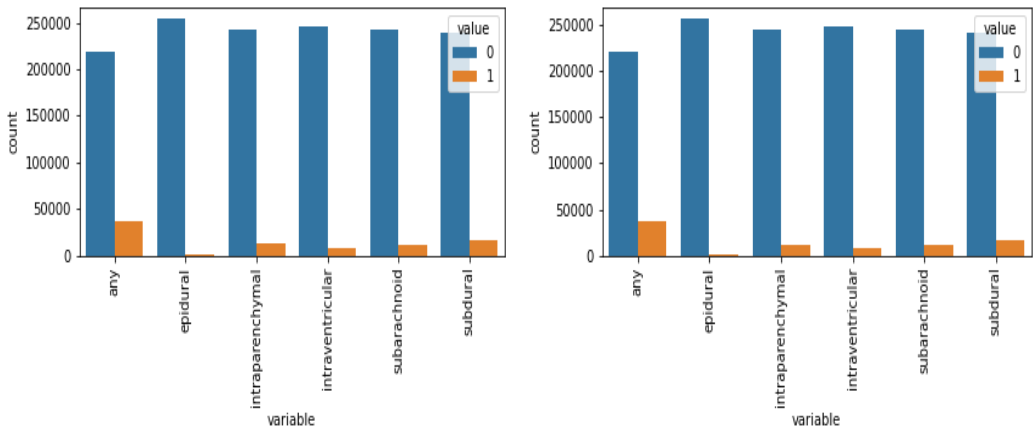


Figure 6. IID Data Split

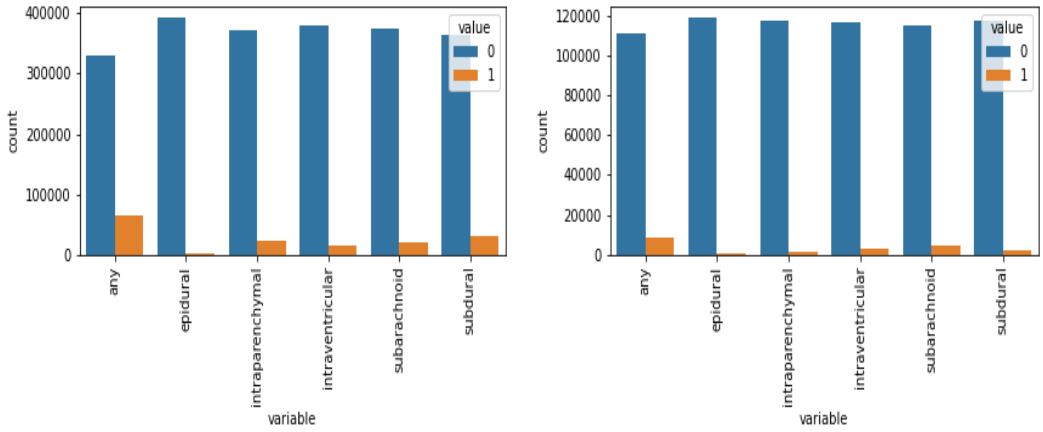


Figure 7. Non-IID Data Split

In Figure 6 the amount of data on each node is the same, indicated by the scale. Further each class has an identical distribution. However, in Figure 7, Client 1 has much more data than Client 2 as is indicated by the scale. Further, both clients don't have identical class distribution either as is seen from varying distributions in the *Intraparenchymal* and *Subdural* classes.

5.1.3 Results

The models for classification were trained on the ResNet architecture with a custom fitted Linear head in centralized and decentralized (federated) methods. The Adam optimizer [22] was used with a learning rate of 0.00002.

Table 1. Centralized Classification Results

Datasets	Weighted Log Loss
Train Set	0.0556
Test Set	0.0906

The results of classification on the training and testing set using the central model is given in Table 1. This model was trained for 10 epochs with two quantitative metrics to measure the performance namely weighted log loss and ROC AUC Score.

