Imperial College London

MEng Individual Project Interim Report

Imperial College London

Department of Computing

Federated Deep Learning for Healthcare Data

Author:
Erik Babu

Supervisor:
Prof. Daniel Rueckert

Second Marker:
Dr. Bernhard Kainz

February 10, 2020

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 2015, a study [1] was conducted on the accuracy of cancer diagnoses. As part of the experiment, 16 testers had to decide whether or not images of breast tissue were cancerous. Independently, the testers correctly assessed 85% of the samples. By pooling the results, i.e. performing majority voting on the independent classifications, the accuracy rate rose to 99%. This experiment was remarkable, not only because of the testers' performance, but also their identity. They were neither oncologists nor pathologists; they were pigeons. Identifying patterns in medical data is not a uniquely human skill.

Because machine learning (ML) algorithms excel at uncovering patterns from data, they have the potential to address problems in healthcare, such as diagnosis [2] and prediction of future health outcomes [3]. These algorithms, particularly deep learning approaches, typically require large training sets to achieve good performance. Labelling medical data requires expert knowledge. Ideally, medical institutions could address this challenge through collaboration i.e. aggregating their anonymised data and annotations to a central location. This would facilitate the creation of sufficiently large datasets, since the healthcare system generates approximately one trillion gigabytes of data each year, and this amount is doubling every two years [4]. However, due to the sensitive nature of healthcare data, there are ethical concerns [5] and data protection regulations such as the General Data Protection Regulation (GDPR) [6] and Health Insurance Portability and Accountability Act (HIPAA) [7] which preclude its sharing, especially among international institutions.

To make use of the large amounts of data available while still preserving privacy, we propose the use of federated learning (FL); a decentralised solution which "brings" a model to the data, rather than data to a model. FL addresses the issues of privacy and data ownership, since the healthcare data never leaves the institution.

1.1 Objectives

The main objective of this project is to develop a FL solution, for **classification** and **segmentation** of medical imaging data, which performs similarly to its centralised implementation i.e. the model which has access to the aggregated data from every institution.

With respect to medical imaging, we will use two datasets. Firstly, the CheXpert (Chest Expert) [8] dataset; a collection of 224,316 chest radiographs (2D) of 65,240 patients labeled for the presence of 14 typical chest radiographic observations. The test set, comprised of 500 studies from 500 unseen patients, is annotated by eight board-certified radiologists. By evaluating our solution against the best available ground truth, we provide a foundation for the clinical relevance of our results.

To verify the generality of our solution on medical imaging data, we will also evaluate its performance on the Brain Tumour Segmentation (BraTS) [9, 10, 11] dataset; a collection of 65 MRI scans (2D and 3D) of patients with low-grade (non-cancerous) and high-grade (cancerous) brain tumours. These scans, referred to as the clinical image data, were annotated by a trained team of radiologists, to obtain the best available ground truth. The dataset also contains synthetic image

data; 65 MRI images and ground truth which have been artificially generated using software.

Although the focus of the project is on medical imaging data, we will also show the generalisation of our FL solution on other imaging tasks by evaluating it on two datasets unrelated to medical imaging. The first dataset, FEMNIST, is a federated version of the the Extended MNIST dataset [12, 13, 14] dataset; a database of over 800, 000 handwritten digits, lowercase and uppercase letters. In this federated version, the dataset is partitioned based on the writer of the digit/character. The main benefit of using this dataset is that it provides a fairly simple classification task; it will allow us to quickly prototype an initial end-to-end FL solution. For a more robust challenge, we will evaluate our solution on the FedVision - Street Dataset [15]; a compilation of over 900 images, representing seven categories and annotated with detailed bounding boxes, captured by 26 different street cameras. A benefit of using this dataset is that the data distribution is non-IID and unbalanced; characteristics that represent real-world FL settings. Since the CheXpert and BraTS datasets do not distinguish by radiologist / institution, artificial partitions will be created to simulate a federated environment.

After achieving similar performance to current state-of-the-art techniques, a stretch goal of the project is to investigate the trade-off between data privacy and model performance, through the incorporation of ε-Differential Privacy (ε-DP) and Secure Multi-Party Computation (SMPC). These concepts are further discussed in Chapter 2.11.

1.2 Challenges

During the project, we expect the following to be significant challenges:

1.2.1 Regarding Classification/Segmentation of Medical Images

- Lack of domain knowledge: Although we are only developing federated implementations of existing object detection, image segmentation and classification algorithms, a stronger understanding of how to interpret brain and chest scans may be required. This is because certain state-of-the-art implementations may leverage domain-specific knowledge to enhance their algorithms' performance. Sufficient domain knowledge would potentially assist us with adapting an implementation to perform well in a federated setting, or enable us to make informed justifications as to why it may not be possible.

1.2.2 Regarding Implementation

- Training times: Due to the size of the CheXpert dataset (∼11GB) and likely complexity of the deep neural networks developed, models could take anywhere between hours and days to train. With 3D data, as found in the BraTS dataset, models will take even longer to train. Since we will be evaluating our solution on several datasets, time management will be pivotal.

- Scaling the solution with number of institutions: For the scope of this project, we will assume that no more than 15 medical institutions will be collaborating. We need to ensure that the performance of our proposed solution does not significantly degrade as more institutions join as participants. The BraTS and CheXpert datasets will also have to be artificially partitioned such that they represent realistic data distribution scenarios, to ensure experimental relevance.

- Trade-off between data utility and privacy: We would like a solution that utilises as much data as possible, while offering some formal privacy guarantees.

In Chapter 3, we discuss our plan to address these challenges.

Chapter 2

Background

In this chapter, we provide a brief overview on supervised deep learning i.e. developing predictive models based on both input and output data, followed by a discussion of some applications of computer vision in the real-world. We then survey existing image classification, detection and segmentation algorithms. The chapter concludes with an analysis of existing state-of-the-art approaches to privacy-preserving deep learning and the datasets we will evaluate our proposed solutions on.

2.1 Overview

Deep Learning (DL) is a subset of Machine Learning (ML), which is itself a subset of Artificial Intelligence (AI). Inspired by the human brain, DL algorithms are capable of learning from large amounts of data. DL pipelines typically consist of a neural network, loss function and optimisation method. The neural network builds a mapping from the input to the output. The loss function is a quantitative metric that evaluates how well the algorithm models the data. The goal of the optimisation step is to find the parameters which minimise the loss function.

As increasing amounts of training data are being made available and our ability to perform these computationally intensive tasks is continuously improving, DL techniques have become very successful in computer vision tasks; they achieve near-human performance on image classification [16] and segmentation [17] challenges.

2.2 Real-World Applications of Computer Vision

2.2.1 Retail

In 2018, Amazon opened its first Go store [18] - a concept that enables customers to walk in, pick up their groceries and walk out i.e. without having to pay at a register. Computer vision is used to determine if, and by whom, an item has been picked. The cameras, which track shoppers at all times, ensure they are billed appropriately when leaving the store. Large supermarket chains are currently losing billions of pounds to shoplifters [19]; integrating similar technology could significantly reduce the damage.

2.2.2 Automotive

With advances in DL and computer vision, there has been an emergence of autonomous vehicles on the road. Waymo [20] and Tesla [21] use computer vision for their driver-less software. These algorithms analyse input from 360 degree cameras and other sensors to control the vehicle. The vehicles are able to manoeuvre through different weather conditions, terrains and traffic scenarios. This technology is not only relevant to personal vehicles; it is becoming increasingly applied to logistics [22] and public transport [23].

The World Health Organisation report [24] that human error and lack of attention cause the majority of deaths from traffic accidents. They predict that in a decade, such incidents will

become the seventh leading cause of death, unless action is taken. The integration of autonomous vehicles on the road will go a long way towards preventing that forecast from becoming a reality.

2.2.3 Personal Security

In the last few years it has become possible to unlock mobile phones through facial recognition. This is possible because of computer vision algorithms that are able to accurately identify and authenticate the owner of the phone under different camera angles, variable distances and contrasting lighting.

Internet of Things (IoT) devices are becoming more prevalent in our day to day lives. There are now smart CCTV surveillance cameras [25] that use computer vision to alert homeowners of any possible threats. These smart-cameras identify movement, detect if it is caused by a human and apply facial recognition to determine if the person is a friend/family of the owner, or an intruder.

2.2.4 Healthcare

This will be the main focus-area of the project. As discussed in Chapter 1, computer vision is being applied to problems in medicine. One such application is health monitoring; Gauss Surgical uses Triton [26], a computer vision-aided platform to monitor blood loss during surgeries.

Another application - medical diagnosis and prognosis - is an area we are starting to see computer vision algorithms outperform human experts. Just this year, Google Health announced that they have developed an AI system which is as good as doctors are at detecting breast cancer [27].

We do not suggest that these computer vision algorithms will replace doctors anytime soon. Instead, our hope is that they will serve as useful assistants, by performing pre-screening or acting as a second opinion. This would enable the medical professionals to deliver healthcare services as efficiently and accurately as possible. Another benefit is that it is easier to train 100 models than 100 doctors; these computer vision algorithms could potentially offer world-class healthcare in countries where access is typically available at a premium.

2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a form of deep neural network that have significantly contributed to the ability of machines to exceed human performance on imaging challenges. CNNs are similar to multilayer perceptrons (MLPs); they are composed of neurons with learnable weights and biases. The problem with regular MLPs is that they do not scale well for large images. Consider MRI scans, which could have dimensions of 256 x 256 x 12 [28, 29]. In an MLP, a single fully connected neuron in the first hidden layer would therefore have 256 x 256 x 12 = 786,432 weights. As there will most likely be several neurons and layers, the trainable parameters would accumulate quickly and result in overfitting [30].

2.3.1 Structure

Figure 2.1 shows the structure of a CNN used to classify an image. We see that the layers of the CNN arrange neurons in three dimensions. This solution scales well because neurons in a layer are only connected to a small region in the previous layer, unlike in an MLP. CNNs are created by stacking three main types of layers:

Input Layer

This layer holds the raw pixel values of the image; with height h, width w and three channels - R, G, B, using Figure 2.1 as an example.
Input Dimension: h x w x 3

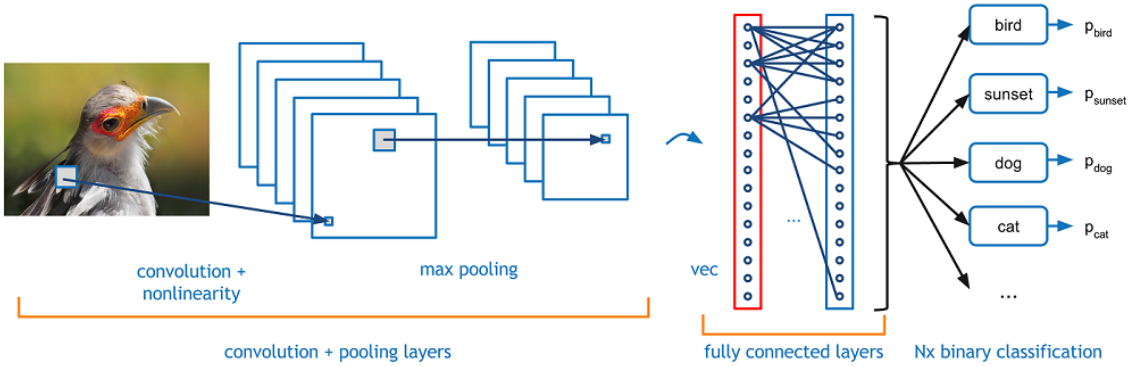


Figure 2.1: Typical CNN architecture [31]

Convolution + Activation Layer

Convolution Layer: Computes the output of the neurons which are only connected to local regions in the input, to produce a feature map. The computation involves "sliding" an n x n kernel/filter over the input and performing element-wise matrix multiplication and summation to get a result. To capture different features, we slide k filters over the input.
Output Dimension: h x w x k

Activation Layer: Applies element-wise activation function to the output of the convolutional layer. This allows the network to model response variables which are calculated using non-linear combinations of the inputs. Typically, Rectified Linear Unit (ReLU) is used as the activation function.

$$ReLU(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$$

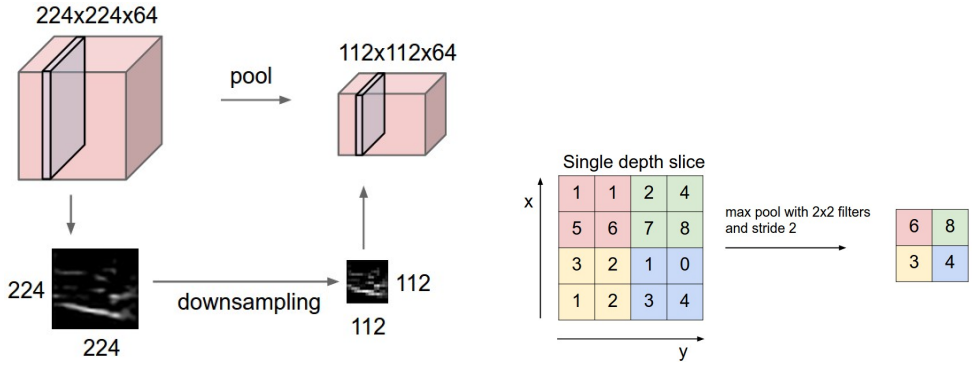
Using other activation functions, such as tanh and sigmoid (detailed in A.1), which map their input to a small output range can introduce the vanishing gradient problem. This is occurs particularly when multiple layers, each using these activation functions, are stacked together. The first layer maps a large input region to a smaller output region, which will be mapped to an even smaller output region by the next layer and so on. This causes the gradients of the network's output with respect to the parameters in the early layers to become extremely small. Multiplying n of these small numbers, to compute gradients of the early layers in an n-layer network during back-propagation, results in the value tending to zero (vanishing) [32]. Hence, the network is unable to train effectively.
Output Dimension: h x w x k

