A REPORT

ON

THE APPLICATIONS OF SPLIT LEARNING IN HEALTHCARE, THREATS TO DECENTRALIZED LEARNING AND DEFENSIVE TECHNIQUES

By

Name of the student	ID No.	Discipline
Aadit Deshpande	2019A7PS0077P	Computer Science

Prepared in the fulfilment of the

Design Project Course (EEE F376)

At IOT Lab, BITS Pilani

with Prof. Vinay Chamola

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

May, 2023

TABLE OF CONTENTS

1. INTRODUCTION

Deep Learning has massive applications in healthcare particularly in imaging analytics and diagnostics. Training neural networks requires massive amounts of data and computational power. The healthcare domain additionally poses a unique challenge – that of privacy. While multiple organizations (like hospitals, universities, medical schools etc.) might collaborate to train deep learning models, they would be hesitant to share sensitive data about patients, especially demographic characteristics, medical images and so on. In this regard, Split Learning appears to be a viable decentralized solution to the privacy problem.

Split learning is a decentralized deep learning paradigm, wherein multiple client models send their activations (not the input data) to a server model which completes the processing and passes back the gradients for the client models' backpropagation. Split learning is successful both due to its effectiveness and efficiency (Chang et. al, 2021 **[3]**). However, split learning presents its own challenges, as malicious actors may try to disrupt the training process of the clients, or attempt to recover the input data from the clients' activations (thus, nullifying the privacy-preserving benefits of split learning).

In this project, we analyse two split learning models – the ResNet and the U-Net on two datasets of medical images (for classification and semantic segmentation respectively). Experiments are conducted to observe if altering the number of clients and the position of the 'split' affect the performance on the two datasets above. Further, we explore two threat models to the split learning paradigm – Feature Space Hijack Attack (FSHA) and Adversarial Reconstruction Attack (ADRA), as well as defensive techniques to prevent reconstruction of input images by malicious servers. Thus, the major contributions of this project are as follows: 1) We propose a novel split learning architecture for the U-Net 2) We study whether hyperparameters like the cut layer or number of clients affects task performance. 3) We explore two threat models to split learning and the respective defensive techniques.

The rest of this report is structured as follows. First, in the Methodology section, we discuss the model architectures for the Split ResNet and the Split U-Net. This is followed by a formalization of the two threat models (FSHA and ADRA) and their defensive techniques. Next, in the Experiments section, we describe the hyperparameters for experiments run on the Diabetic Retinopathy and Chest X-ray datasets with the Split ResNet and Split U-Net respectively. This is followed by the Results section, which briefly discusses the salient findings of the project. Finally, the conclusion summarizes the project and outlines the further work.

2. METHODOLOGY

2.1. Split Learning

Split learning allows decentralized neural network training by partitioning the data vertically. In the simplest case (one client model and one server model), the client model ***f*** owns the first n layers of the model, whereas the server model ***s*** owns the remaining layers. The client model passes the activations of its private dataset ***X***, ***f(X)*** to the server (and not X itself)**.** The server further performs its computation ***s(f(X))***, calculates the task loss and passes the gradients back to the client, which can then complete its backpropagation. In the simplest case, the client shares the labels with the server, although methods without label-sharing have been devised as well. The server has no decision-making power with respect to model architecture or the hyperparameters. This paradigm can be easily scaled to multiple clients, with the clients following a round0robin protocol as they interact with the server (as shown above). After its turn the client then simply passes its model weights to the next client.

2.2. Split ResNet Architecture

In this paper, we use the Split ResNet architecture for binary classification of a Diabetic Retinopathy dataset. The Residual Neural network (or the ResNet) is characterized by its skip connections and is particularly useful for classification tasks. Residual Networks enable deep learning models with hundreds of layers to improve on task metrics without depth constraints. In this study, we use the PyTorch implementation of *ResNet-34* in particular, without its pretrained weights. ***Fig 1.*** shows the successive activations of each residual block of the Client ResNet Model on the input images.