Figure 8 shows the ROC AUC curve of each of the 6 classes in the train set. Figure 9 shows the ROC AUC curves and scores obtained by the Centralized model on each of the 6 classes in the test set.

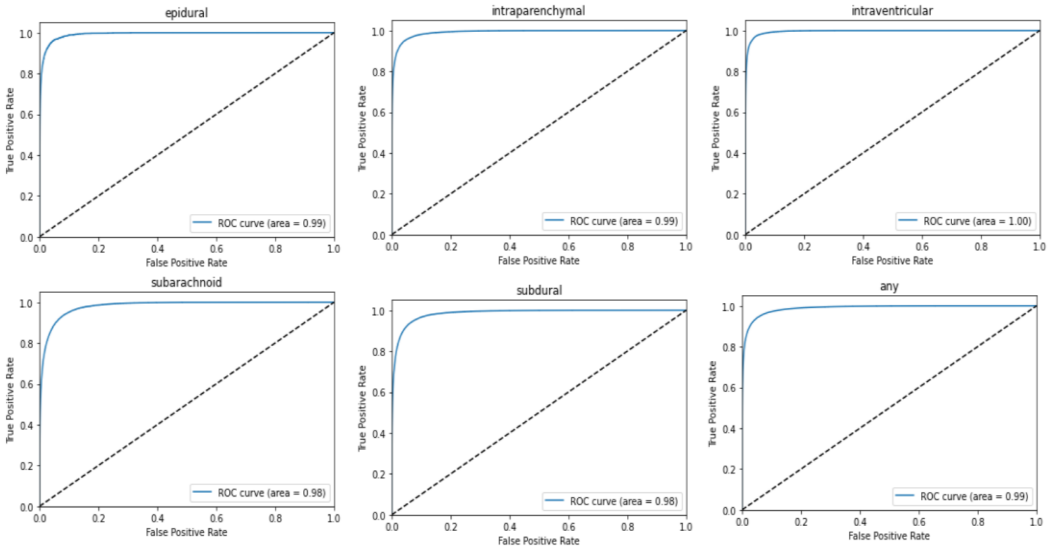


Figure 8. ROC AUC Curves for Centralized Implementation using Train Set

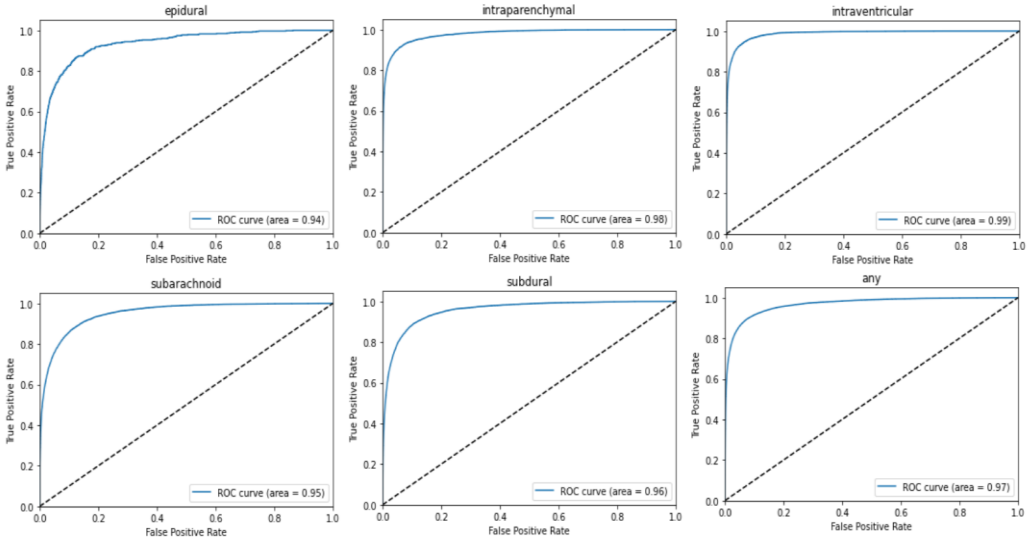


Figure 9. ROC AUC Curves for Centralized Implementation using Test Set

For **IID Data Splits**, the federated model was trained for 10 rounds with each client sending an update per epoch. The result obtained on training the classification model in a federated learning way is shown in Table 2.

