A Broader Impact

As this study solely involves the repurposing of existing open-source materials and benchmarking, there are limited risks associated within the study itself. However, it should be noted that all datasets included in this study could be subject to biases originated during the collection process, such as gender or ethnicity biases. Unfortunately, on the images' datasets (5 datasets out of 7), the sources of such potential biases cannot be easily checked, since data were properly pseudonymised and image-based medical records cannot be straightforwardly tied back to a particular ethnicity or gender by non-medical experts. Nevertheless, as our work exposes more clearly some metadata (e.g. geographical origin) of the datasets, it might help revealing underlying geographical biases, and thus help building more heterogeneous benchmarks, as expected in real scenarios for FL.

As we focused on simplicity and ease of use, the current benchmark does not encompass privacy metrics. However, privacy is of paramount important in healthcare cross-silo FL, and we urge the community not to ignore these aspects. Thanks to the modularity of FLamby, it is easy to add privacy components, as we show in a DP example in Appendix L.1. Thus, we hope FLamby will help tackle privacy questions in healthcare cross-silo FL.

B Datasets repository and Authors Statement

B.1 Dataset repository.

The code is now available at https://github.com/owkin/FLamby

The code respects best practices for reproducibility and dataset sharing. The installation process is detailed and allow to install only requirements of specific datasets. Regarding code readability, the code is linted with black and flake8 and most functions have docstrings. Documentation is automatically generated from markdown with sphinx, including tutorials. Unit tests ensure FL strategies perform correctly.

Regarding licenses, all datasets documented in this repository come with links towards data terms or licenses. Every time a user downloads a dataset for the first time, he or she is prompted with a link towards the data terms or license, and has to explicitly agree to it in order to proceed.

B.2 Maintenance plan

We will follow a maintenance plan to ensure the code remains correct and the datasets provided by the suite follow adequate standards. In particular, this maintenance plan encompasses:

- Fixing bugs affecting the correctness of the code, whether brought out by the community or ourselves;
- Ensuring security updates in the dependencies are performed;
- Regarding datasets, reviewing, on a monthly basis, potential updates of the datasets referenced in the suite, including but not limited to patients opting out or ethical concerns raised by the work. Such modification may go to the extent of a full revocation of the related dataset if need be;
- Reviewing contributions from the community, whether they are related to the benchmark or to incorporating new datasets to the suite, ensuring they are at the highest standards.

B.3 Authors statement.

As authors of this repository and article we bear all responsibility in case of violation of rights and licenses. We have added a disclaimer on the repository to invite original datasets creators to open issues regarding any license related matters.

Table 2: Information for the different clients in Camelyon16

Number	Client	Dataset size	Train	Test
0	RUMC	243	169	74
1	UMCU	156	101	55

C Fed-Camelyon16

C.1 Description

Camelyon16 [LBEB$^+$18] is a histopathology dataset of 399 digitized breast biopsies' slides with or without tumor collected from two hospitals: Radboud University Medical Center (RUMC) and University Medical Center Utrecht (UMCU). The client information can be read directly from the training slides as the first 170 slides belong to RUMC and the others to UMCU. For the test slides we use a manual approach based on clustering to recover the centres and visual inspection. The slides split are summarized Table 2

C.2 License and Ethics

The Camelyon data is open access (CC0)[1].

The collection of the data was approved by the local ethics committee (Commissie Mensgebonden Onderzoek regio Arnhem - Nijmegen) under 2016-2761, and the need for informed consent was waived [LBEB$^+$18].

C.3 Download and preprocessing

As the original dataset is stored in Google Drive, we provide code relying on the Google drive API's python SDK to batch download all the images (800GB) efficiently. It requires the user to have a Google account and to setup a service account. Detailed instructions are provided in the repository.

Once all tif images have been downloaded we use the histolab package[MAB] to tile the slides with patches of size 224x224 at the second level of the image pyramid corresponding to ≈ 0.5 µm / pixel. We only keep tiles with sufficient amount of tissue on them thanks to the `check_tissue=True` option of the `GridTiler` histolab object. We then perform Imagenet preprocessing [HZRS16] and extract a 2048 feature vector from an Imagenet-pretrained Resnet50 [HZRS16] on each patch. As slides have different amount of matter this produces a variable number of features per slide. We subsequently save those features in the numpy format [VDWCV11].

C.4 Task

Each of this slide represented as a bag of features has a binary label indicating the presence of a tumour on the breast. The task is to predict if a slide contains a tumour or not so it is framed as a binary classification problem under the Multiple Instance Learning paradigm[ITW].

C.5 Baseline, loss function and evaluation

Loss function We use a traditional binary cross entropy loss [Goo92] and evaluate the performance with the Area under the ROC curve or AUC [Bra97].