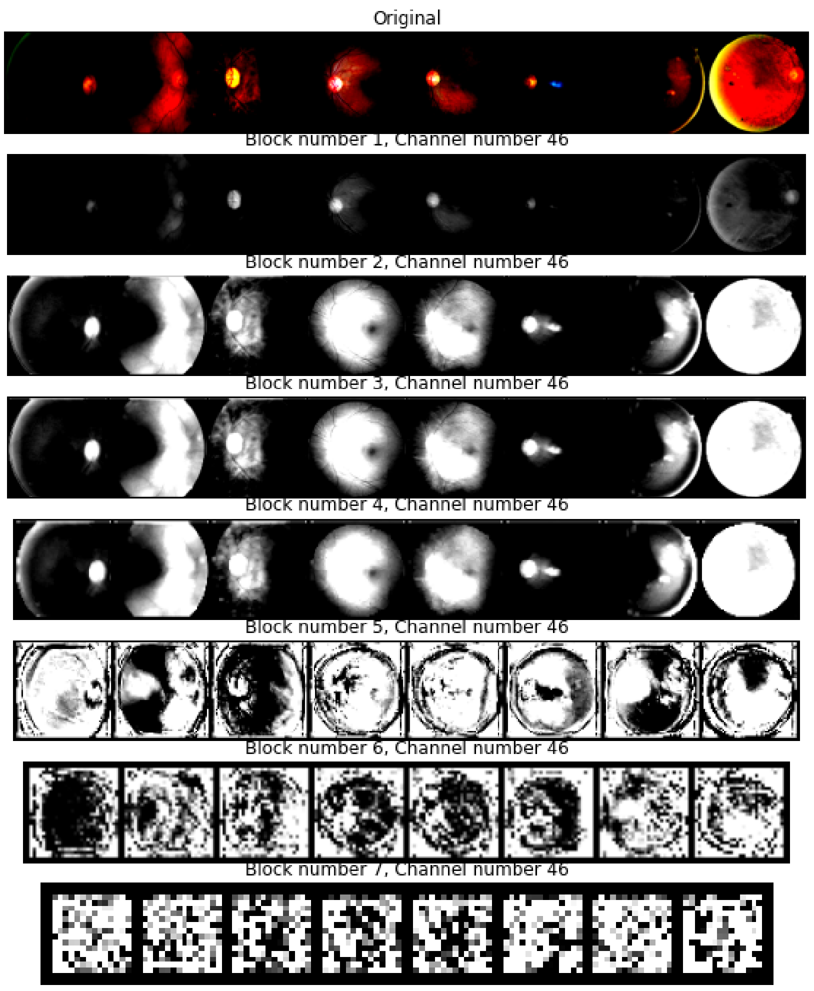
Original
Block number 1, Channel number 46
Block number 2, Channel number 46
Block number 3, Channel number 46
Block number 4, Channel number 46
Block number 5, Channel number 46
Block number 6, Channel number 46
Block number 7, Channel number 46

Fig 1. Successive Activations of the Input image of the Client ResNet Model

2.3. Split U-Net Architecture

We propose a new Split learning version of the U-Net for Semantic Segmentation on the Lung Segmentation from Chest X-Ray Dataset (Kaggle Competition, 2018). Here, the client model ***f*** is further split into the Client Encoder, ***fe*** and the Client Decoder ***fd***. Thus, the server performs the intermediate compression and converse transpose computations (encoding and decoding) and returns its activations back to the client. Thus, the output masks from the input dataset of images ***X*** is represented by ***fd(s(fe(X)))***. Interestingly, the server has no knowledge of the segmentation masks, unlike the case of the Split ResNet, where the server required knowledge of the labels to calculate the loss. However, in the Split U-Net, the client decoder itself produces the predicted mask, and not the server. Again, the server has no decision-making power in the architecture or hyperparameters of the model.

In this project, we build a U-Net from scratch using convolutional blocks of two Conv2d and two BatchNorm layers followed by a Max Pooling layer (using PyTorch). The 'split' or 'cut' layer can be changed by symmetrically allowing the client encoder to own the first n layers of the U-Net, the client decoder to own the last n layers, and the server to own the remaining intermediate layers..

2.4. Threat Model 1: Feature-Space Hijacking Attack (FSHA)

The FSHA attack first proposed by Pasquini et al. (2021) **[1]** attempts to reconstruct the private input data ***X_priv*** using the clients activations by altering its feature space. The FSHA assumes that: 1) The attacker has no information about the clients (architecture or model weights), 2) The attacker ignores the client's target task, 3) The attacker has no access to the private data, but does have access to a similar dataset ***X_pub***.

The attack is performed in two phases. First, in the Setup phase, the attacker trains two models ***f1*** (encoder) and ***f2*** (decoder), along with a discriminator network ***D***. The attacker's two goals are to veer the client model's feature space similar to its own model ***f1*** by training D and then to train ***f2*** to be a good inverse function of ***f1***. Next, in the Inference phase, the attacker simply uses the activations from the hijacked client network ***f*** to reconstruct the input images as follows: ***f2(f(X_priv))***.

The defensive technique for this threat model is known as 'Distance Correlation Minimization' or DCOR. The logic is for clients to produce activations that are different from the input, making it more difficult for the attackers to reconstruct data from its activations. The client does so by using a modified loss function which is a weighted sum of the task loss as well as the distance between the input and its activation.