Pooling Layer

This layer is typically placed between successive convolutional layers to gradually reduce the size of the representation along the width and height dimensions. This reduces the number of parameters, and therefore computation in the network, and also prevents overfitting. Figure 2.2 shows that when using a 2 x 2 filter "slid" across the input two at a time i.e. stride = 2, only 25% of the activations are preserved. The most common pooling technique is max-pooling; the function selects the MAX value that the sliding window covers. Other pooling functions such as average pooling and L2-norm pooling have been proposed but do not yet achieve comparable performance.
Output Dimension: $\frac{h}{2}$ x $\frac{w}{2}$ x k (for 2 x 2 pooling filter)

Fully Connected Layer

The fully connected input layer takes the output from the layers before it and "flattens" them into a single vector to be fed as input to the next stage, as shown in Figure 2.1. As with regular MLPs, each neuron in this layer will be connected to all the neurons in the previous one. This layer will



(a) Input pooled with 2 x 2 filter, stride 2 (b) Max pooling with 2 x 2 filter, stride 2

Figure 2.2: Pooling layers downsample spatial volume of input [33]

then produce c class scores to be used for classification.
Output Dimension: 1 x 1 x c

2.4 Image Classification

2.4.1 Overview

Terminology

x - input features i.e. vector representation of images
y - label i.e. meaningful tag telling us what is in the image
Θ - parameters i.e. learnable weights and biases

Goal

Given a training set $T = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$ containing m observations, we would like to train a predictor, h_Θ, such that [34]:

$$\forall i[h_\Theta(\mathbf{x}^{(i)}) \approx y^{(i)}]$$

2.4.2 Popular architecture

The following are CNN-based networks that have achieved state-of-the-art performance on image classification tasks. These networks will also form the building blocks of our federated solutions.

VGGNet

Many other models are built on top of VGGNet[35] or use the idea of representing filters with multiple, smaller ones to cover the same effective area. There are several variants of VGGNet, which differ by the total number of layers the network uses. We describe VGG-19, which has 16 convolutional layers and three fully connected layers, as shown in Figure 2.3.

VGGNet was created because CNNs were becoming deeper and using more parameters, leading to longer training times and more overfitting. The solution: ensure that all convolutional kernels are of size 3 x 3. This was unlike the 11 x 11 and 5 x 5 convolutional kernels used by AlexNet [37], the previous state-of-the-art. This can be done because a 5 x 5 convolutional kernel can be replicated, in terms of receptive field covered, by two stacked 3 x 3 convolutional kernels having stride = 1. Similarly, an 11 x 11 convolutional kernel can be replicated by five stacked 3 x 3 convolutional kernels having stride = 1. As a result, fewer parameters need to be trained: five stacked 3 x 3 convolutional kernels require 5 x 3^2 x $c = 45c$ weights, as opposed to an 11 x 11 convolutional kernel, which requires 11^2 x $c = 121c$ weights.

VGGNet was runner-up in the 2014 ILSVRC [16] classification task, achieving 7.3% top-5 error rate.

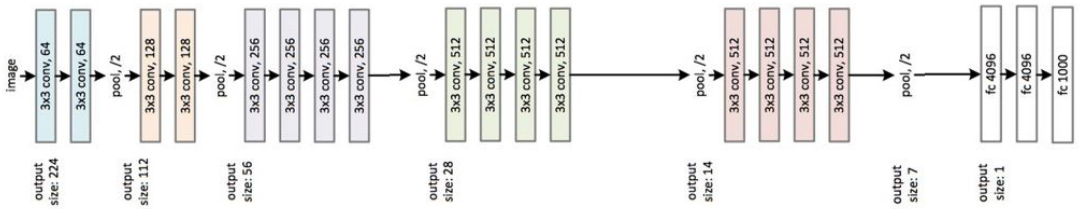
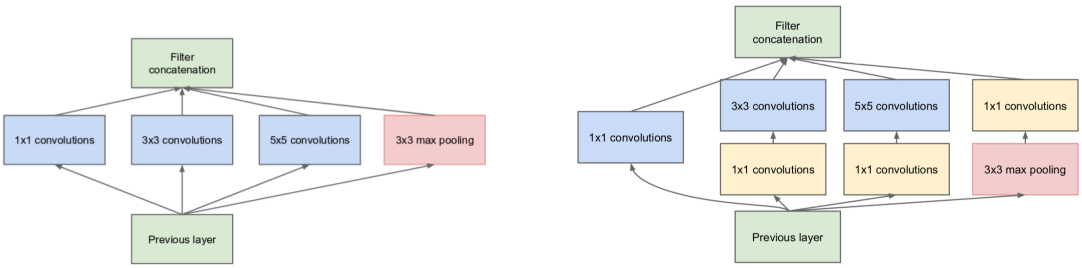


Figure 2.3: VGG-19 architecture, adapted from [36]



(a) Naive version

(b) Dimension reductions

Figure 2.4: Inception Module [38]

GoogLeNet/Inception v1

In image classification tasks, the size of what we are trying to classify can vary considerably. This makes it difficult to decide what kernel size to use; larger ones are preferred for features distributed over large portions of the image, whereas smaller kernels are good at detecting information distributed in a local region. To recognise variable-sized features, we therefore require different-size kernels. Typically, deep CNNs have many stacked layers, using kernels of varied dimensions. This leads to many computationally expensive convolution operations and the risk of overfitting. GoogLeNet [38] mitigates this by implementing multiple kernels of different sizes in the same layer, as shown in Figure 2.4. This makes the network wider, instead of deeper. The 1 x 1 yellow blocks are used for depth reduction i.e. reducing the number of input channels.

Although there are several extensions to the original GoogLeNet implementation [39, 40], they all share a fundamental backbone. At a given level, all extracted features are concatenated and fed into the next inception module. Inception v1 has nine stacked inception modules, containing 22 convolutional layers altogether, as shown in Figure 2.5. To combat the vanishing gradient problem, the authors introduce two auxiliary classifiers; the branches in the network. The total loss function is a weighted sum of the auxiliary loss from the two intermediate inception modules and the real loss. These auxiliary classifiers are only used for training purposes, not during testing or inference.

Instead of stacking fully connected layers after the final convolutional layer, Inception v1 uses a simple global average pooling. This drastically reduces the total number of parameters. The authors are able to remove the fully connected layers without affecting accuracy, due to the depth and width of the network. GoogLeNet won the aforementioned 2014 ILSVRC classification task; it achieves 6.7% top-5 error rate while also being faster than VGGNet.

DenseNet

DenseNet [41] is composed of several dense blocks, as shown in Figure 2.6. In these blocks, each convolutional layer receives additional inputs from all preceding layers. These are concatenated with the layer's own feature maps and forwarded to the convolutional layers ahead of it. This

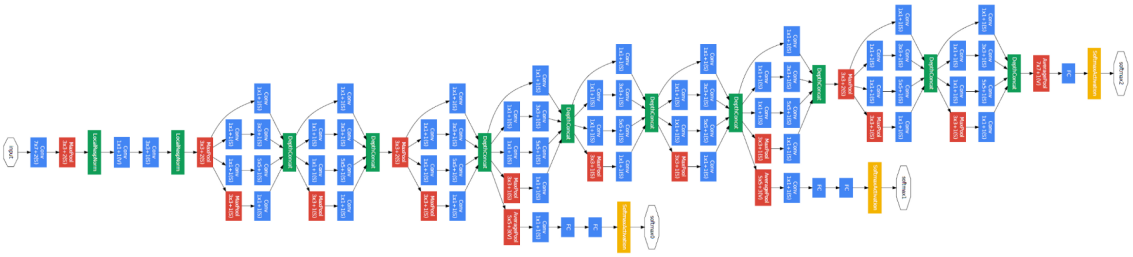


Figure 2.5: GoogLeNet/Inception [38]

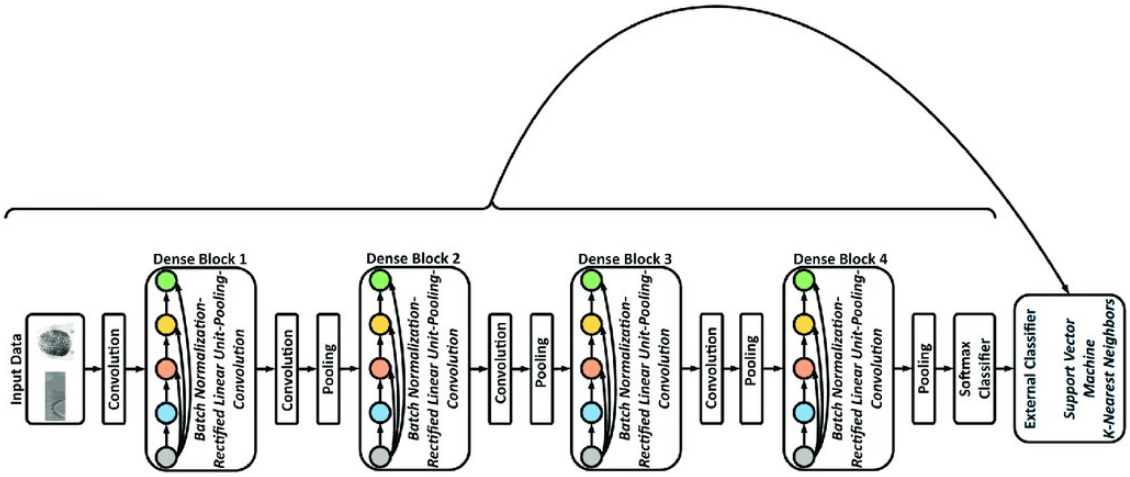


Figure 2.6: DenseNet, adapted from [42]

enables the network to be thinner and use fewer channels. Between dense blocks, for the same reason as with GoogLeNet, 1 x 1 convolutions are applied. The outputs from the final dense block are fed into a global average pooling layer and finally, a softmax classifier.

There are several advantages of DenseNet as an image classifier. The errors are propagated to earlier layers more directly, due to the architecture. DenseNet also requires fewer parameters than typical CNNs because it does not have to re-learn feature maps from previous layers. Since each layer receives the previous layers as input, the classifier takes high- and low-level features into account, unlike in standard CNNs where the classifier uses the most complex features. DenseNet-264, containing 260 convolutional layers, achieves 5.2% top-5 error rate on ILSVRC.

2.5 Object Detection

2.5.1 Overview

Terminology

Input - Image containing one or more objects.
Bounding Box - Coordinates of a rectangular border which fully encloses part of an image. They are defined by a point, width and height i.e. (x, y, w, h).
Output - One bounding box for each instance of a detected object and a corresponding object classification.
Ground truth - The true coordinates of the bounding box.
mAP - mean Average Precision. Described in Section A.2.4.

Goal

Given some input, we would like to detect all instances of objects from known classes, i.e. those that the algorithm has been trained on, and draw bounding boxes around them such that there is

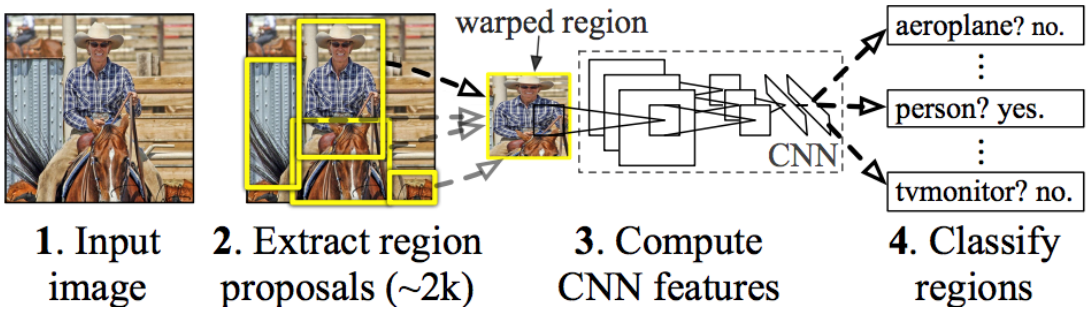


Figure 2.7: R-CNN architecture [43]

minimal difference between the output and ground truth.

2.5.2 Popular architecture

The following are popular CNN-based networks which have achieved state-of-the-art performance on object detection tasks. Faster R-CNN and You Only Look Once (YOLO) will form the building blocks of our federated solutions.