Table 2. Federated Classification on IID Data Experiment Results

Experiment	Weighted Log Loss
FL Classification 1_epoch	0.0930

Table 2 shows the weighted log loss obtained on the test set. This result is comparable to the weighted log loss in Table 1. As can be seen, the FL model performs almost at par with the Centralized model on the test set.

Figure 10 shows the ROC AUC curves and scores obtained by the FL model for each of the 6 classes in the test set. These curves are comparable to the ones shown in Figure 9. As can be seen, the FL model performs at par with the Centralized model.

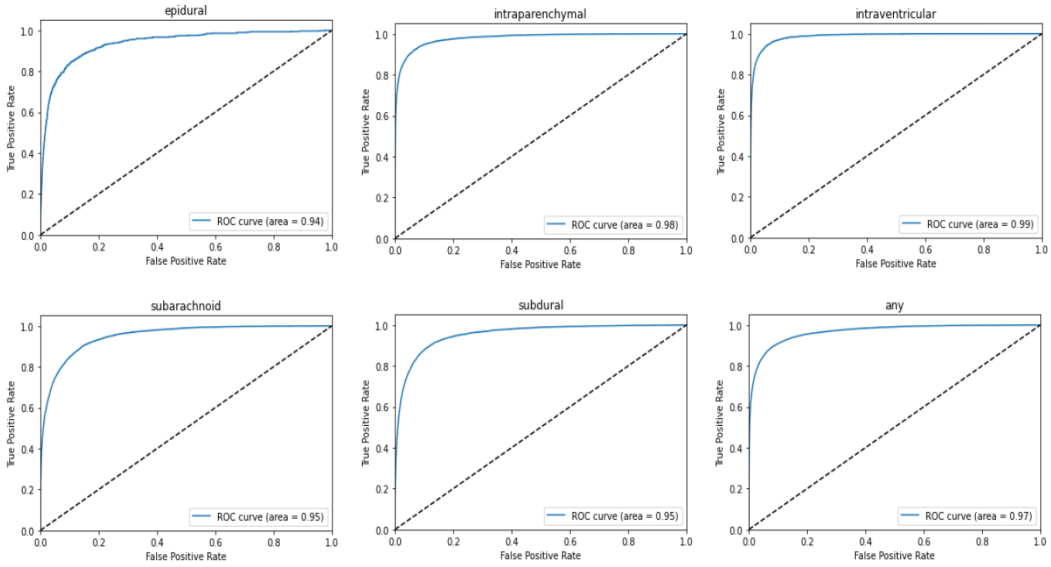


Figure 10. ROC AUC Curves for Federated Implementation [IID] using Test Set

Non IID Data is most like real world data and hence two experiments were performed on the classification data.

The first experiment was to perform classification on Non IID data where the clients send updates after 5 epochs of local training. Totally there were 6 FL rounds.

The second experiment was to perform classification on the Non IID data where the clients send updates after 5 epochs of local training and use the Unbiased Gradient Aggregation Scheme (UGA). 6 FL rounds were conducted in total. The following are the results of the two experiments on the test set of the Non IID data. As can be seen from Table 3, using UGA on the client side does lead to improvements when using Non IID Data. Figures 11 and 12 show the ROC AUC Curves for the experiments listed in Table 3.