2.5. Threat Model 2: Adversarial Reconstruction Attack (ADRA)

The ADRA attack proposed in the Workshop on Split Learning for Distributed Machine Learning, 2021 **[2]**, also attempts to reconstruct the private input data ***X*** using the clients activations using an adversarial model. The ADRA assumes that: 1) The attacker does not influence the client's training process and trains its own adversarial model independently (unlike FSHA). 2) The attacker does have access to the private data, but only to calculate its reconstruction loss.

The attacker exploits the client's activations to reconstruct the original data by training an adversarial model. The defensive technique for this threat model involves the use of a Proxy Adversary Network during the client's training process. Similar to DCOR, the client modifies its loss function to minimize its own loss, and simultaneously maximize the attacker's loss, to worsen the attacker's reconstructions. This is achieved by including the adversarial inputs' gradients while the server passes back the gradients to the client model.

3. EXPERIMENTS

Two variables were assessed in the following experiments with the Split learning models. First, the 'Cut Layer' which represents the number of layers of the model the Client model owns. Smaller values of cut layer imply that the majority of the computation happens on the Server model. Second, we observed the number of clients in the training process. More clients naturally means that more data will be processed during the training (all the clients use datasets that are mutually exclusive).

3.1. ResNet (Cut Layer)

We used the ResNet34 for both the client and server models. We randomly sampled 1000 training images and 200 test images of the roughly 35K images in the original Diabetic Retinopathy dataset. The labels were converted to a 0/1 binary classification and were normalized using the ImageNet transforms. The ResNet-34 models used Cross Entropy loss functions and Adam Optimizers (learning rate = 0.01). The Client and Server models were subsequently trained for 20 epochs and the experiment was repeated for Cut Layer = 2, 3, 5, and 7. **Table 1** shows the accuracies and train and test times for the above experiments. Additionally, ***Fig. 2*** shows the running losses for the Client and Server models' training process.

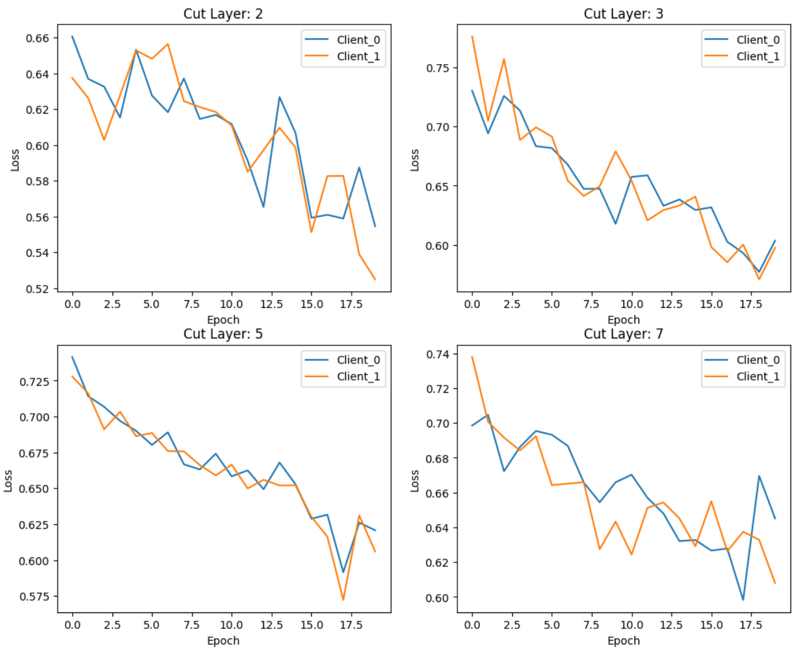

Fig 2. Split ResNet Experiments with different Cut Layers

Table 1. Results for Experiments on Split ResNet and different Cut Layers

Cut Layer	Train Images	Test Images	Epochs	Test Accuracies	Train Time	Test Time
2	1000	200	20	[63.5, 60.25]	1384.26	14.71
3	1000	200	20	[69.4, 69.45]	1690.03	66.95
5	1000	200	20	[68.7, 62.85]	1450.43	159.03
7	1000	200	20	[70.8, 62.5]	1458.54	97.26

3.2. ResNet Num_Clients

We used the same hyperparameters for the ResNet34 client and server models. This time, instead of just a single client, we additionally used n_clients = 2, 3, 4, and 6. We randomly sampled 1000 training images and 200 test images for each of the client models of the roughly 35K images in the original Diabetic Retinopathy dataset. The labels were converted to a 0/1 binary classification and were normalized using the ImageNet transforms. The ResNet-34 models used Cross Entropy loss functions and Adam Optimizers (learning rate = 0.01). The Client and Server models were subsequently trained for 20 epochs and the experiment was repeated for Cut Layer = 2, 3, 5, and 7. **Table 2** shows the accuracies and train and test times for the above experiments. Additionally, ***Fig. 3*** shows the running losses for the Client and Server models' training process.