R-CNN

R-CNN [43] or Region-Based CNN, was the first application of CNNs to the problem of object detection. A regular CNN cannot solve this problem because the length of the output layer is variable; there could be several instances of an object in an image. A naive solution would be to take different patches from the image and use a CNN to classify if the object is in the region. The problem with this method is that each object of interest may have different aspect ratios and may be located in different areas of the image, making this computationally infeasible.

In R-CNN, only 2000 regions of interest are extracted from an image. These region proposals are generated using selective search; a greedy algorithm that generates many sub-segments and recursively combines similar regions into larger ones to produce the final proposals.

Each of the candidate region proposals is then reshaped into a square and fed as input into a CNN for feature extraction. These extracted features are then fed into an SVM to be classified, as shown in Figure 2.7. Apart from predicting the presence of an object in a region proposal, R-CNN separately performs a regression to optimise the bounding box values - bbox reg.

There are problems with R-CNN: long training times due to having to classify 2000 region proposals in each image, inability to perform real-time predictions and selective search being a fixed algorithm, resulting in generation of poor region proposals. R-CNN achieves an mAP score of 66.0 on the VOC 2007 [44] test.

Fast R-CNN

Fast R-CNN [45] addresses some issues of R-CNN. Fast R-CNN, as the name suggests, is a faster object detection algorithm.

In Fast R-CNN, the model takes the image and region proposals as input. These are fed into a pre-trained CNN, such as VGGNet, for feature extraction, as shown in Figure 2.8. The feature map generated by the CNN is fed into a custom layer - Region of Interest (RoI) pooling layer - which extracts specific features from a given region. The output is fed into fully connected layers which produces two outputs; one which is fed into a softmax to get a class prediction, and another representing the bbox. This process is repeated for every region of interest in the image.

Unlike R-CNN, which requires independent training for classification and localisation, Fast R-CNN includes the bbox reg as part of the neural network training. Overall training and test times are also reduced because the convolution operation only happens once per image. Test times do not

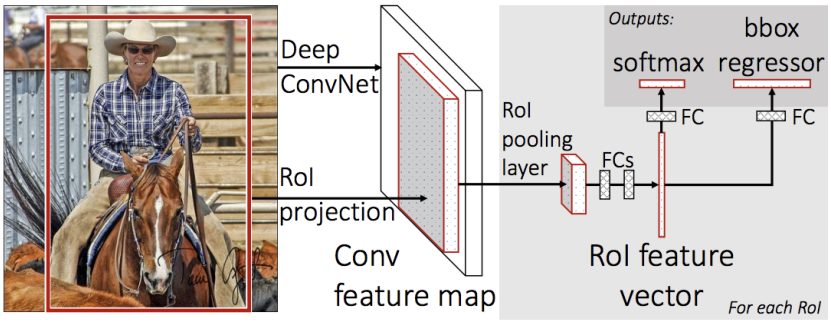


Figure 2.8: Fast R-CNN architecture [45]

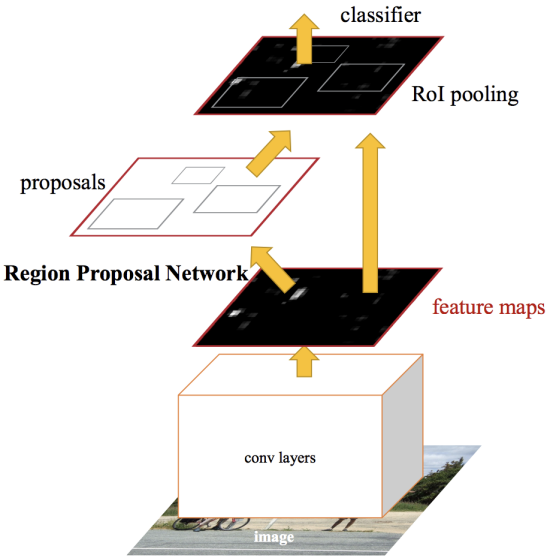


Figure 2.9: Faster R-CNN architecture [46]

improve as significantly because a set of candidate regions still need to be proposed with each input image; this bottlenecks the network's speed. The R-CNN accuracy is higher because of the end to end learning performed by the CNN. Fast R-CNN achieves an mAP score of 66.9 on the VOC 2007 test.

Faster R-CNN

Whereas R-CNN and Fast R-CNN use selective search to find region proposals, Faster R-CNN [46] eliminates the need for the algorithm because a small CNN generates the regions of interest instead. Faster R-CNN is comprised of two modules; the Region Proposal Network (RPN) and Fast R-CNN.

As shown in Figure 2.9, we see that an image is provided as input to a convolutional layer. Not shown in the figure, at each sliding-window location in the convolutional layer, nine different anchor boxes with different combinations of scale and aspect ratio are also generated. These help the RPN handle variations in aspect ratio and scale of objects, therefore improving region proposals. At each location, these nine anchor boxes are fed into the RPN, which outputs bounding boxes of various sizes and the corresponding probabilities for each class. The rest is similar to Fast R-CNN; the feature maps from the CNN and region proposals from the RPN are fed into an RoI pooling layer, before being fed into the classifier.

Test times are an order of magnitude faster using Faster R-CNN; it can even be used for real-time object detection. Faster R-CNN achieves an mAP score of 68.5 on the VOC 2007 test.

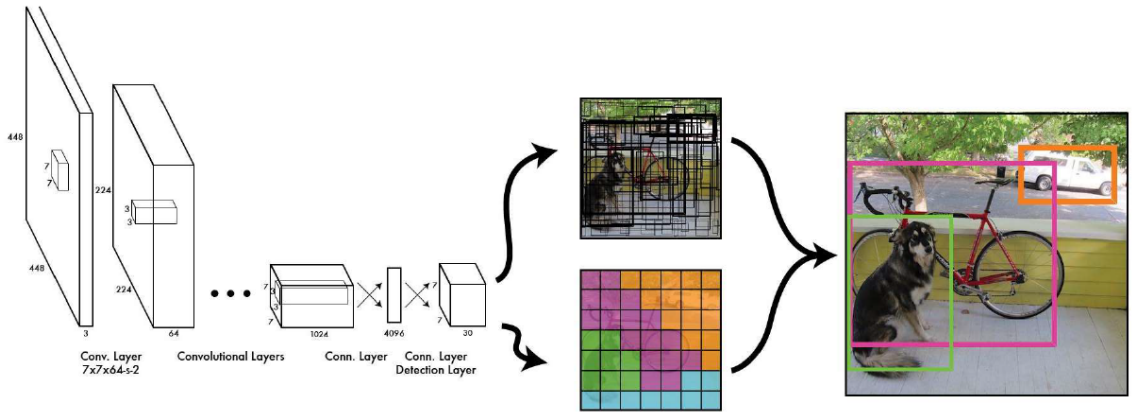


Figure 2.10: YOLO architecture, adapted from [47]

You Only Look Once

The previous networks treat detection as a classification problem by building a pipeline where object proposals are generated and sent for classification/regression. You Only Look Once (YOLO) [47] models detection as a regression problem instead. Figure 2.10 shows the model architecture: a single convolutional network predicts the bounding boxes (top) and their class probabilities bottom).

Each image is divided into an S X S grid. Within each square of the grid, YOLO predicts N bounding boxes and confidence i.e. for a given image, S x S x N bounding boxes and confidence are predicted. The confidence reflects the bounding box accuracy and, regardless of class, whether the bounding box actually contains an object. For each bounding box, YOLO also outputs a class probability for every class being trained. Many of these boxes have low confidence scores. Non-Maximum Suppression is applied i.e. only bounding boxes with confidence above a certain threshold are selected to locate the object within the image. This eliminates the majority of the boxes, as shown in Figure 2.10.

Due to the simplicity of the algorithm, inference is orders of magnitude faster than the previous methods we have discussed. This enables the algorithm to perform real-time detection. However, this comes at a cost; reduced accuracy. Looking at Figure 2.10, we see that each grid square (bottom) is assigned a single class. If there are multiple objects within a grid cell, not all of them can be detected. Hence, due to the spatial constraints of the algorithm, it performs poorly at identifying small objects in images. However, because YOLO sees the complete image at once, as opposed to looking at only region proposals, it is better at avoiding false positives than the previously discussed methods. When combined with Fast R-CNN, YOLO is able to achieve an mAP score of 72.4 on the VOC 2007 test.

2.6 Image Segmentation

2.6.1 Overview

Terminology

Ground truth - A mask of the image, representing what our model should predict in the segmentation.
Upsampling - Technique to increase size of an image.
Transposed convolution - Converting an image from low- to high-resolution. Done by taking transpose of filter to reverse the convolution.

Goal

Given an image as input, the output is an image, typically of the same dimensions as the input image, in which each pixel is classified to a particular class. In other words, the goal is to perform

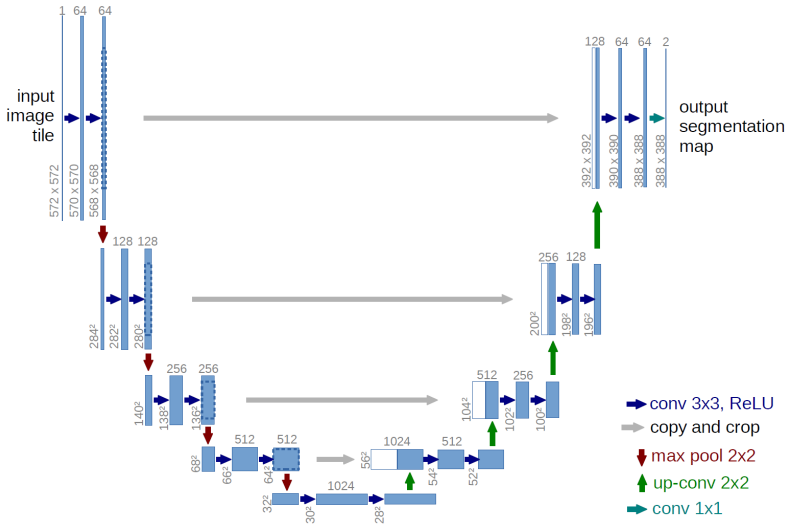


Figure 2.11: U-Net architecture [51]

pixel-level image classification.

2.6.2 U-Net

Since we will evaluate our federated solution in the context of semantic segmentation on a medical imaging dataset, we discuss U-Net - a CNN-based algorithm that has achieved state-of-the-art performance. Variations [48, 49] of the architecture exist, but they all share the same underlying structure. Our federated solution will use a version [50] of U-Net which has been modified to maximise brain tumour segmentation performance.

Architecture

The architecture of U-Net is shown in Figure 2.11. It consists of two sections and a layer which mediates between them (bottom-most):

Contraction: Also known as the encoder, this is where regular convolutions and max-pooling are applied. The size of the image gradually reduces, as the depth increases. The network has learnt what is in the image but lost information about its position.

Expansion: Also known as the decoder, this is where both transposed and regular convolutions are applied to generate the segmentation mask. The size of the image gradually increases, as the depth decreases. The positional information is recovered through upsampling.

Training

One reason that U-Net is appropriate in the context of medical imaging is that it achieves good performance after being trained on only a few images. This is unlike previously discussed Deep CNNs, which required large datasets. This is because of data augmentation techniques applied during the training stage. The loss is defined, for each pixel, such that there is a higher weight at each object's segmentation border. Pixel-wise softmax is applied on the generated image, combined with the cross-entropy loss function; each pixel is classified into one of the classes. This treats the segmentation task as a multi-class classification.

2.7 Conventional Federated Systems

The concept of federated computing systems are not only specific to applications in machine learning. In the 90s, a Federated Database System (FDBS) [52] - collection of independent databases co-operating for mutual benefit - was proposed. This does not perform any actual data integration

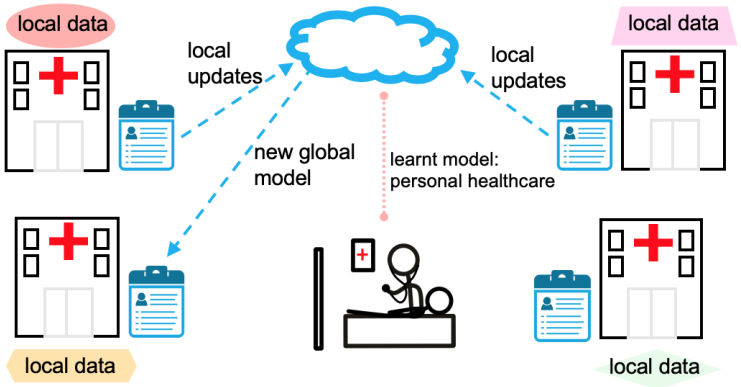


Figure 2.12: FL for healthcare via heterogeneous patient records from multiple institutions [54]

between the independent databases i.e. there is data federation. FDBS provides a uniform interface to allow clients to retrieve data from several, possibly heterogeneous i.e. having different semantics and structure, databases with a single query. A benefit of FDBS is the concept of autonomy; each database owner can still manage data without the FDBS. The concept of autonomy is present in FL as well. Participants should be able to associate or disassociate themselves from a network, and participate in more than one FL system. FDBS has some bottlenecks though. Performing schema matching to create a global view of the databases does not scale well with the number of attributes being pair-wise mapped. There were also issues with concurrency control, specifically ensuring global serialisability of transactions. However, this was been addressed with the introduction of commitment ordering.