Table 3. Federated Classification on Non IID Data

Experiment	Update_epoch	Weighted Log Loss
FL Classification without UGA	3	0.1171
FL Classification with UGA	3	0.0888

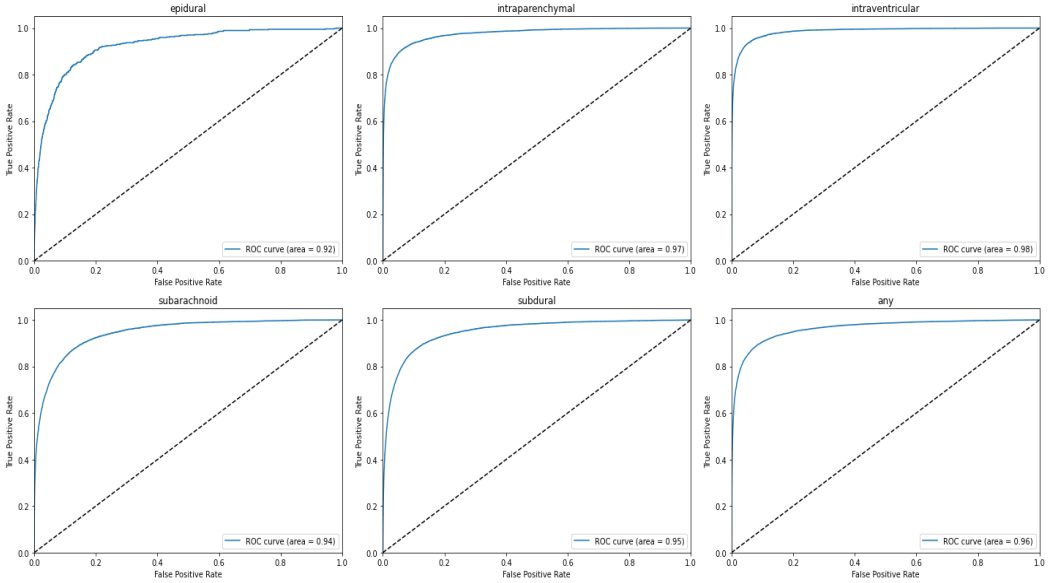


Figure 11. ROC AUC Curves for Federated Implementation [Non-IID without UGA] using Test Set

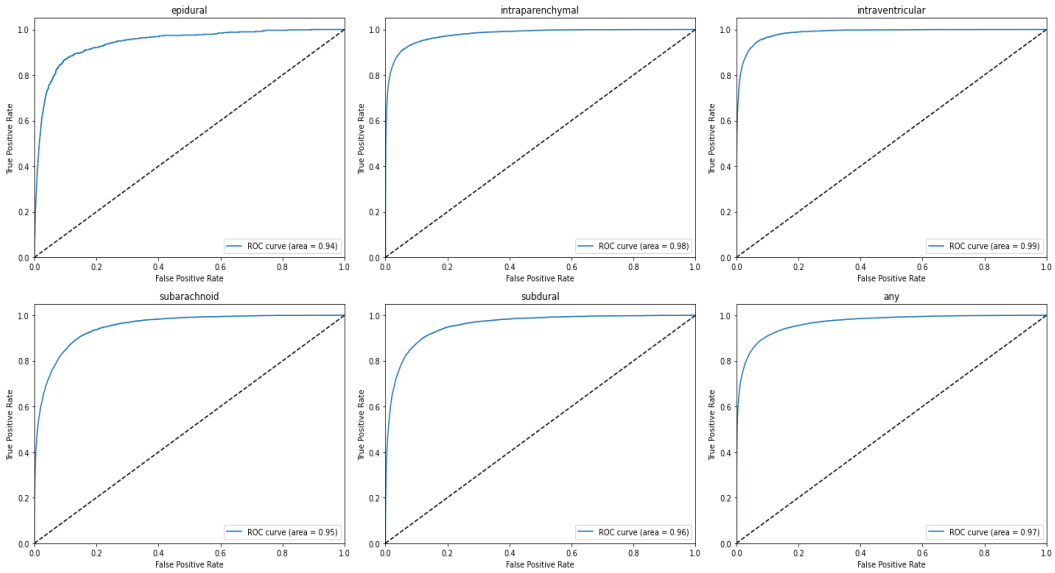


Figure 12. ROC AUC Curves for Federated Implementation [Non-IID with UGA] using Test Set

The FL model that trained on IID Data and sent an update after every epoch performed similarly to the centrally trained model. Further, it was observed that from the FL models that trained on Non IID Data and sent an update after 5 epochs, the model which used UGA had increased accuracy as shown in Figure 13.