Table 2. Results for Experiments on Split ResNet and Different No. of Clients

No. of Clients	Train Images	Test Images	Epochs	Test Accuracies	Train Time	Test Time
1	1000	200	20	[63.1]	676.73	32.99
2	1000	200	20	[74.3, 64.35]	1335.19	64.26
3	1000	200	20	[74.0, 62.7, 59.97]	2265.60	376.75
4	1000	200	20	[70.4, 70.35, 70.8, 71.05]	697.24	127.89
6	1000	200	20	[89.1, 90.6, 90.333, 90.78, 90.68, 90.6]	4008.59	188.25

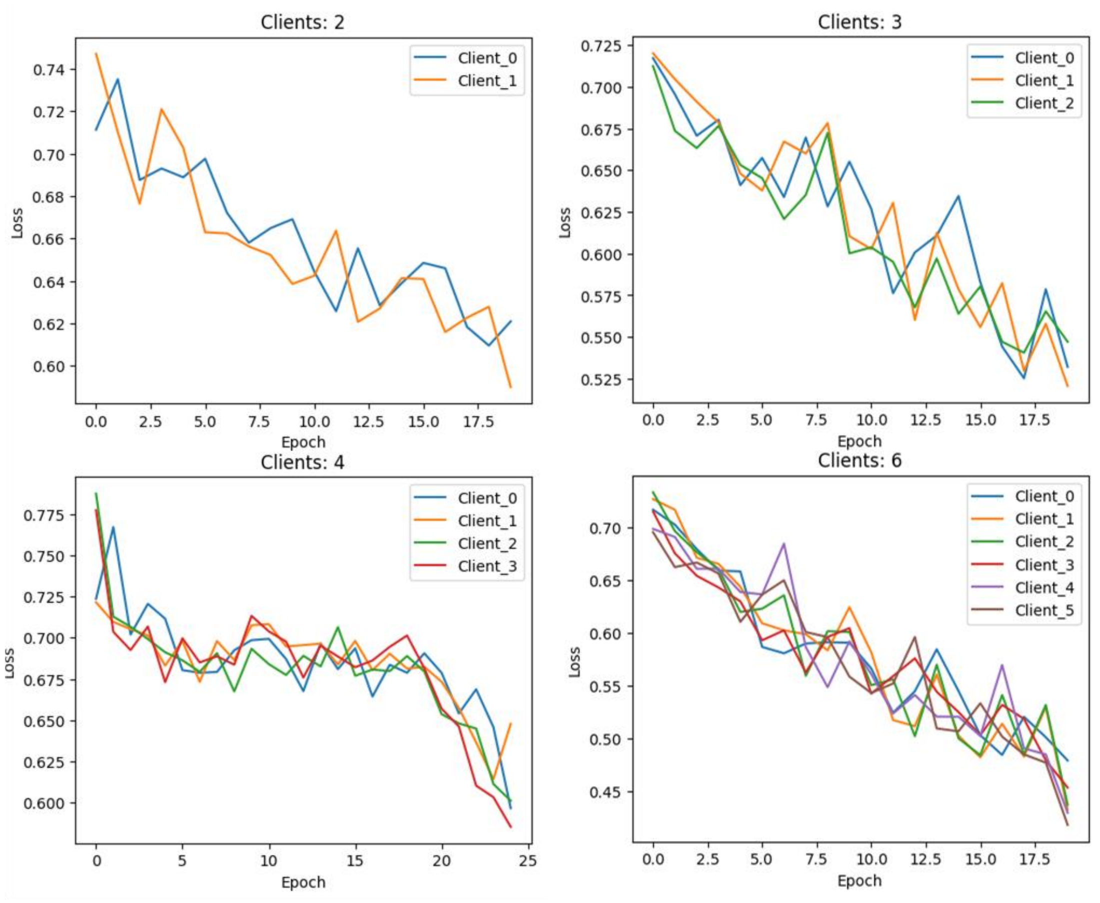

Fig 3. Split ResNet Experiments with different Number of Clients

3.3. U-Net Cut_Layer

The U-Net Split model featured only a single client for simplicity. The clients used a nearly 80:20 train-test split of the 700-image Chest X-ray dataset. The input images are of dimension 228228x3 and the output mask consists of 1 channel only, with dimensions 228x228x1. The U-Net model uses the Dice loss function, along with the Adam optimizer (learning rate = 0.01). The models are trained for 10 epochsm by varying the cut layer (values used are 1, 2, and 3). **Table 3** shows the results for the different values of the cut layers. ***Fig 5.*** shows the output of the trained U-Net models (client and server) and a comparison of the model's prediction and the actual mask. ***Fig. 6***. shows the running losses, accuracies and the jaccards for the different values of cut layer for the client and server.