In the past decade, due to the emergence of cloud computing, there have been studies conducted on federated clouds (FC) [53]. In an FC, multiple external and internal cloud computing services are deployed and managed. They give clients the option to select the best cloud services provider in terms of flexibility, cost and availability, to meet their requirements. As a result, applications run in the most appropriate infrastructure environments. FC also allows an enterprise to distribute workloads around the globe and move data between disparate networks. A drawback of FC is the difficulty in connecting a client with a given external cloud provider; each provider has their own network addressing scheme. The FC has to provide a uniform way for customers to access cloud services without the need for reconfiguration when using resources from different service providers.

2.8 Motivation for Federated Learning

2.8.1 Problem Formulation

FL problems typically involve using data from several remote participants to learn a single, global model, as shown in Figure 2.12. A requirement is that a participant's data must be stored and processed locally i.e. not transmitted to any other participants or a central server. Formally, the goal is to minimise this distributed optimisation problem [55]:

$$\min_{w} F(w), \text{ where } F(w) := \sum_{k=1}^{m} p_k F_k(w)$$

m is the total number of participants. p_k is the relative impact of each participant - this is typically either evenly shared i.e. $p_k = \frac{1}{m}$ or according to the proportion of data provided i.e. $p_k = \frac{n_k}{n}$ where n is the total number of samples and n_k is the number of local samples. We require $\forall k[p_k \geq 0]$ and $\sum_k p_k = 1$. F_k is the local objective function for the k-th participant. It typically represents the empirical risk over the data at that participant. The overall goal of the federated learning algorithm is to find a hypothesis for which the risk, i.e. uncertainty on performance, is minimal. We measure empirical risk on a known set of training data because we do not know the true distribution of the data that the algorithm will work on during inference.

We assume there are m different participants, each denoted by T_i where $i \in [1, m]$. In a non-federated environment, every participant T_i uses only its local data D_i to train a model M_i, having performance P_i. In a federated environment, all participants train a model M_f, having performance P_f. A valid federated system requires that $\exists i \ [P_f > P_i]$ i.e. even though some participants may not receive a better model via FL, there is at least one which achieves better utility [56].

2.8.2 Why Federated Learning?

Cellphones, IoT devices and autonomous vehicles are some of the modern distributed networks creating large volumes of data daily. As there are increasing concerns over transmitting private information over these networks, and the computational power of these devices is continuously improving, an attractive solution is to store data locally and perform training of statistical models directly at the source (edge computing [57]) or close to where it was created (fog computing [58]), instead of at a central server. Recent works have also considered training statistical models centrally but serving and storing them locally [59]. However, such proposals still involve sending data to a central server, and are not feasible when regulations preclude data sharing. FL solutions address these critical issues of data privacy and ownership in a way that can be scalable to a large number of participants, while achieving excellent model performance.

Relevance to Healthcare Data

Medical institutions are perfect candidates for FL: they have large volumes of sensitive patient data and the resources to perform on-site computation. The application of FL to medical data could have significant positive impact in the healthcare space. Models could potentially uncover patterns between previously disconnected datasets from several institutions. For institutions which contribute large amounts of data, predictive models could be developed that are fine-tuned [60] to their patients. Smaller institutions such as clinics could still receive a model that generalises well enough, based on the data seen from every contributing institution, to benefit their patients as well.

2.8.3 FL Systems in Production

Federated Learning Systems (FLS) have already been deployed by major service providers and are playing a crucial part in supporting privacy-sensitive applications. In this section, we discuss the their impact and how to incentivise participants to adopt FL.

Predictive Keyboards: One of the first use cases of federated learning was implemented [61] by Google for Gboard - their predictive keyboard. Due to high regulatory pressure and data overhead, it was no longer feasible to upload all users' text messages centrally to train their word guessing predictive algorithm. Despite the limited memory and computing power of smartphones, Google developed a practical and scalable privacy-preserving FLS.

Healthcare: Training a model to solve complex medical problems requires a lot of data from diverse institutions; they are extremely strict about maintaining control of their sensitive patient data. Assisted by highly traceable technologies, such as distributed ledgers [62], FL has enabled researchers to train predictive models on large amounts of sensitive data in a way that ensures it never leaves the medical institution [63, 64, 65, 60]. These algorithms are able to model heterogeneous data from both pharmaceutical companies and medical institutions.

Transport industry: Due to the potentially large number of self-driving vehicles we will see on the road and the need for them to quickly respond to real world situations, it is only a matter of time before FL solutions are used to reduce the amount of data transferred and accelerate the learning process. An additional benefit of FL over traditional cloud approaches is that they do not provide automakers with real-time access to when and where people are going, thereby preventing potential safety risks and privacy violations.

Incentive

In real-world applications, parties need to be persuaded to become participants in the FLS. Organisations and companies are typically incentivised to adopt FLS due to regulations. Users of a service are not required to provide their data to improve the models, and must therefore be incentivised to participate. Using the example of Gboard from above, Google cannot prevent users who do not provide data from using the keyboard. However, they can incentivise those who agree to participate by promising a higher accuracy of word prediction. Consequently, more users are motivated to participate, and the performance of the overall model improves.

However, it is still a challenge to design reasonable incentive mechanisms. These are pivotal to the success of the FLS. Apart from the Gboard incentive mechanism of providing better accuracy in exchange for participation, other incentive designs have been proposed to attract participants with high-quality data for FL by offering contract-based rewards [66].

2.9 Federated Learning Algorithms

2.9.1 Terminology

K - total number of participants
n_k - number of training samples at participant k
C - fraction of participants selected for each round
$\ell(w, b)$ - loss function for weights w and batch b
w_k^t - model's weight vector for participant k during federated round t

Hyperparameters

B - local minibatch size
E - number of local epochs
η - learning rate

2.9.2 Federated Stochastic Gradient Descent (FedSGD)

With SGD, gradients are computed using a random subset of the entire dataset, and used to perform a step of gradient descent. FedSGD is similar, but selects a random subset C of all participants and uses all their data to compute gradients. The central server averages the gradients proportionally to n_k and uses the result to perform gradient descent [67].

2.9.3 Federated Averaging (FedAVG)

Federated Averaging [67] is similar to FedSGD. However, local nodes perform more than one batch update on local data. They also exchange updated weights, instead of gradients, with the central server. This is detailed by the algorithm in Figure 2.13a. The authors show that models trained using FedAvg are able to achieve better accuracy and convergence in significantly fewer communication rounds, as shown in Figure 2.13b.

In each round of FL, the server selects a random subset S_t of participants. It then sends w_t to all selected participants. Each participant in S_t updates w_t for E epochs of SGD on client k with learning rate ℓ to obtain w_k^{t+1}. They send these updated weights back to the server, where they are averaged.

Equal Averaging

The updates from each participant are equally weighted during the aggregation at the server i.e. $w_{t+1} = \sum_{k=1}^{K} \frac{1}{K} w_{t+1}^k$

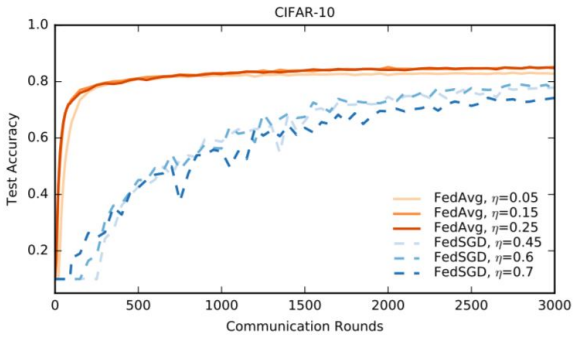
Weighted Averaging

A participant's updates are weighted by the proportion of samples they provide (weighted FedAvg), as shown in Figure 2.13a.

Server executes:
initialize w_0
for each round $t = 1, 2, \dots$ **do**
$m \leftarrow \max(C \cdot K, 1)$
$S_t \leftarrow$ (random set of m clients)
for each client $k \in S_t$ **in parallel do**
$w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate(k, w): *// Run on client k*
$\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B)
for each local epoch i from 1 to E **do**
for batch $b \in \mathcal{B}$ **do**
$w \leftarrow w - \eta \nabla \ell(w; b)$
return w to server

(a) FedAvg Algorithm



(b) FedAvg vs FedSGD performance on CIFAR-10

Figure 2.13: FedAvg algorithm and its performance benefits [67]

2.9.4 FedProx

FedProx [68] is a generelisation and re-parametrisation of FedAvg. The minor modifications made to the algorithm enable the solution to provide convergence guarantees in environments of non-IID data distributed across the network i.e. displaying statistical heterogeneity. This makes it a more robust method than FedAvg for FL. FedProx does not assume a uniform amount of work will be done by each participant, i.e. it allows for incorporating variable amounts of local work resulting from any system heterogeneity. The authors show that in highly heterogeneous settings; FedProx achieves more stable and accurate convergence, improving test accuracy by 22% on average.

2.10 Challenges of FLS

2.10.1 Expensive Communication

Communication overhead is a typical concern in any distributed system. FLS are typically composed of a large number of participants, sometimes in the magnitude of millions [61]. However, for the scope of this project, we assume that only 15 participants will be involved in a given round of FL. For large scale FLS to succeed, the systems have to employ communication-efficient methods during the training process. Communication overhead can be reduced by:

Reducing number of federated rounds: As discussed in Section 2.9.2, the use of FedAVG instead of SGD enables convergence in significantly fewer communication rounds.

Reducing the size of transmitted messages: Model compression techniques [69, 70] have been applied to reduce the size of communicated messages at each round. These have still provided convergence guarantees in the presence of non-IID data. These solutions do not take low participation into account; for this project we also assume full participation from all parties.

2.10.2 Systems Heterogeneity

In a typical FLS, the storage, communication and computational capabilities of each participant may vary. Participants may also be unreliable and drop out of the network [61], leading to possible biases in the global model. Ideally, an FLS should be robust to potential drop outs. It also should not fully depend on any single party. For the scope of this project, we assume that the medical institutions have similarly high-performing hardware, storage and network capabilities.

2.10.3 Statistical Heterogeneity

In the context of medical data, different hospitals will have different distributions of patient records. This could be due to geography; people in drier countries may suffer fewer cases of pneumonia than

those that experience large amounts of rain. As previously discussed, medical institutions have a lot to gain from FLS. If one institution has representative data for one task, and another institution has representative data for a different task, if the institutions agree to participate in FL, they both can potentially receive models capable of improving performance on the two tasks. If different sized medical institutions, such as hospitals and clinics, are participating in the FLS, it is likely that the number of data points across each participant will vary. As mentioned in Section 2.9.2, FedAvg diverges when data is not identically distributed across network participants.

Relevance to Medical Imaging Data Collection

The global diagnostic imaging market is dominated by three companies [71]; Siemens (23.2%), General Electric (22.2%) and Philips (19.7%). The imaging devices will produce different scans depending on the manufacturer [72]. For example, Magnetic Resonance Imaging (MRI) signals are not recorded in absolute values, resulting in different intensities for a given contrast, depending on the platform used for acquisition. This introduces further statistical heterogeneity into the FLS. However, for our project we will be using maximally homogenised medical imaging datasets i.e. those that have been normalised and interpolated to standard resolutions to enable comparison between different methods.

2.10.4 Privacy Concerns

By not transporting raw data, FLS already provide some form of data protection. However, model updates can still leak sensitive information to a third party or the central server. In Section 2.11, we discuss potential attack vectors and privacy-preserving solutions.

2.11 Privacy Mechanisms

For privacy-preserving algorithms to be feasibly incorporated into FLS, they must be efficient with respect to computation and communication overhead. Of significant importance, they should also not overly compromise an algorithm's inference capabilities. These privacy-preserving methods can typically be grouped into the following categories:

Global Privacy: Requires that the model updates generated at each round are private to all untrusted third parties, other than the central server.

Local Privacy: Requires that the updates generated at each round are also private to the central server.

In the rest of this section, we discuss potential attack vectors to FLS, and popular methods of mitigating them. The incorporation of formal privacy-preserving guarantees typically comes at the cost of model accuracy. It is important to find an appropriate trade-off based on nature of the data and requirements of the application.

2.11.1 FLS Attack Vectors

The incorporation of formal guarantees is not the main goal of this project. In this section we explore the types of attacks that an FLS is generally susceptible to. However, for simplicity, we assume that the aggregator is the only component in the FLS that can compromise the privacy of the other FLS participants, i.e. other parties are honest participants.

Membership inference and model inversion

These can happen when the aggregator is honest-but-curious - follows the protocol correctly but also attempts to infer as much as possible about the data of the parties communicating with it. In a membership inference attack, the attacker can determine if a given individual was present in the training data of an ML model [73]. Unlike model inversion, no additional personal data is learnt about the individual. However, in medical contexts, this information could still potentially be sensitive; it may be possible to determine if an individual is a patient at a particular hospital.