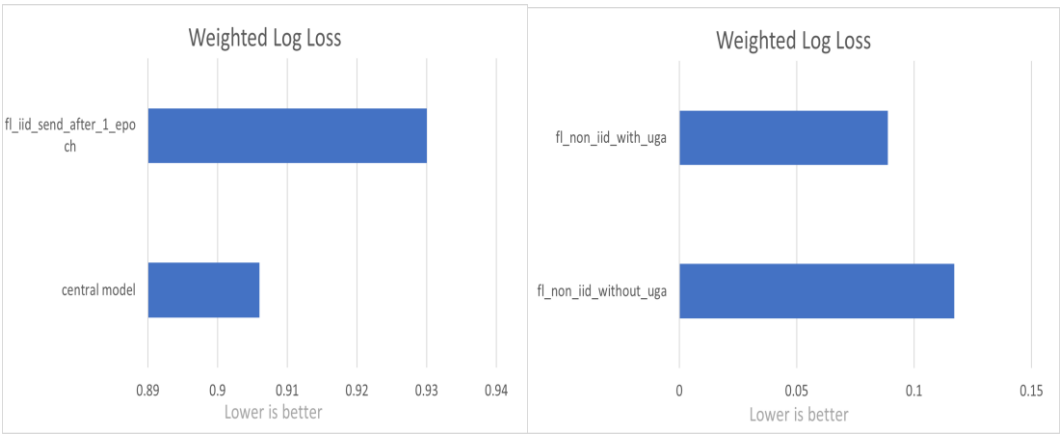


Figure 13. Performance Analysis of classification using Test Set

5.2 Object Detection Results

The image classification dataset, FL configuration set-up and results of centralized and decentralized are obtained and analyzed in the following subsections.

5.2.1 Dataset

For benchmarking the Object Detection task using FL, we use a dataset provided by the Radiological Society of North America (RSNA) for predicting whether pneumonia exists in each image. This is done by predicting bounding boxes around areas of lung opacities in chest X-Ray Images.

The dataset consists of **26,684** Chest X-Ray images in *DICOM (.dcm)* format. The dataset contains patient id, coordinates of the bounding box, width and height of the bounding box, target indicating evidence of pneumonia and class of the sample.

5.2.2 Setup

The dataset was split into 5 folds where each *patientId* belongs to only one-fold. Of these 5 folds, 4 folds were used for training and one-fold was reserved for testing. For creating a validation set, 15 percent of each training fold was set aside. There are no common *patientIds* between the training set and validation set of each fold. The class imbalance of the original dataset was maintained while creating the folds.

The architecture chosen for this task was the Faster-RCNN model [23]. This model was fitted with a ResNet50 backbone and a Feature Pyramid Network (FPN). The Stochastic Gradient Descent (SGD) optimizer was used with a learning rate of 0.001.

5.2.3 Results

The results of performing object detection on a central model and a federated model on IID data and non IID data are analyzed. The central model was trained for 10 epochs and the federated model was trained for 10 rounds with each client sending an update per epoch.

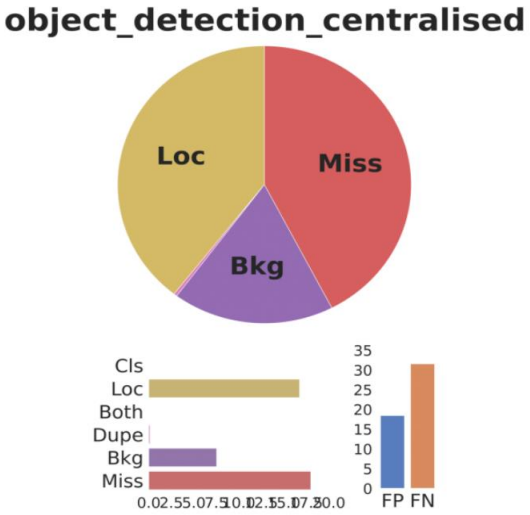
The metric used is Mean Average Precision at an IoU of 0.5 is shown in Table 4 for the model trained in a centralized fashion on the train and test sets. Also, the results obtained in a federated way are tabulated in Table 5.