Table 3. Results for Experiments on Split U-Net and different Cut Layers

Cut Layer	Train Images	Val Images	Epochs	Train Time	Train Accuracies	Val Accuracies	Train Jaccard	Val Jaccard
1	560	140	10	1860.23	0.9820	0.9826	0.9313	0.9314
2	560	140	10	1881.30	0.9822	0.9818	0.9314	0.9301

3	560	140	10	1838.1 1	0.9832	0.9812	0.9351	0.9268

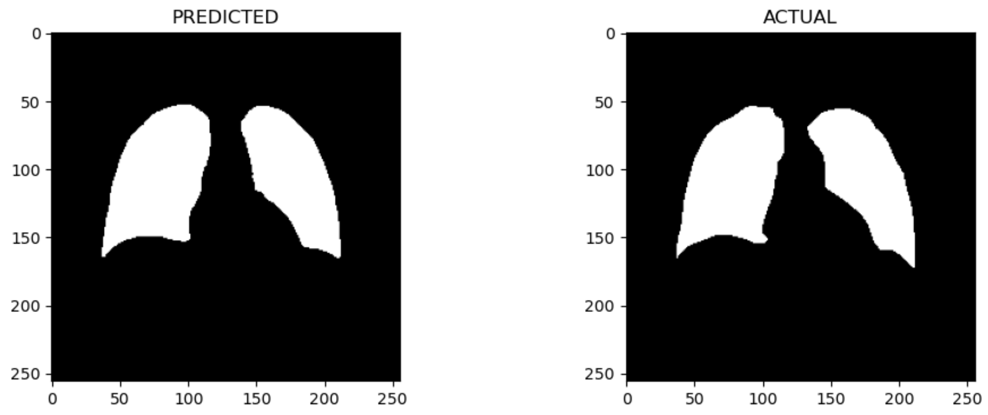

Fig 5. Prediction of the Split U-Net Model (left) and the Actual Mask (right)