In a model inversion attack, the output of a model is applied to some hidden input, such as an adversarial model, to infer some if its features. In the case of facial recognition systems, a model inversion attack could construct an artificial image of a person, using an average of the images of that person which were provided during the training phase. The generated image does not produce a specific image from the training dataset, unlike with membership inference. Previous work [73] has shown that it is typically difficult to perform such attacks on CNNs. However, Generative Adversarial Networks (GANs) have been used recently [74] to effectively extract such information in collaborative settings. Despite this, it has been shown [75] that the attack has a severe limitation; it is more of a threat when there are only two parties involved. The attack performance significantly drops when the number of participants increases.

Poisoning attacks

This is an active adversarial attack where a malicious participant intentionally "poisons" the training phase in attempt to cause targeted misclassification of the global model [76]. The attacker - a participant in a federated round - controls the local data, training procedure, hyperparameters and the weights submitted to the server for aggregation [77]. In general, this type of attack is not very effective on large-scale FLS. Due to the randomness in choice of participants, the malicious participant is rarely selected for a federated round if there is a large number of them. Updates from all participants are averaged, which could lead to the malicious update having negligible effect on the global model. If the aggregator detects a significant degradation in global model performance after a federated round, it can kick the malicious participant out of the network. In this project, we assume that there are no such adversarial medical institutions.

Byzantine faults

In distributed systems, components may arbitrarily fail or deviate from the protocol without detection; this can lead to misleading results. Existing research [78] shows that it is possible to devise byzantine-resilient algorithms that guarantee convergence for distributed SGD. However, if the FLS uses a central trusted aggregator, this then becomes a single point of failure. To keep this project simple, we make the weak assumption that there will be no byzantine faults.

2.11.2 ε-Differential Privacy

This is the most widely used approach in privacy-preserving collaborative machine learning. This is due to the strong theoretical guarantees it provides, the simplicity of the algorithm and the small systems overhead it incurs. A randomised algorithm M is ε-differentially private [79] if for all $S \subseteq \text{Range}(M)$, when applied to two datasets $D1$ and $D2$ which differ by a single element:

$$\Pr[A(D_1) \in S] \leq e^{\varepsilon} \text{ x } \Pr[A(D_2) \in S]$$

It works by adding sufficiently large, typically Laplace, noise to the result of a computation to hide an individual contribution. This prevents inference about whether a particular sample is used in the training phase. The amount of noise added is dependent on the ε value. Gradients are usually clipped before applying the noise mechanism. Intuitively, adding more noise improves privacy, but can degrade accuracy significantly. This has also proven to be the case when incorporated in federated learning environments, especially when there are a large number of parties, each with limited amounts of data [80, 64].

2.11.3 Secure Multi-Party Computation (SMPC)

With SMPC, two or more participants collaboratively compute an agreed-upon function over a network, on some private input provided by each participant [81]. The output is decrypted by one or more participants. SMPC is a lossless method, and retains original accuracy with high privacy guarantees. However there are some disadvantages; it incurs a large communication overhead. Additionally, even though the inputs from one party cannot be derived by another, information can still be inferred from the decrypted output. Hence, SMPC is vulnerable to inference. Alternative approaches [80] have been proposed to utilise both DP and SMPC to balance their trade-offs and provide stronger privacy guarantees. This has shown to be a scalable approach, as it enables the reduction of injected noise when the number of participants increases, without sacrificing privacy or accuracy.

2.11.4 Fully Homomorphic Encryption

This is a method of securing the learning process by computing on encrypted data [81]; inputs, outputs and intermediate values in FHE are always encrypted. The main benefits of FHE is that it supports arbitrary computations and enables privacy-preserving computation in the presence of an untrusted server. It also involves a much lower communication overhead than SMPC. However, FHE incurs a significantly higher computational overhead. Frameworks such as Zama [82] are being developed to enable fast and accurate inference over encrypted data, while minimising the performance overhead. A drawback of FHE is that it has so far only shown to be applicable in FL when learning linear models [83, 84].

2.11.5 Trusted Execution Environment (TEE)

Distributed collaborative learning designs are vulnerable to data poisoning and backdoor attacks. In FLS, there is typically a limited sense of accountability; it is difficult to determine which participants are contributors of "bad" training data. A TEE, such as IntelSGX, ensures that computation can take place in a way that guarantees integrity and confidentiality of data. This is done through the provision of a hardware-isolated processing environment [85]. TEEs enable remote attestation; a method by which a participant authenticates its hardware and software to remote participants, i.e. a central server. In decentralised systems, this feature can be utilised to identify corrupt or malicious participants. The drawback of using a TEE is the significant execution overhead, due to the numerous context switches between the TEE and the regular processing environment [86].

CALTRAIN [87] is a TEE-based multi-party collaborative learning system that is able to achieve data confidentiality and model accountability, while overcoming the capacity and performance constraints of secure enclaves. The authors have achieved the same prediction accuracy compared to deep learning models trained in non-protected environments. CALTRAIN supports accountability by maintaining secure links between training instances and the participants that contribute them; the system is able to identify malicious/compromised participants that impact the inference capabilities of the global model.

2.12 Collaborative learning applied to medical imaging

2.12.1 FL vs IIL vs CIIL

In [63], the authors compare the performance of U-Net on the BraTS dataset, using three different collaborative learning approaches:

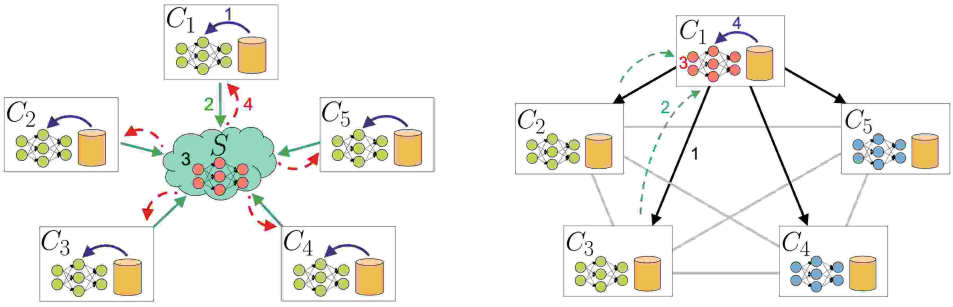
1. A typical FLS, as shown in Figure 2.14a.

2. Institutional Incremental Learning (IIL) - Each institution sequentially trains a model, only once, in any way it chooses.

3. Cyclic Institutional Incremental Learning (CIIL) - Effectively repeats IIL, with each institution training the model for a fixed number of epochs, before passing it to the next institution.

IIL and CIIL are completely decentralised approaches; i.e. there is no central aggregator. IIL uses significantly less communication bandwidth than FL because each institution only sends the model it has trained, and receives the initial and final models. However, the IIL technique suffers from "catastrophic forgetting", i.e. the model forgets previously learnt patterns when new data from an institution replaces data from the previous one. As a result, it does not perform as well as either of the other approaches.

CIIL mitigates the issue of catastrophic forgetting by fixing the number of epochs at each institution. However, FL is still a more feasible approach as it effectively does the same thing, but in parallel. This means that FL is better for scaling to a larger number of institutions. Though they both perform similarly on the segmentation tasks, FL is much more stable; CIIL results showed significant variance.



(a) FL with server

(b) BrainTorrent: P2P serverless FL

Figure 2.14: Overview of FL process [65]

The authors achieve 99% of the centralised model's performance using an FLS. However, a weakness of their study is that they do not evaluate how the model's performance degrades when formal privacy guarantees are incorporated.

2.12.2 FL with ε-DP

Last year, a study [64] was conducted by researchers from King's College London, in partnership with NVIDIA, to investigate the feasibility of applying DP techniques to protect patient data in an FLS. Their solution is evaluated on the BraTS dataset, and the general architecture of the system is shown in Figure 2.14a.

At the client-side, a momentum-based SGD is used as the optimiser. At the end of each federated round, the participants' momentum values are reset. When performing the reset operation, the authors observe slightly faster convergence, but similar model performance, compared to when it is not reset. For privacy, a selective parameter update is adopted; to limit the amount of information shared by a participant, only a fraction of the weights - those that are greater than a threshold - are passed to the next stage, where the values are clipped to lie within a fixed range. Before being sent to the server for aggregation, the DP module adds noise to the clipped values via a Laplace mechanism. At the server-side, weighted FedAvg is performed.

The evaluation shows that sharing partial updates causes a noticeable decrease in model performance. Surprisingly, gradient clipping does not affect the convergence speed of the model, and the model performance decrease is almost negligible. The authors show that by sharing a partial model, with reasonable DP guarantees applied to the updates, the performance degradation is reasonable; $DC \approx 0.80$, compared to the FL model with no formal-privacy guarantees having $DC \approx 0.85$. This is a justifiable trade-off for the increased privacy of the healthcare data.

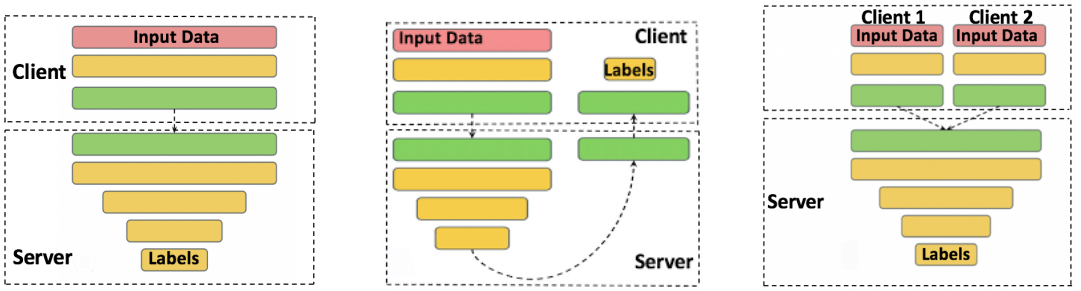
2.12.3 Serverless FL

We have previously only considered FLS with a trusted central server that performs aggregation. With BrainTorrent [60], the authors propose a completely decentralised approach where the medical institutions communicate directly among themselves.

Since there is no central server, a new strategy is proposed to coordinate training. All participants are connected directly in a peer-to-peer network, as shown in Figure 2.14b. All N participants maintain a vector $v \in \mathbb{N}^N$ containing their own version and the last versions of models used to create their current models. At the beginning, $v = \overrightarrow{0}$ for all participants. Every time it initiates a training process, it increments its own version number in v. Training at a single participant works as follows:

1. Each participant is locally trained in parallel for a few iterations.

2. A random participant C_i initiates the training process. It asks for the latest models from the other participants to generate v_{new}.



(a) Vanilla split learning

(b) Split learning without label sharing

(c) Split learning for vertically partitioned data

Figure 2.15: SplitNN configurations [65]

3. All clients C_j with updates (C2 and C3 in Figure) send their weights and number of training samples to C_i.

4. All models sent from the clients C_j are merged with C_i's current model to a single one, using weighted averaging.

While this is an interesting concept, the evaluation itself is particularly weak. The authors claim, using statistically insignificant results, that BrainTorrent outperforms traditional server-based FL approaches. They also never evaluate the performance overhead or how formal privacy-guarantees can be incorporated into this fully decentralised solution.

2.12.4 FL vs Split Learning

The proposed configurations [65] of SplitNN [88], shown in Figure 2.15, facilitate:

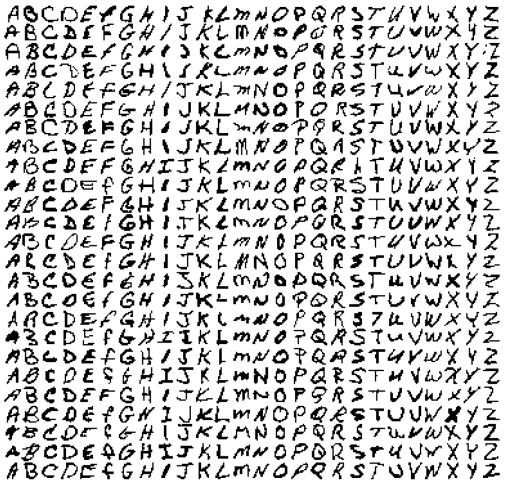
- The collaborating institutions needing to perform multiple tasks.
- Performing learning without shared labels.
- Collaborative learning when institutions hold different modalities of patient data.

In the simple vanilla configuration for split learning (Figure 2.15a), each medical institution trains a partial deep network up to a specific layer - the cut layer. The output from this layer are transmitted to a server to complete the training procedure. The gradients are propagated from the last layer of the server to the cut layer. Only the gradients at the cut layer are forwarded back to the medical institutions. This enables a split learning network to be trained without sharing raw data.