Table 4. Centralized Object Detection Experiment Results

Dataset	Precision (IOU > 0.5)
Train Set	0.345
Test Set	0.311

Table 5. Federated Object Detection Experiment Results using Test Set

Experiment (Obj Detection)	Precision (IOU > 0.5)
FL Object Detection 1_epoch	0.240



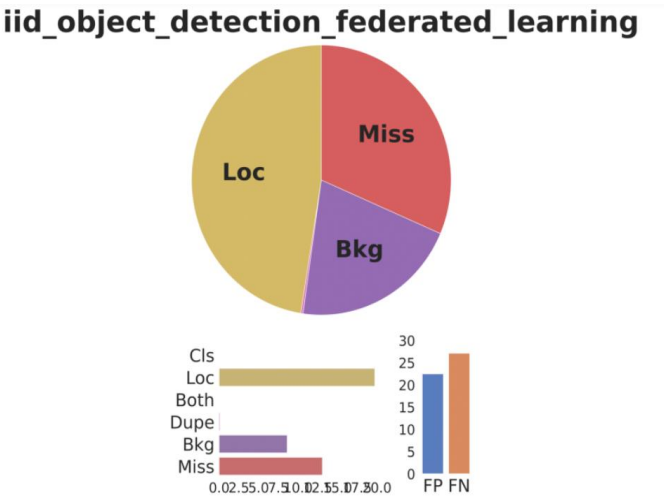

```
-- object_detection_centralised --

bbox AP @ 50: 31.06

                        Main Errors
============================================================
  Type      Cls      Loc      Both     Dupe      Bkg      Miss
------------------------------------------------------------
   dAP     0.00    16.75     0.00     0.15     7.55    18.00
============================================================

        Special Error
============================
  Type   FalsePos   FalseNeg
----------------------------
   dAP      18.45      31.53
============================
```

Figure 14. TideCV Analysis of Central Model




```
-- iid_object_detection_federated_learning --

bbox AP @ 50: 24.05

                    Main Errors
============================================================
  Type      Cls      Loc      Both     Dupe      Bkg      Miss
------------------------------------------------------------
   dAP     0.00    19.71     0.00     0.12     8.62    13.08
============================================================

       Special Error
============================
  Type   FalsePos   FalseNeg
----------------------------
   dAP      22.49      27.11
============================
```

Figure 15. TideCV Analysis of Federated Model

From the results, it can be inferred that the centralized model performs much better than the federated object detection model and a scope for improvement exists. TideCV [24] was also used to analyze the results on the test set. The findings are shown in Figures 14 and 15.

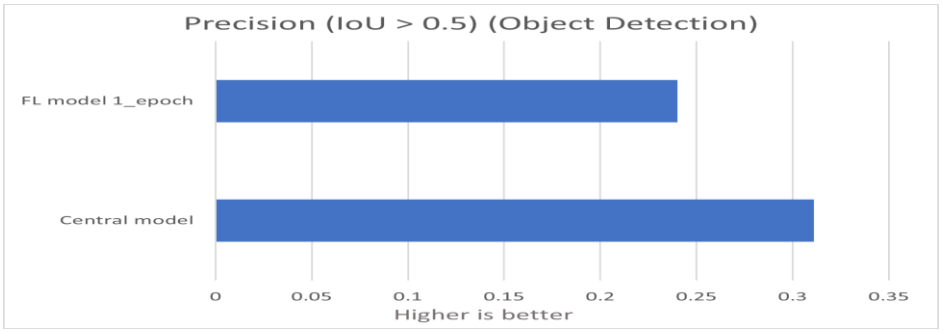


Figure 16. Performance Analysis of object detection using Test Set

From Figure 16, it can be observed that the centrally trained model performed better than the FL model which sent an update after every epoch.