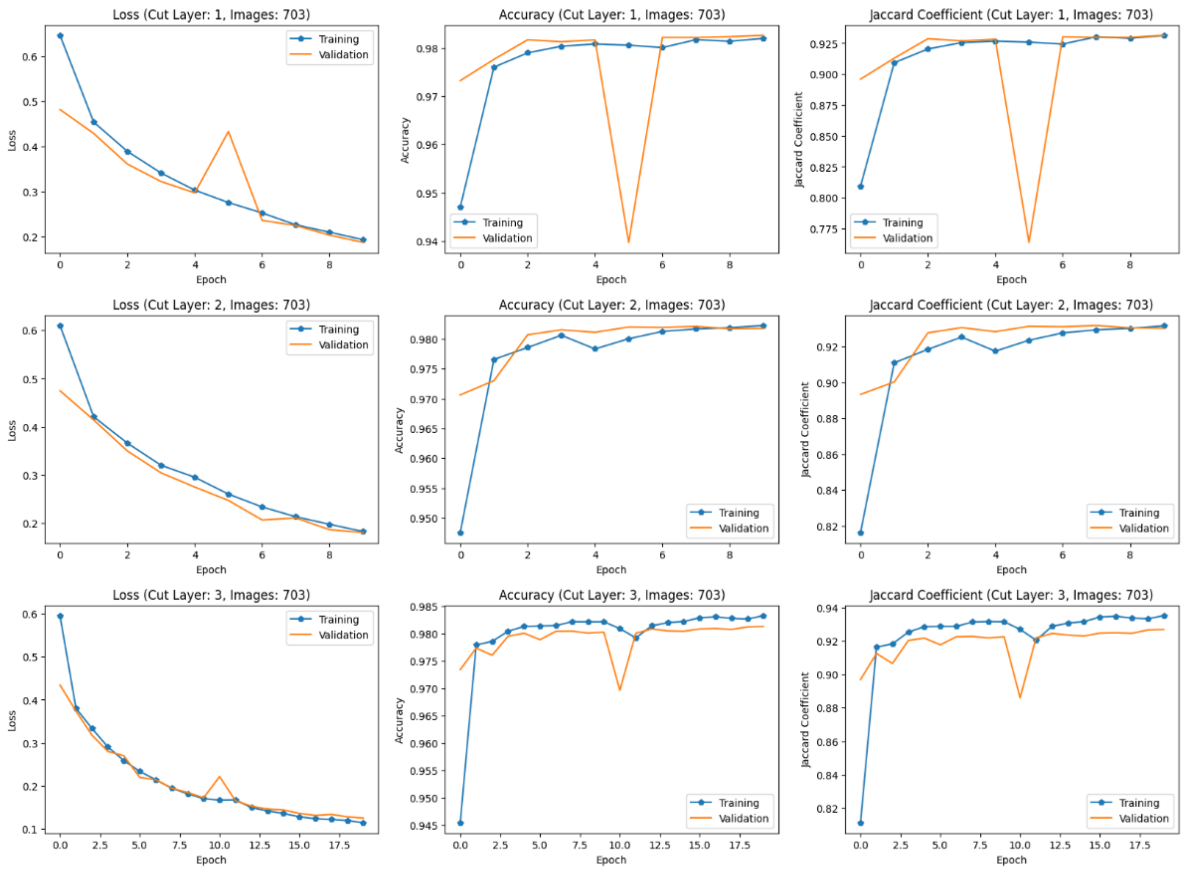

Fig 6. Split U-Net Experiments with different Cut Layers

3.4. Threat Model 1: FSHA

The FSHA model was trained using 1000 training images and 200 test images from the Diabetic Retinopathy dataset. Training took place over 10000 iterations, using the WGAN and a gradient penalty value of 500, with learning rate 0.00001 for the client (and the networks f1 and f2) and

0.0001 for the discriminator (D). ***Fig. 7*** shows the training process and running loss for the client (f) and each of the attacker's models, as well as the final reconstruction error.

The defensive technique DCOR is also used to counter the FSHA attack and hinder its reconstruction. ***Fig. 8*** shows the comparison between the original images, the malicious FSHA reconstruction, and the malicious reconstruction after using DCOR.