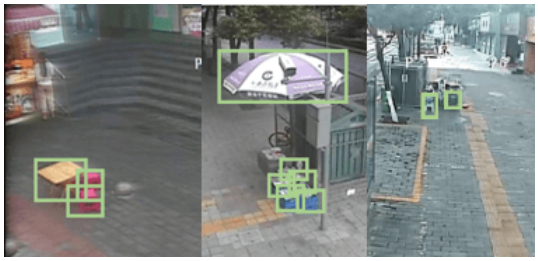
In the variant without label sharing (Figure 2.15b), a U-shaped configuration is devised to prevent institutions needing to transmit labels to the server. The outputs from the final layer of the server are sent back to the institution. This enables the clients to generate the gradients without sharing the corresponding labels. This is ideal in scenarios where labels include highly sensitive information.

The final configuration (Figure 2.15c) has been proposed to enable split learning on vertically partitioned data. Two institutions with overlapping patients, containing different modalities of patient data, train partial models upto the cut layers. The output at the cut layer from both institutions is then combined and sent to the central server to complete the training process. A real-world example of this is radiology centres collaborating with pathology test centres and a server to perform diagnosis.

The authors show that their SplitNN modifications use significantly lower computational resources and communication bandwidth than FL approaches, while achieving higher accuracy. However, the authors do not address the incorporation of formal-privacy guarantees to the configurations. Additionally, the authors claim that this is "split learning for health" but do not evaluate any of their proposed configurations on medical datasets.



(a) FEMNIST samples, adapted from [89]



(b) FedVision - Street Dataset samples [15]

Figure 2.16: Non-medical imaging dataset samples

FEMNIST			
Number of devices	**Total Samples**	**Samples per device**	
		mean	stdev
3,550	805,263	226.83	88.94

Table 2.1: Statistics of FEMNIST dataset, adapted from [14]

2.13 Proposed Datasets

2.13.1 FEMNIST

We use FEMNIST to evaluate the generalisation of our proposed FL solution on image classification tasks. It is a database of over 800, 000 handwritten digits, lowercase and uppercase letters. Samples are shown in Figure 2.16a. Unlike the original EMNIST dataset, FEMNIST is curated for evaluation in federated settings; the dataset is partitioned based on the writer of the digit/character. The statistical breakdown of the dataset is presented in Table 2.1.

2.13.2 FedVision

The FedVision - Street Database was created to boost the academic research and industrial applications of computer vision based on federated learning. This initiative is in partnership with the WeBank AI Group - another large service provider exploring the potential of FL [90] in various applications.

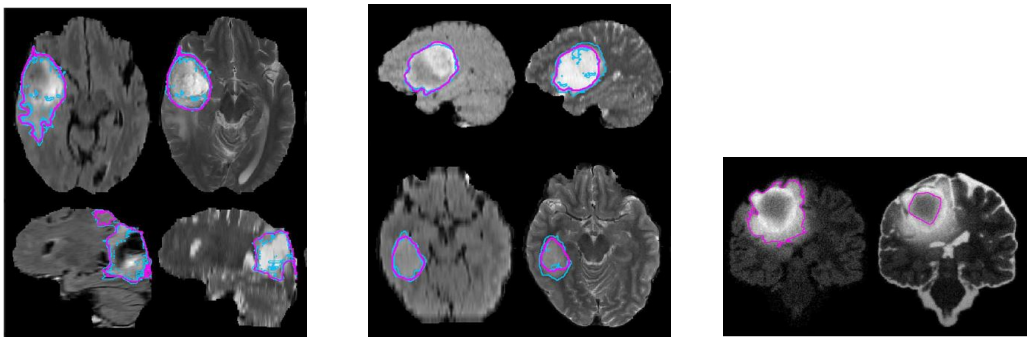
The Street Database is a compilation of over 900 images, captured by 26 different street cameras, representing seven categories and annotated with detailed bounding boxes. Samples are shown in Figure 2.16b. Each image has dimension 704 x 576 pixels, is taken during the day and has at least one labelled object; an image may contain multiple instances of an object with the same class. The data distribution is non-IID and unbalanced; characteristics that represent real-world FL settings.

The entire dataset of 956 images is partitioned into 80% training and 20% testing, where the test images come from six cameras with very few images. The remaining 20 cameras' pictures are used to represent federated participants. With the Street-20 dataset, each camera represents a participant and with the Street-5 dataset, cameras geographically co-located are clustered to represent a single participant. A detailed distribution of the object labels in both variants of the dataset are presented in Table 2.2.

Street-5	Images/Client			Frequency/Client		
Dataset	total	mean	stdev	total	mean	stdev
Basket	127	25.40	15.08	165	33.00	22.20
Carton	133	26.60	27.94	215	43.00	55.13
Chair	369	73.80	59.36	494	98.80	87.60
Electro-mobile	257	51.40	45.23	510	102.00	105.27
Gastank	71	14.20	26.42	106	21.20	36.19
Sunshade	255	51.00	31.89	413	82.60	56.02
Table	73	14.60	29.20	102	20.40	40.80

Street-20	Images/Client			Frequency/Client		
Dataset	total	mean	stdev	total	mean	stdev
Basket	127	6.35	11.06	165	8.25	14.93
Carton	133	6.65	12.03	215	10.75	26.23
Chair	369	18.45	17.70	494	24.70	28.54
Electro-mobile	257	12.85	14.81	510	25.50	37.55
Gastank	71	3.55	11.14	106	5.30	16.63
Sunshade	255	12.75	19.14	413	20.65	37.87
Table	73	3.65	13.06	102	5.10	19.26

Table 2.2: Distribution of labels in the training set of Street-5 and Street-20 datasets [15]



(a) High-grade tumours (b) Low-grade tumours (c) Synthetically generated tumours

Figure 2.17: BraTS whole tumour volume dataset samples [9]

2.13.3 BraTS

The BraTS dataset contains multi-institutional multi-modal MRI scans of 285 subjects with brain tumours. Each subject's scans are associated with voxel-level annotations of "whole tumour", "tumour core" and "enhancing tumour". We focus our evaluation on the "whole tumour" samples of the dataset, shown in Figure 2.17. The blue outlines represent the individual experts' annotations and the magenta lines represent the consensus segmentation. There are no individual annotations for 2.17c, as they represent the synthetic cases generated by software.

2.13.4 CheXpert

The dataset contains 224, 316 chest radiographs from 65, 240 patients, labelled for the presence of 14 typical chest radiographic observations. The task is to predict the probability of these different observations from multi-view chest scans, as shown in Figure 2.18.

For each scan, each of the 14 labels are classified as positive, negative or uncertain. The statistics are shown in Table 2.3.

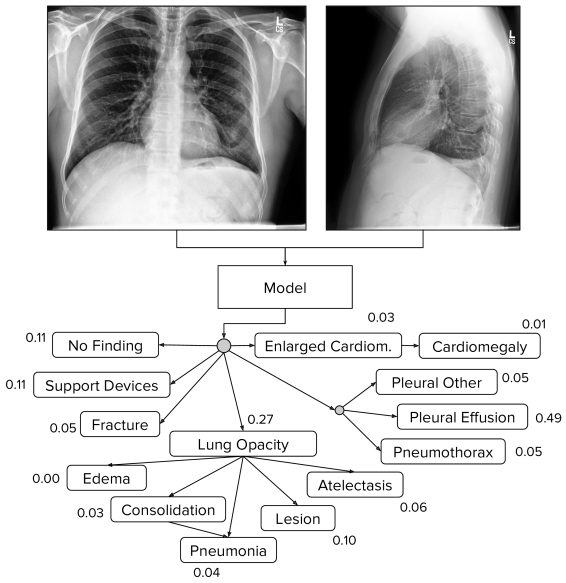


Figure 2.18: CheXpert sample with probability of different observations [8]

Pathology	Positive	Uncertain	Negative
No Finding	16627	0	171014
Enlarged Cardiomegaly	9020	10148	168473
Cardiomegaly	23002	6597	158042
Lung Lesion	6856	1071	179714
Lung Opacity	92669	4341	90631
Edema	48905	11571	127165
Consolidation	12730	23976	150935
Pneumonia	4576	15658	167407
Atelectasis	29333	29377	128931
Pneumothorax	17313	2663	167665
Pleural Effusion	76696	9419	102526
Pleural Other	2441	1771	183429
Fracture	7270	484	179887
Support Devices	105831	898	80912

Table 2.3: Distribution of observations in dataset, adapted from [8]

Baseline models

Since the training labels for each study in the dataset have either 0, 1 or u to represent negative, positive and uncertain observations respectively, the authors explore different approaches to using the uncertainty labels when training their baseline.

U-Ignore: Ignore u labels when training
U-SelfTrained: Use model trained by U-Ignore to make predictions on unlabelled observations. Each uncertainty label is replaced with the model's prediction.
U-Zeros: Treat all u instances as 0 (negative)
U-Ones: Treat all u instances as 1 (positive)
U-MultiClass: Treat u as separate class.

Validation Set

The validation set contains 200 studies from 200 patients that are randomly sampled from the full dataset. The ground truth is obtained by taking consensus on the individual annotations from three board-certified radiologists.

Test Set

The evaluation set consists of 500 studies from 500 previously unseen patients [91]. The ground truth labels are obtained by performing majority voting on the annotations from five out of eight board-certified radiologists. The remaining three radiologists annotations are used to benchmark radiologist performance.

Chapter 3

Project Plan

Apart from the report and presentation, in order to complete the project, we require:

- Background knowledge on existing FL techniques and viable alternatives. Additionally, sufficient domain knowledge to perform the medical imaging classification and segmentation tasks. (Requirement addressed in Chapter 2)
- Methods for evaluating the performance of the various algorithms on the different imaging tasks of classification and segmentation. (Requirement addressed in Chapter 4)
- A method to benchmark the computational and communication overheads of the systems we will be evaluating. (LEAF benchmarking tool [14] identified to satisfy this requirement)
- Access to a public cloud or the department's internal compute server, to handle the intense computational and storage requirements needed during the training phase.
- A solution that can scale to involve at least 10-15 different participants during training.

3.1 Expected project timetable

The FEMNIST and FedVision datasets were not only selected because they would demonstrate the generality of our proposed FL solutions; they are also significantly smaller than their medical imaging counterparts. The down-sampled version of CheXpert is nearly 11GB, whereas the FEMNIST dataset is only slightly larger than 0.5GB. Similarly, the BraTS dataset is larger than 3.5GB, whereas the FedVision dataset is slightly over 0.1GB. Developing our initial FL solutions on these smaller and simpler imaging tasks will enable us to rapidly prototype different configurations before moving onto the main tasks.

At the time of writing, we are currently working through the pysyft tutorial series [92]. These will enable us to have a better understanding of how to implement elegant FL solutions. At the end of January, our target is to have an end-to-end implementation and evaluation of FL on the FEMNIST dataset. The next target is to have an implementation and evaluation of FL on the FedVision dataset. These results will serve as reference points for how we can expect the FL solutions to perform on the CheXpert and BraTS datasets.

In February, feedback from the meeting with Dr. Kainz about the interim report will be used to create an outline for the final report. A goal in February would also be to have configured LEAF to benchmark the computation and communication overhead in different FL settings. We hope this will not be too complicated, due to LEAF's modular design.

Exams will be taking place in the middle of March. At the beginning of March, ideally we would have configured training on a public cloud, such as Azure, or the department's internal compute server. This would enable us to start the potentially long training procedures on the CheXpert and BraTS datasets, allowing revision while still progressing on the project-front. After exams, the preliminary results should be available. If the results are not close to what is expected, we may have to consider alternative solutions. Otherwise, we shall proceed by setting up hyper-parameter

tuning, to find the "optimal" configurations for our FL solutions. Since we are evaluating performance on four datasets, we can dispatch the training jobs in parallel, to reduce lead times. While the models are training, more work on the report will be carried on. In particular, a template for how to report the evaluation results can be created.

April will be used as a "recovery buffer" for tasks that could not be completed during the term e.g. due to extra time taken to experiment with and select off-the-shelf state-of-the-art models to use in our federated solutions. If there are not many of these, our stretch goals, discussed in Section 3.2 will be undertaken during this period. As this will involve more long training periods, the remaining sections of the report can be completed.

During the project health check up in the middle of May, a road-map will be created, detailing what is left to complete the project successfully. The first draft of the report should hopefully be completed during this time. A feedback session will be organised with my supervisor and/or the second marker to get suggestions on how to improve the final version.

We aim to submit the final version of the report in the first week of June. From then, we will work on the project presentation; two weeks should be more than sufficient time to prepare a high-quality presentation.

3.2 Extensions

3.2.1 Likely

- An investigation into how classification/segmentation performance degrades, and how communication overhead increases, as the number of participants increases ($n = 1, 5, 15, 30, 50$).
- A reasonable stretch goal of the project is to incorporate and quantify the formal privacy guarantees. We shall attempt to incorporate:

  ➔ ε-DP

  ➔ SMPC

  into our proposed solutions, in that order.

3.2.2 Unlikely

- Optimising the computational and communication overhead involved in our proposed FL solutions. We determined that this would make the scope of the project too broad to complete within the available time frame.
- Showing how robust our FL solution is by developing an adversarial model to attack it; this would also be too challenging given the time constraints.