6. CONCLUSION

Federated Learning is an eminent field of research which is relatively new and upcoming in the field of Artificial Intelligence. The very few frameworks that exist support this from the perspective of research and cannot be deployed in a real-world scenario. In this work, a framework that allows for building Federated Learning models in a real-world scenario is developed and explained.

The framework provides support for two common Computer Vision tasks, image classification and object detection. It also provides many different CNN architectures and aggregation algorithms that allow the user to choose models and techniques based on their application. It makes use of configuration files and thus provides an easy interface to the clients. The framework also provides a UI which allows the users to monitor the Federated Learning process and the status of the participants in the process. The framework is benchmarked with 2 medical datasets, one for each task, considering both IID and Non-IID data.

For the case of image classification, it was observed that the FL model that trained on IID Data and sent an update after every epoch performed similarly to the centrally trained model with a weighted log loss of 0.09. The FL models that trained on Non IID Data and sent an update after 3 epochs performed better than the previously mentioned models. The FL model that trained on Non IID Data used the UGA technique resulted in a reduced weighted log loss of 0.08 and hence its performance is better than centrally trained model and FL without UGA technique. For the case of object detection, it was observed that the centrally trained model performed slightly better (0.311) than the FL model (0.240) which sent an update after every epoch.

As part of the future work, the framework can be extended to accommodate a lot of other domains. Furthermore, support for different tasks, such as Image Segmentation, and other algorithms and architectures can be implemented and added to the FL framework.

REFERENCES

[1] O. Russakovsky, J. Deng, H. Su, J Krause, S. Satheesh, S. Ma, SS. Ma and L. Fei-Fei, **Imagenet Large Scale Visual Recognition Challenge**, *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211-252, 2015.

[2] C. Bycroft, C. Freeman, D. Petkova, G. Band, L. T. Elliott, K. Sharp, A. Motyer, D. Vukcevic, O. Delaneau, and J. O'Connell, **The UK Biobank Resource with Deep Phenotyping and Genomic Data,** *Nature*, vol. 562, no. 7726, pp. 203–209, 2018.

[3] M. Shaheen M. S. Farooq, T. Umer and B. S. Kim, **Applications of Federated Learning; Taxonomy, Challenges, and Research Trends,** *Electronics*, vol. 11, no. 4, pp. 1-33, 2022.

[4] D. Ng, X. Lan, M. M Yao, W. P. Chan and M. Feng, **Federated Learning: A Collaborative Effort to achieve better Medical Imaging Models for Individual Sites that have Small, Labelled Datasets,** *Quant Imaging Med Surg*. vol. 11, no. 2, pp. 852-857, 2021.

[5] J. M. Sheller, B. Edwards, G. Anthony Reina, J. Martin, S. Pati, A. Kotrotsou, Mikhail Milchenko, Weilin Xu, Daniel Marcus and Rivka R. Colenand Spyridon Bakas, **Federated Learning in Medicine: Facilitating Multi-Institutional Collaborations without Sharing Patient Data**, *Scientific reports*. vol. 10, no. 1, 2020.

[6] K. V. Sarma, S. Harmon S, T. Sanford, H. R. Roth, Z. Xu, J. Tetreault, D. Xu, M. G. Flores, A. G Raman, R. Kulkarni and B. J. Wood, **Federated Learning improves Site Performance in Multicenter Deep Learning without Data Sharing**, *Journal of the American Medical Informatics Association,* vol. 28, no. 6, pp. 1259-1264, 2021.

[7] T. Naeem Anees, R. A. Naqvi and W. K. A. Loh, **A Comprehensive Analysis of Recent Deep and Federated Learning based Methodologies for Brain Tumor Diagnosis,** *J. Pers. Med*. vol. 12, no. 2, 2022.