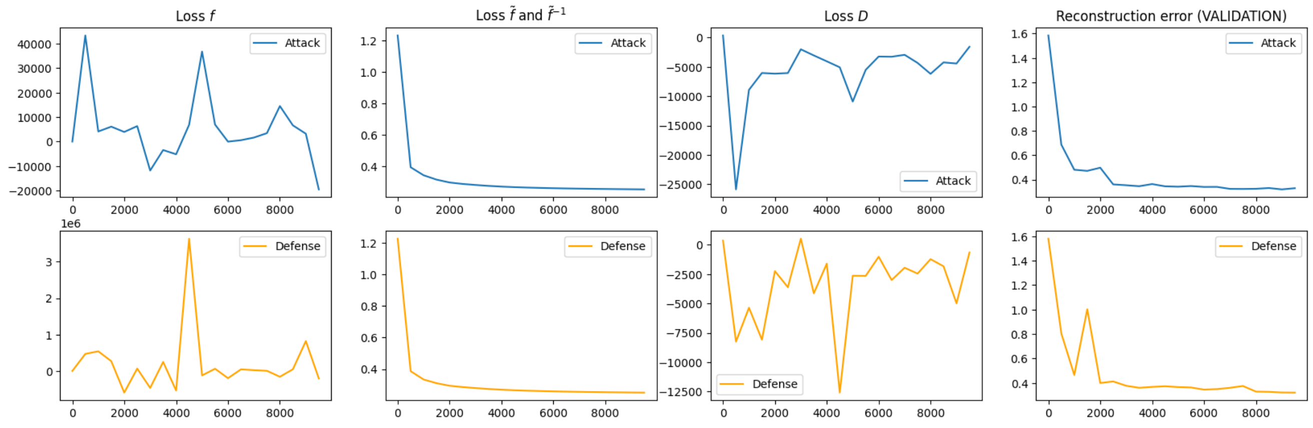

Fig 7. FSHA Model training plots for Adversary and Client

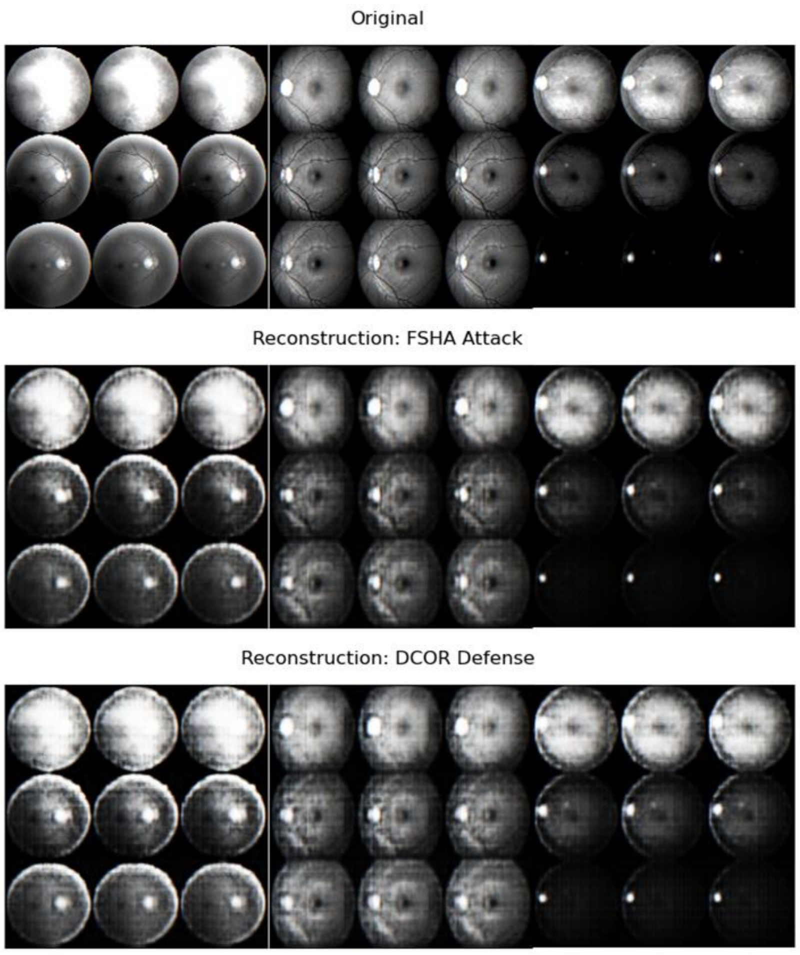

Fig 8. Feature-Space Hijack Attack without and with defensive technique

3.5. Threat Model 2: ADRA

The ADRA model was also trained using 1000 training images and 200 test images from the Diabetic Retinopathy dataset. Training took place over 20 epochs with a client (cut layer = 7) with frozen weights. The adversarial model used the L1 loss as its reconstruction loss and the SGD optimizer with a learning rate of 0.01 and momentum of 0.9. ***Fig. 9*** shows the adversarial loss (left) as well as the adversarial loss after being disrupted by the defensive technique of using the proxy adversary network (right). ***Fig. 10*** shows the comparison between the original images, the malicious ADRA reconstruction, and the malicious reconstruction after using the Proxy Adversarial network.

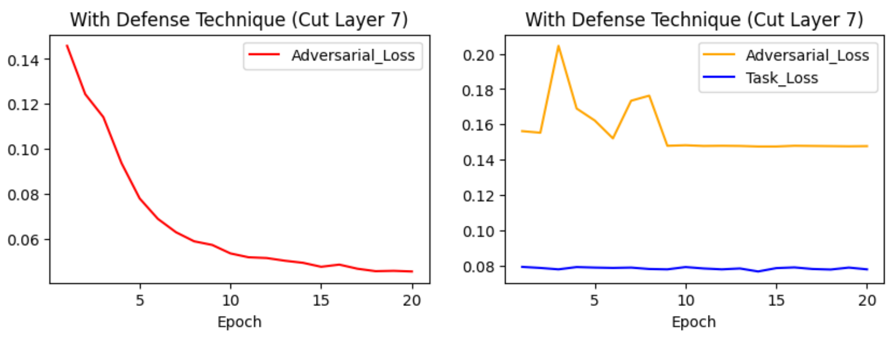

Fig 9. ADRA Model loss plots for Adversary and Client.

