Federated Learning for brain segmentation

Alexander Samoylenko
Optimization Class Project. MIPT

Introduction

Deep learning methods require a large amount of data for training. This problem is relevant for medical images, firstly, due to the relatively small number of recorded cases of a specific disease in a particular medical institution, and secondly, because labelling of medical images requires professional knowledge . A possible solution to this problem could be the joint work of several medical centers, however, centralizing the data on one machine or in a datacenter is often impossible because of a number of technical and legal restrictions. Due to these limitations, in recent years Federated learning, a concept of distributed learning that does not require centralized data storage, is gaining popularity. In this paper, we are comparing performance of neural networks trained using Federated learning approach and networks trained in the classical way on the example of brain segmentation task.

Concept

Consider an architecture where training data is separated between n servers each of which may only interact with an aggregation server. In this case single training epoch for Federated Learning consists of the following parts:

- Send copies of centralized model to every data server;
- Train every model independently on local training data;
- Send locally trained models back to the aggregation server;
- Average weights of local models to obtain updated centralized model.

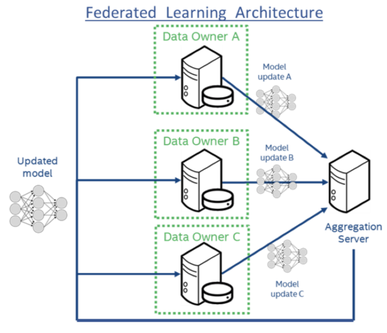

Figure 1: System architecture of Federated Learning

Problem

We search for a function f (neural network) that maps input brain MRI images x to corresponding binary masks y:

$$f : x \to y$$

For the training data we have MRI scans taken from 6 different MRI scanners (Siemens 1.5T/3.T, GE 1.5T/3T, Phillips 1.5T/3T). Data is split into 6 domains in accordance with MRI scanners so that every domain corresponds to single data server where local training is conducted.

As an architecture of neural network we use U-Net that showed promising results in various segmentation tasks with lack of training data.

Figure 2: Different data domains

Algorithm

Data server

```
model = Recv(aggregation_server);
for epoch = 1:local_epochs
    for batch in local_batches
        train_model(model, batch);
Send(model, aggregation_server);
```

Aggregation server

```
for epoch = 1:global_epochs
    models = [];
    for server in data_servers
        Send(model, server);
    for server in data_servers:
        updated_model = Recv(server);
        models.append(updated_model);
    model = average_weights(models)
```

Averaging techniques

Two averaging techniques were tested which are:

1. Dataset size based averaging (DS averaging);
2. Validation loss value averaging (VL averaging).

   Below the we can see graphics representing training process for both techniques. VL avegaing seemed to explode at the last epochs.

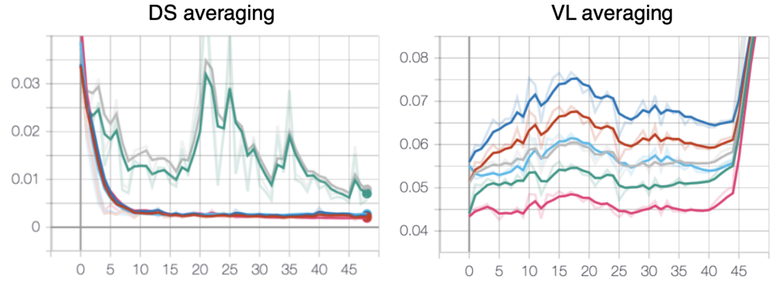

Figure 3: Averaging techniques

Training process

For model performance comparison we end up with training 2 models: Federated U-Net and straightforward U-Net with no data separation.

All of the models were trained using Adam optimizer with initial learning rate of $5 \cdot 10^{-3}$. Federated U-Net was trained for 50 global rounds and 3 local epochs with LR-Scheduler multiplying learning rate by 0.5 on each of the following global rounds: 10, 20, 30, and multiplying by 0.1 on 40 and 45 epochs.

Model with no Federated approach was trained for 200 epochs with LR-Scheduler multiplying learning rate by 0.5 every 30 epochs.

All models were created and trained using PyTorch framework.

Results

The first table shows difference in models training time.

	Non-Federated	Sequential Federated	Parallel Federated
Time, hours	30	20	8

Test metrics of Federated and non-Federated U-Net for every image domain are represented below.

Algorithm	Dice Score						
	Dom. 1	Dom. 2	Dom. 3	Dom. 4	Dom. 5	Dom. 6	Overall
Non-Federated U-Net	0.9892	0.9955	0.9954	0.9959	0.9959	0.9970	0.9949
Federated U-Net	0.9945	0.9951	0.9955	0.9955	0.9809	0.9824	0.9907

The next table shows relative change of Dice Score in comparison with Non-Federated U-Net.

Algorithm	Dice Score change						
	Dom. 1	Dom. 2	Dom. 3	Dom. 4	Dom. 5	Dom. 6	Overall
Non-Federated U-Net	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Federated U-Net	+0.54%	-0,04%	+0,01 %	-0,04%	-1,50 %	-1,46 %	-0.42%

We can see that Federated U-Net shows comparable results and achieves 99.5 % of the model performance of a data-sharing model (non-Federated). Code for Federated model training is available here `https://github.com/samoyl11/FederatedUNet`here.

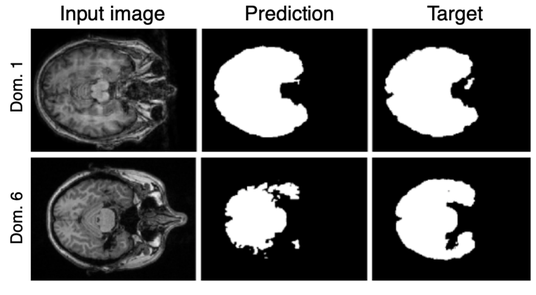

Figure 4: Examples of Federated model prediction

Conclusion and futher steps

Our experiments demonstrate that it is possible to train a model with comparable results without data-sharing. The futher step is increasing number of domains (adding coronal and sagittal projections).