[8] I. Dayan, H. R. Roth, A. Zhong, A. Harouni, A. Gentili, A. Z. Abidin, A. Liu, A. B. Costa, B.. J. Wood, C. S. Tsai and C. H. Wang, **Federated Learning for Predicting Clinical Outcomes in Patients with COVID-19,** *Nature Medicine,* . vol. 27, pp. 1735-1743, 2021.

[9] A, Rahman, M. S. Hossain, G. Muhammad et al., **Federated Learning-based AI Approaches in Smart Healthcare: Concepts, Taxonomies, Challenges and Open Issues,** *Cluster Computers*, 2022. https://doi.org/10.1007/s10586-022-03658-4

[10] M. J. Sheller, B. Edwards and G. A. Reina, **Federated Learning in Medicine: Facilitating Multi-institutional Collaborations without Sharing Patient Data,** *Scientific Reports*, vol. 12598, pp. 1-14, 2020.

[11] N. Rieke, J. Hancox and W. Li, **The Future of Digital Health with Federated Learning,** *Digital Medicine*, vol. 119, pp. 1-7, 2020.

[12] S. M. Smith, D. Vidaurre, F. Alfaro-Almagro, T. E. Nichols, and K. L. Miller, **Estimation of Brain Age Delta from Brain Imagin**g, *Neuroimage*, vol. 200, pp. 528–539, 2019.

[13] M. Nieuwenhuis, H. G. Schnack, N. E. Van Haren, J. Lappin, C. Morgan, A. A. Reinders, D. Gutierrez-Tordesillas, R. Roiz-Santiañez, M. S. Schaufelberger and P. G. Rosa, **Multi-center MRI Prediction Models: Predicting Sex and Illness Course in First Episode Psychosis Patients**, *Neuroimage*, vol. 145, pp. 246–253, 2017.

[14] D. Kondor, B. Hashemian, Y. A. De Montjoye, and C. Ratti, **Towards Matching User Mobility Traces in Large-scale Datasets**, *IEEE Transactions on Big Data*, vol. 6, no. 4, pp. 714–726, 2018.

[15] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, **Communication-efficient Learning of Deep Networks from Decentralized Data**, *Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.
[16] R. Wightman, (2020, January 17), **Pytorch Image Models**, https://github.com/rwightman/pytorch-image-models
[17] M. Tan, and Q. Le, **Efficientnet: Rethinking Model Scaling for Convolutional Neural Networks**, *Proceedings of International Conference on Machine Learning*, pp. 6105–6114, 2019.
[18] M. Shafiq, Z. Gu, **Deep Residual Learning for Image Recognition: A Survey,** *Applied Sciences*, vol. 12, no. 18, 2022.
[19] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, **Aggregated Residual Transformations for Deep Neural Networks**, *Proceedings of the IEEE Conference On Computer Vision and Pattern Recognition*, pp. 1492–1500, 2017.
[20] X. Yao, T. Huang, R. X. Zhang, R. Li, and L. Sun, **Federated Learning with Unbiased Gradient Aggregation and Controllable Meta Updating**, arXiv preprint arXiv:1910.08234, 2019.
[21] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. Kalinin, **Albumentations: Fast and Flexible Image Augmentations**, *Information*, vol. 11, no. 2, pp. 125, 2020.
[22] D. P. Kingma, and J. Ba, **Adam: A Method for Stochastic Optimization**, arXiv preprint arXiv:1412.6980, 2014.
[23] S. Ren, K. He, R. Girshick, and J. Sun, **Faster RCNN: Towards Real-time Object Detection with Region Proposal Networks**, arXiv preprint arXiv:1506.01497, 2015.
[24] D. Bolya, S. Foley, J. Hays, and J. Hoffman, **Tide: A General Toolbox for Identifying Object Detection Errors**, arXiv preprint arXiv:2008.08115, 2020.
[25] Z. Xiao, X. Xu, H. Xing, F. Song, X. Wang, and B. Zhao, **A Federated Learning System with Enhanced Feature Extraction for Human Activity Recognition**, *Knowledge-Based Systems*, vol. 229, 2021.