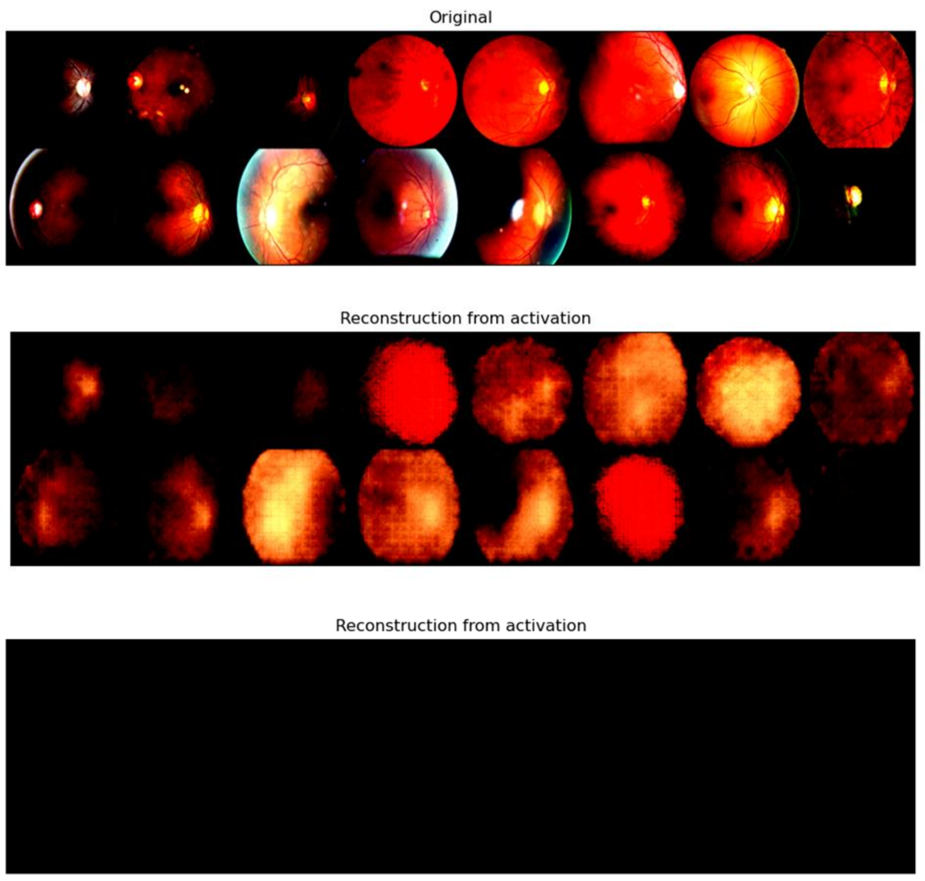
Original
Reconstruction from activation
Reconstruction from activation

Fig 10. Reconstruction attack without and with defensive technique

4. RESULTS

These are the salient findings of the project, from the experiments described above:

- The split learning performance is slightly worse than centralized learning, since each of the clients possess a different mutually exclusive dataset, and they each share the model weights.
- The cut layer has little to no effect on the accuracy of the model. All the cut layer values achieve an accuracy in the 65-70 range, corresponding to the difficulty of the classification task of Diabetic Retinopathy. The cut layer only influences the proportion of computational responsibility taken on by the clients and server.
- More number of clients enhances the training accuracy, since the model weight is trained for more epochs, directly proportional to the number of clients. This is due to the round-robin turn-taking protocol followed in the training process. Naturally, the training time increases proportionally with the number of client models.
- We propose a functioning Split U-Net model architecture in this project, which achieves comparable results to the vanilla centralized U-Net, with a jaccard score of ~0.93 after only 10 epochs of training. For the U-Net, the cut layer again makes little to no difference to the Jaccard (Intersection over union) value or the accuracies.
- With respect to FSHA, the threat model reconstructs the input image well, however, DCOR does not seem effective enough at preventing reconstruction.
- However, the Proxy Adversarial Network is far more effective at countering the ADRA attack, by completely disrupting its training process, to the point where the reconstruction is unsuccessful.

5. CONCLUSION AND FURTHER WORK

In this project we explored the Split learning paradigm in healthcare to address the concerns of privacy in this domain. We implemented the Split ResNet for classification on the Diabetic Retinopathy Kaggle dataset, and also proposed a new Split U-Net model for semantic segmentation on the Chest X-ray dataset. We experimented by altering hyperparameters like cut layer and the number of clients, and found that increased number of clients results in a pronounced improvement on task performance. Additionally, we explored two threat models - FSHA, and ADRA along with their defensive techniques.The threat models attempt to reconstruct the input dataset using feature space hijacking and adversarial attack respectively. We observed that the Proxy Adversary Netwrok defense technique was far more effective at disrupting the malicious attack. Additionally, hardware such as the NVIDIA Jetson can be used to accelerate the training process and achieve even better results. Finally, as this work focuses on privacy-preserving machine learning, a natural next step is the integration of blockchain.

6. REFERENCES

1. Pasquini, Dario, Giuseppe Ateniese, and Massimo Bernaschi. "Unleashing the tiger: Inference attacks on split learning." In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2113-2129. 2021.
2. 2021. *Workshop on Split Learning for Distributed Machine Learning (SLDML'21)*. https://splitlearning.github.io/workshop.html. (2021).

3. Chang, Ken, Niranjan Balachandar, Carson Lam, Darvin Yi, James Brown, Andrew Beers, Bruce Rosen, Daniel L. Rubin, and Jayashree Kalpathy-Cramer. "*Distributed deep learning networks among institutions for medical imaging.*" Journal of the American Medical Informatics Association 25, no. 8 (2018): 945-954.