Chapter 4

Evaluation Plan

The solution will be quantitatively evaluated on its ability to achieve near state-of-the-art performance on medical imaging tasks related to classification and segmentation. With zero formal privacy guarantees, we would constitute our federated solutions as successful if they can perform within 5% to 15% of their centralised implementations. Our stretch goals will be to study and quantify the trade-off in performance of the solutions when i) formal privacy guarantees are incorporated and ii) the number of participating institutions increases.

4.1 Experimental Methodology

The seeds of every random number generator will be set to known values to ensure the reproducibility of results. Each dataset will be partitioned into training, validation and test sets. The models will be trained on the training data, hyper-parameters tuned on the validation data, and performance evaluated on the test data. To investigate the effects of data distribution between participating institutions, each federated implementation will be evaluated on two separate cases; data being both evenly and unevenly distributed among the institutions. The evaluations will either take place on a public cloud, such as Microsoft Azure, or the department's internal compute server. If time permits, we could also evaluate our solutions using the official test benchmarks [91, 93].

4.2 Imaging Metrics

Details of all imaging metrics can be found in Section A.2.

4.2.1 Classification

FEMNIST

The success of our solution will be determined by its **accuracy**. We acknowledge that this is not the most appropriate metric due to the class imbalance in the EMNIST dataset [13]. Thus, we shall also report the **F1 scores** for each class. To represent the classifier's overall F1 score, we shall also report its **weighted-average F1 score**.

CheXpert

Although the dataset consists of 14 different observations, the competition task only evaluates a model's performance on five of them; Ateclectasis, Cardiomegaly, Consolidation, Edema and Pleural Effusion. We will evaluate the success of our solution using its **ROC AUC** score on these five observations [8].

4.2.2 Detection

FedVision

The success of our solution will be determined by two metrics; **Intersection over Union (IOU)** and **mean Average Precision (mAP)**.

4.2.3 Segmentation

BraTS

We shall use the **Dice Coefficient (DC)** as the metric to evaluate the success of our segmentation algorithm.

4.3 Privacy Metrics

To investigate how formal privacy guarantees affect the performance of the algorithms, we will evaluate the performance of our federated solutions with SMPC and varying amounts of ε-DP incorporated.

4.4 Systems Analysis Metrics

We shall use the LEAF benchmarking framework [14] to measure the amount of resources required for each implementation during the training phase. The 2 metrics we shall report are:

- **Computation:** Floating Point Operations Per Second (FLOPS)
- **Communication:** Number of bytes downloaded/uploaded

It must be noted that the purpose of this project is not to optimise the computational and communication overhead of our proposed solution. Instead, we include these results to serve as a baseline for future implementations to be evaluated against.

4.5 Benchmark

We shall benchmark each federated implementation, using the above metrics, against a completely centralised solution i.e. a single model trained on the entire dataset. The focus of this project is not to improve the performance of a state-of-the-art classification/detection/segmentation algorithm. Instead, the goal is to take an existing state-of-the-art algorithm and design a federated, privacy-preserving solution that is able to achieve similar performance on the imaging metrics.

4.6 Proposed Implementations

To reach the performance goals specified by the benchmark, we shall develop the following FL implementations:

- Weighted Federated Averaging solution: A more heuristic approach to the FedAvg algorithm which weights updates based on the proportion of data that an institution provides.
- Weighted Federated Averaging solution utilising a momentum based optimiser: A refinement to the above approach, inspired by the implementation [64] detailed in Section 2.12.2.
- Completely Decentralised Peer-to-Peer solution, inspired by the implementation [60] detailed in Section 2.12.3.

4.7 Baseline

For each experiment, the FedAvg algorithm will be used to establish a baseline. The performance of this baseline will serve as the lower bound of the expected performance of our proposed FL implementations.

Appendix A

A.1 Activation Functions

A.1.1 Sigmoid

Maps input to the range (0, 1).

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

A.1.2 Tanh

Maps input to the range (-1, 1).

$$f(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

A.1.3 Softmax

Given a vector, x, of real numbers, normalises it into a probability distribution proportional to the exponential of the input. i.e. each x_i gets mapped to (0, 1) and the sum of the output x values is 1.

$$f_i(x) = \frac{e^{x_i}}{\sum j = 1^{J} e^{x_j}} \qquad \text{for } i = 1, ..., J$$

A.2 Evaluation Metrics

A.2.1 Accuracy

The accuracy is the number of correct classifications divided by the total number in the dataset.

Terminology

TP - True Positives
TN - True Negatives
FP - False Positives
FN - False Negatives

Formula

$$ACC = \frac{TP + TN}{TP + TN + FN + FP}$$

A.2.2 F1 score

The F1 score conveys the balance between the precision and the recall of classifications.

Terminology

Precision = $\frac{TP}{TP+FP}$

Recall = $\frac{TP}{TP+FN}$

Formula

$$F1 = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

A.2.3 Weighted-Average F1 score

Gives a classifier's overall F1 score. The F1 score for all classes are averaged and weighted by the number of samples from each class.

Terminology

c - Number of classes
n - Total samples

Formula

$$\text{Weighted F1} = \frac{1}{n} \sum_{i=1}^{c} n_i * F1_i$$

A.2.4 mean Average Precision (mAP)

Common metric to compare object detection networks. It is a score representing the average for average precision of all classes.

Terminology

c - Number of classes
AP_i - Average of maximum precision at different recall values for class i

Formula

$$mAP = \frac{1}{c} \sum_{i=1}^{c} \text{AP}_i$$

A.2.5 Intersection Over Union (IOU)

Popular metric used to measure the accuracy of an object detector on a particular dataset. Ranges from 0 to 1, with 1 signifying the greatest similarity between predicted and truth.

Terminology

A - Ground Truth bounding box
B - Predicted bounding box

Formula

$$\text{IoU} = \frac{A \cap B}{A \cup B}$$

A.2.6 Dice Coefficient (DC)

This is a particularly popular metric for evaluating medical imaging segmentation algorithms. It is very similar to the IoU - they are positively correlated. Also ranges from 0 to 1, with 1 signifying the greatest similarity between predicted and truth.

Terminology

A - Ground Truth
B - Predicted Truth

Formula

$$DC = \frac{2 * |A \cap B|}{|A \cup B|}$$

A.2.7 ROC AUC

When we need to check or visualise the performance of a multi-class classification problem, we use ROC AUC. It measures performance at various thresholds settings. ROC is a probability curve and AUC represents the degree or measure of separability.

It shows how good a model is at distinguishing between classes. A score of 1.0 means it is perfect. A score of 0.5 means the model is unable to distinguish between the positive and negative class i.e. no better than random. A score of 0.0 means the model always makes the opposite decision from the correct one.

Terminology

AUC - Area Under the Curve. Represents the degree or measure of separability
ROC - Receiver Operating Characteristics. It is a probability curve

Bibliography

[1] Levenson RM, Krupinski EA, Navarro VM, Wasserman EA. Pigeons (Columba livia) as Trainable Observers of Pathology and Radiology Breast Cancer Images. PLOS ONE. 2015 Nov;10(11):e0141357. Available from: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0141357.

[2] Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, et al. Dermatologist-level classification of skin cancer with deep neural networks. Nature. 2017 Feb;542(7639):115–118. Available from: https://www.nature.com/articles/nature21056.

[3] Ayer T, Alagoz O, Chhatwal J, Shavlik JW, Kahn CE, Burnside ES. Breast cancer risk estimation with artificial neural networks revisited. Cancer. 2010 Jul;116(14):3310–3321. Available from: https://acsjournals.onlinelibrary.wiley.com/doi/10.1002/cncr.25081.

[4] Stern J. The Fragmentation of Health Data – Datavant;. Available from: https://datavant.com/2018/08/01/the-fragmentation-of-health-data/.

[5] Frank O. Opinion | Donate Your Health Care Data Today. The New York Times. 2019 Oct;Available from: https://www.nytimes.com/2019/10/02/opinion/health-care-data-privacy.html.

[6] Wikipedia F. General Data Protection Regulation. Wikipedia; 2020. Page Version ID: 933500662. Available from: https://en.wikipedia.org/w/index.php?title=General_Data_Protection_Regulation&oldid=933500662.

[7] Wikipedia F. Health Insurance Portability and Accountability Act. Wikipedia; 2019. Page Version ID: 932269315. Available from: https://en.wikipedia.org/w/index.php?title=Health_Insurance_Portability_and_Accountability_Act&oldid=932269315.

[8] Irvin J, Rajpurkar P, Ko M, Yu Y, Ciurea-Ilcus S, Chute C, et al. CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison. arXiv:190107031 [cs, eess]. 2019 Jan;ArXiv: 1901.07031. Available from: http://arxiv.org/abs/1901.07031.

[9] Menze BH, Jakab A, Bauer S, Kalpathy-Cramer J, Farahani K, Kirby J, et al. The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). IEEE Transactions on Medical Imaging. 2015 Oct;34(10):1993–2024.

[10] Bakas S, Akbari H, Sotiras A, Bilello M, Rozycki M, Kirby JS, et al. Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features. Scientific Data. 2017;4:170117.

[11] Bakas S, Reyes M, Jakab A, Bauer S, Rempfler M, Crimi A, et al. Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge. arXiv:181102629 [cs, stat]. 2019 Apr;ArXiv: 1811.02629. Available from: http://arxiv.org/abs/1811.02629.

[12] LeCun Y, Cortes C, Burges CJC. MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges; 1999. Available from: http://yann.lecun.com/exdb/mnist/.

[13] Cohen G, Afshar S, Tapson J, van Schaik A. EMNIST: an extension of MNIST to handwritten letters. arXiv:170205373 [cs]. 2017 Mar;ArXiv: 1702.05373. Available from: http://arxiv.org/abs/1702.05373.

[14] Caldas S, Duddu SMK, Wu P, Li T, Konečný J, McMahan HB, et al. LEAF: A Benchmark for Federated Settings. arXiv:181201097 [cs, stat]. 2019 Dec;ArXiv: 1812.01097. Available from: http://arxiv.org/abs/1812.01097.

[15] Luo J, Wu X, Luo Y, Huang A, Huang Y, Liu Y, et al. Real-World Image Datasets for Federated Learning. arXiv:191011089 [cs, stat]. 2019 Dec;ArXiv: 1910.11089. Available from: http://arxiv.org/abs/1910.11089.

[16] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, et al. ImageNet Large Scale Visual Recognition Challenge. arXiv:14090575 [cs]. 2015 Jan;ArXiv: 1409.0575. Available from: http://arxiv.org/abs/1409.0575.

[17] Zeng T, Wu B, Ji S. DeepEM3D: approaching human-level performance on 3D anisotropic EM image segmentation. Bioinformatics. 2017 Aug;33(16):2555–2562. Available from: https://academic.oup.com/bioinformatics/article/33/16/2555/3096435.

[18] Amazon. Amazon Go;. Available from: https://www.amazon.com/b?ie=UTF8&node=16008589011.

[19] Moshakis A. Nation of shoplifters: the rise of supermarket self-checkout scams. The Observer. 2018 May;Available from: https://www.theguardian.com/global/2018/may/20/nation-of-shoplifters-supermarket-self-checkout.

[20] Waymo. Waymo - We're building the World's Most Experienced Driver;. Available from: https://waymo.com/.

[21] Tesla. Autopilot;. Available from: https://www.tesla.com/autopilot.

[22] VolvoTrucks. Automated Trucks | Volvo Trucks;. Available from: https://www.volvotrucks.com/en-en/about-us/automation.html.

[23] EricssonIOT. Self-driving buses in Stockholm, Sweden; 2018. Available from: https://www.ericsson.com/en/internet-of-things/trending/driverless-buses-in-stockholm-sweden.

[24] WHO. Road traffic injuries; 2018. Available from: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries.

[25] Google. Nest Cam IQ Outdoor - Outdoor Home CCTV System - Google Store;. Available from: https://store.google.com/gb/product/nest_cam_iq_outdoor.

[26] Gauss. Triton^{TM} AI-enabled platform for real time monitoring of surgical blood loss; 2019. Available from: http://www.gausssurgical.com.

[27] McKinney SM, Sieniek M, Godbole V, Godwin J, Antropova N, Ashrafian H, et al. International evaluation of an AI system for breast cancer screening. Nature. 2020 Jan;577(7788):89–94. Available from: https://doi.org/10.1038/s41586-019-1799-6.

[28] Seibert JA. Archiving, Chapter 2: Medical Image Data Characteristics - Society for Imaging Informatics in Medicine;. Available from: https://siim.org/page/archiving_chapter2.

[29] Bankman I. Handbook of Medical Imaging: Processing and Analysis Management; pp772. 2nd ed. Academic Press; 2000.

[30] O'Shea K, Nash R. An Introduction to Convolutional Neural Networks. ArXiv e-prints. 2015 Nov;.

[31] Deshpande A. A Beginner's Guide To Understanding Convolutional Neural Networks; 2016. Available from: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/.

[32] Nwankpa C, Ijomah W, Gachagan A, Marshall S. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. arXiv:181103378 [cs]. 2018 Nov;ArXiv: 1811.03378. Available from: http://arxiv.org/abs/1811.03378.

[33] Karpathy A. CS231n Convolutional Neural Networks for Visual Recognition; 2015. Available from: http://cs231n.github.io/convolutional-networks/.

[34] Glocker B. Lecture 2: Introduction to Machine Learning. Presented at Imperial College London; 2020.

[35] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:14091556 [cs]. 2015 Apr;ArXiv: 1409.1556. Available from: http://arxiv.org/abs/1409.1556.

[36] Abdelbaki A. Computer Vision Lab SS16 - P-CNN features for Action Recognition; 2016.

[37] Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. Advances in Neural Information Processing Systems 25. Curran Associates, Inc.; 2012. p. 1097–1105. Available from: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[38] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going Deeper with Convolutions. arXiv:14094842 [cs]. 2014 Sep;ArXiv: 1409.4842. Available from: http://arxiv.org/abs/1409.4842.

[39] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. arXiv:151200567 [cs]. 2015 Dec;ArXiv: 1512.00567. Available from: http://arxiv.org/abs/1512.00567.

[40] Szegedy C, Ioffe S, Vanhoucke V, Alemi A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:160207261 [cs]. 2016 Aug;ArXiv: 1602.07261. Available from: http://arxiv.org/abs/1602.07261.

[41] Huang G, Liu Z, van der Maaten L, Weinberger KQ. Densely Connected Convolutional Networks. arXiv:160806993 [cs]. 2018 Jan;ArXiv: 1608.06993. Available from: http://arxiv.org/abs/1608.06993.

[42] Salem M, Taheri S, Yuan JS. Utilizing Transfer Learning and Homomorphic Encryption in a Privacy Preserving and Secure Biometric Recognition System. Computers. 2018 Dec;8:3.

[43] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. arXiv:13112524 [cs]. 2014 Oct;ArXiv: 1311.2524. Available from: http://arxiv.org/abs/1311.2524.

[44] Everingham M, Van-Gool L, Williams CKI, Winn J, Zisserman A. The PASCAL Visual Object Classes Challenge 2007 (VOC2007); 2007. Available from: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/.

[45] Girshick R. Fast R-CNN. arXiv:150408083 [cs]. 2015 Sep;ArXiv: 1504.08083. Available from: http://arxiv.org/abs/1504.08083.

[46] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. 2015;.

[47] Redmon J, Divvala S, Girshick R, Farhadi A. You Only Look Once: Unified, Real-Time Object Detection. arXiv:150602640 [cs]. 2016 May;ArXiv: 1506.02640. Available from: http://arxiv.org/abs/1506.02640.

[48] Iglovikov V, Shvets A. TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation. arXiv:180105746 [cs]. 2018 Jan;ArXiv: 1801.05746. Available from: http://arxiv.org/abs/1801.05746.

[49] Cicek O, Abdulkadir A, Lienkamp SS, Brox T, Ronneberger O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. arXiv:160606650 [cs]. 2016 Jun;ArXiv: 1606.06650. Available from: http://arxiv.org/abs/1606.06650.

[50] Isensee F, Kickingereder P, Wick W, Bendszus M, Maier-Hein KH. Brain Tumor Segmentation and Radiomics Survival Prediction: Contribution to the BRATS 2017 Challenge. arXiv:180210508 [cs]. 2018 Feb;ArXiv: 1802.10508 version: 1. Available from: http://arxiv.org/abs/1802.10508.

[51] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:150504597 [cs]. 2015 May;ArXiv: 1505.04597. Available from: http://arxiv.org/abs/1505.04597.

[52] Sheth AP, Larson JA. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Comput Surv. 1990 Sep;22(3):183–236. Available from: https://doi.org/10.1145/96602.96604.

[53] Kurze T, Klems M, Bermbach D, Lenk A, Tai S, Kunze M. Cloud federation. Cloud Computing. 2011;p. 32–38.

[54] University MLD Carnegie Mellon. Federated Learning: Challenges, Methods, and Future Directions; 2019. Available from: https://blog.ml.cmu.edu/2019/11/12/federated-learning-challenges-methods-and-future-directions/.

[55] Li T, Sahu AK, Talwalkar A, Smith V. Federated Learning: Challenges, Methods, and Future Directions. arXiv:190807873 [cs, stat]. 2019 Aug;ArXiv: 1908.07873. Available from: http://arxiv.org/abs/1908.07873.

[56] Li Q, Wen Z, Wu Z, Hu S, Wang N, He B. A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection. arXiv:190709693 [cs, stat]. 2019 Dec;ArXiv: 1907.09693. Available from: http://arxiv.org/abs/1907.09693.

[57] López PG, Montresor A, Epema DHJ, Datta A, Higashino T, Iamnitchi A, et al. Edge-centric Computing: Vision and Challenges. Computer Communication Review. 2015;45:37–42.

[58] Bonomi F, Milito RA, Zhu J, Addepalli S. Fog computing and its role in the internet of things. In: MCC '12; 2012. .

[59] Kuflik T, Kay J, Kummerfeld B. Challenges and solutions of ubiquitous user modeling. In: Ubiquitous display environments. Springer; 2012. p. 7–30.

[60] Roy AG, Siddiqui S, Pölsterl S, Navab N, Wachinger C. BrainTorrent: A Peer-to-Peer Environment for Decentralized Federated Learning. arXiv:190506731 [cs, stat]. 2019 May;ArXiv: 1905.06731. Available from: http://arxiv.org/abs/1905.06731.

[61] Bonawitz K, Eichner H, Grieskamp W, Huba D, Ingerman A, Ivanov V, et al. Towards Federated Learning at Scale: System Design. arXiv:190201046 [cs, stat]. 2019 Mar;ArXiv: 1902.01046. Available from: http://arxiv.org/abs/1902.01046.

[62] Kuchler H. Pharma groups combine to promote drug discovery with AI; 2019. Available from: https://www.ft.com/content/ef7be832-86d0-11e9-a028-86cea8523dc2.

[63] Sheller MJ, Reina GA, Edwards B, Martin J, Bakas S. Multi-Institutional Deep Learning Modeling Without Sharing Patient Data: A Feasibility Study on Brain Tumor Segmentation. arXiv:181004304 [cs, stat]. 2018 Oct;ArXiv: 1810.04304. Available from: http://arxiv.org/abs/1810.04304.

[64] Li W, Milletarì F, Xu D, Rieke N, Hancox J, Zhu W, et al. Privacy-preserving Federated Brain Tumour Segmentation. arXiv:191000962 [cs]. 2019 Oct;ArXiv: 1910.00962. Available from: http://arxiv.org/abs/1910.00962.

[65] Vepakomma P, Gupta O, Swedish T, Raskar R. Split learning for health: Distributed deep learning without sharing raw patient data. arXiv:181200564 [cs, stat]. 2018 Dec;ArXiv: 1812.00564. Available from: http://arxiv.org/abs/1812.00564.

[66] Kang J, Xiong Z, Niyato D, Yu H, Liang YC, Kim DI. Incentive Design for Efficient Federated Learning in Mobile Networks: A Contract Theory Approach. arXiv:190507479 [cs]. 2019 Oct;ArXiv: 1905.07479. Available from: http://arxiv.org/abs/1905.07479.

[67] McMahan HB, Moore E, Ramage D, Hampson S, Arcas BAy. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv:160205629 [cs]. 2017 Feb;ArXiv: 1602.05629. Available from: http://arxiv.org/abs/1602.05629.

[68] Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated Optimization in Heterogeneous Networks. arXiv:181206127 [cs, stat]. 2018 Dec;ArXiv: 1812.06127. Available from: http://arxiv.org/abs/1812.06127.

[69] Tang H, Gan S, Zhang C, Zhang T, Liu J. Communication Compression for Decentralized Training. arXiv:180306443 [cs, stat]. 2019 Jan;ArXiv: 1803.06443. Available from: http://arxiv.org/abs/1803.06443.

[70] Bost R, Popa RA, Tu S, Goldwasser S. Machine learning classification over encrypted data. vol. 4324; 2015. p. 4325. Available from: https://eprint.iacr.org/2014/331.pdf.

[71] Mikulic M. Diagnostic imaging market share top medtech companies 2017 and 2024; 2019. Available from: https://www.statista.com/statistics/331739/top-global-companies-by-diagnostic-imaging-market-share/.

[72] Potvin O, Khademi A, Chouinard I, Farokhian F, Dieumegarde L, Leppert I, et al. Measurement Variability Following MRI System Upgrade. Frontiers in Neurology. 2019;10. Available from: https://www.frontiersin.org/articles/10.3389/fneur.2019.00726/full.

[73] Shokri R, Stronati M, Song C, Shmatikov V. Membership Inference Attacks against Machine Learning Models. arXiv:161005820 [cs, stat]. 2017 Mar;ArXiv: 1610.05820. Available from: http://arxiv.org/abs/1610.05820.

[74] Hitaj B, Ateniese G, Perez-Cruz F. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. arXiv:170207464 [cs, stat]. 2017 Sep;ArXiv: 1702.07464. Available from: http://arxiv.org/abs/1702.07464.

[75] Melis L, Song C, De Cristofaro E, Shmatikov V. Exploiting Unintended Feature Leakage in Collaborative Learning. arXiv:180504049 [cs]. 2018 Nov;ArXiv: 1805.04049. Available from: http://arxiv.org/abs/1805.04049.

[76] Bhagoji AN. Lecture: IBM Research - Model Poisoning Attacks in Federated Learning. Presented at Princeton University; 2018. Available from: http://www.princeton.edu/~abhagoji/files/SecML_2018_fed_learn_poison.pdf.

[77] Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V. How To Backdoor Federated Learning. arXiv:180700459 [cs]. 2019 Aug;ArXiv: 1807.00459. Available from: http://arxiv.org/abs/1807.00459.

[78] Blanchard P, Guerraoui R, Stainer J, others. Machine learning with adversaries: Byzantine tolerant gradient descent. In: Advances in Neural Information Processing Systems; 2017. p. 119–129. Available from: https://papers.nips.cc/paper/6617-machine-learning-with-adversaries-byzantine-tolerant-gradient-descent.pdf.

[79] Dwork C, Roth A. The Algorithmic Foundations of Differential Privacy. Found Trends Theor Comput Sci. 2014 Aug;9(3–4):211–407. Available from: https://doi.org/10.1561/0400000042.

[80] Truex S, Baracaldo N, Anwar A, Steinke T, Ludwig H, Zhang R, et al. A Hybrid Approach to Privacy-Preserving Federated Learning. arXiv:181203224 [cs, stat]. 2019 Aug;ArXiv: 1812.03224. Available from: http://arxiv.org/abs/1812.03224.

[81] Dulay N. Lecture: Privacy Engineering Part II. Presented at Imperial College London; 2019.

[82] Zama; 2019. Available from: https://zama.ai/.

[83] Hardy S, Henecka W, Ivey-Law H, Nock R, Patrini G, Smith G, et al. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. arXiv:171110677 [cs]. 2017 Nov;ArXiv: 1711.10677. Available from: http://arxiv.org/abs/1711.10677.

[84] V Nikolaenko, U Weinsberg, S Ioannidis, M Joye, D Boneh, N Taft. Privacy-Preserving Ridge Regression on Hundreds of Millions of Records. In: 2013 IEEE Symposium on Security and Privacy; 2013. p. 334–348.

[85] M Sabt, M Achemlal, A Bouabdallah. Trusted Execution Environment: What It is, and What It is Not. In: 2015 IEEE Trustcom/BigDataSE/ISPA. vol. 1; 2015. p. 57–64.

[86] J Ekberg, K Kostiainen, N Asokan. The Untapped Potential of Trusted Execution Environments on Mobile Devices. IEEE Security & Privacy. 2014 Aug;12(4):29–37.

[87] Gu Z, Jamjoom H, Su D, Huang H, Zhang J, Ma T, et al. Reaching data confidentiality and model accountability on the caltrain. In: 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). IEEE; 2019. p. 336–348. Available from: https://arxiv.org/abs/1812.03230.

[88] Gupta O, Raskar R. Distributed learning of deep neural network over multiple agents. arXiv:181006060 [cs, stat]. 2018 Oct;ArXiv: 1810.06060. Available from: http://arxiv.org/abs/1810.06060.

[89] Bock JR. A Deep Learning Model of Perception in Color-Letter Synesthesia. Big Data and Cognitive Computing. 2018 Mar;2(1):8. Available from: https://www.mdpi.com/2504-2289/2/1/8.

[90] WeBank, Shenzen, China. Federated Learning White Paper V1.0. 2018 Sep;1.0. Available from: https://www.fedai.org/static/flwp-en.pdf.

[91] CheXpert: A Large Dataset of Chest X-Rays and Competition for Automated Chest X-Ray Interpretation.;. Available from: https://stanfordmlgroup.github.io/competitions/chexpert/.

[92] OpenMined, PySyft. Github Repository, Privacy Preserving Deep Learning Tutorial with PyTorch & PySyft; 2019. Available from: https://github.com/OpenMined/PySyft/tree/master/examples/tutorials.

[93] MICCAI BRATS - The Multimodal Brain Tumor Segmentation Challenge;. Available from: http://braintumorsegmentation.org/.