Baseline Model We use the DeepMIL[ITW] architecture that uses attention to learn to weight patch features importance in an end to end fashion. The network architecture is specified in the code. The model trains in approximately 5 minutes on a P100.

[1] `https://camelyon17.grand-challenge.org/Data/`

Optimization parameters We use a batch size of 16 with Adam [KB14] with a learning rate of 0.001. Both sets of hyperparameters mentioned above used for the network architecture and optimization are taken from [DCM$^+$20], we change the number of pooled epochs to 45 in order to be able to do more than one synchronization rounds when performing federated experiments.

Hyperparameter Search For the pooled dataset benchmark we use the configuration described above without further tuning. For FL strategies we use the following hyperparameter grid: for clients' learning rates (all strategies) {1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1.0}; for server size learning rate (for Scaffold and FedOpt strategies) {1e-3, 1e-2, 1e-1, 1.0, 10.0}, and for FedProx only, μ belongs to {1e-2, 1e-1, 1.0}.

D Fed-LIDC-IDRI

D.1 Description

LIDC-IDRI [AIMB$^+$11, IMB$^+$15, CVS$^+$13] is part of The Cancer Imaging Archive (TCIA) database [CVS$^+$13] with 1009 lung CT-scans (3D images), on which radiologists annotated the presence of nodules.

We split the dataset in 4 different clients that correspond to different medical imagery machine manufacturers, which were previously shown to be a source of heterogeneity in CT image quality [FDZ$^+$15]. We end up with 661 samples gathered by GE Medical Systems, 205 by Siemens, 69 by Toshiba, and 74 by Philips scanner. These datasets are further split in training and testing sets that contain respectively 80% and 20% of the data. This split is stratified according to clients, so that proportions are respected. The exact distribution of the samples between clients are given in Table 3

Table 3: Information for the different clients in LIDC IDRI.

Number	Client	Dataset size	Train	Test
0	GE MEDICAL SYSTEMS	661	530	131
1	Philips	74	59	15
2	SIEMENS	205	164	41
3	TOSHIBA	69	55	14

D.2 License and Ethics

The users of this data must abide by the Data Usage Policies listed on the TCIA webpage under LIDC (links are provided in the README of the LIDC dataset in FLamby repository). It is licensed under a Creative Commons Attribution 3.0 Unported License.

Data was anonymized in each local center before being uploaded to the central repository [AIMB$^+$11]. Further, as per the terms of use of TCIA[2], "users must agree not to generate and use information in a manner that could allow the identities of research participants to be readily ascertained".

D.3 Download and preprocessing

Instructions in the README.md of the LIDC-IDRI dataset in FLamby repository allow to download images and average annotation masks from the TCIA initiative. Flamby code then permits conversion from DICOMs to nifti files to facilitate further analysis.

D.3.1 Preprocessing and sampling

Raw CT scans have varying dimensions which must be standardized prior to training. Therefore, as a first step we resize them to a common $(384, 384, 384)$ shape by cropping dimensions in excess and

[2] `https://wiki.cancerimagingarchive.net/display/Public/Data+Usage+Policies+and+Restrictions`

reflection-padding missing dimensions. During training, this operation is performed in the same way both on the CT scans and the ground truth masks.

Next, the images are normalized. CT scan voxels are originally expressed in the Hounsfield unit (HU) [Fee10] scale: roughly $-1,000$ HU for air, 0 HU for water, and $1,000$ HU for bone. We clip the images to the $[-1024, 600]$ range, add 1024, and then divide voxels by 1624 to obtain values ranging in $[0, 1]$.

D.4 Task

We benchmark federated learning strategies on a nodule segmentation task using a VNet [MNA16]. More precisely, we aim to maximize the DICE coefficient [Dic45] between predictions and the annotated ground truths. For reference, the baseline model trained on the pooled training set achieves a DICE of 41% on the pooled test set.

D.5 Baseline, loss function and evaluation

Sampling The resulting images of size $(384, 384, 384)$ are too voluminous to fit in the memory of most GPUs. Hence, during training we feed the model with sampled patches of size $(128, 128, 128)$. We sample 2 patches from each (image, mask) pair. This implies that batches are constituted of two $(128, 128, 128)$ patches drawn from the same CT scan. As lung nodules are relatively small and rare, there is a strong class imbalance in the LIDC dataset. To alleviate this issue, we ensure that one of the sampled patches contains nodule voxels (by centering it on a nodule voxel drawn at random), and sample the other completely at random. To account for possible nodules at the edges of CT scans, a padding of half the patch size is applied to each dimension of the image prior to sampling.

Loss function Our objective is to maximize the DICE coefficient [Dic45]. However, we observed that maximizing DICE alone during training yielded poor results at inference time on regions that do not contain nodules. To force the model to account for class imbalance, we added a small balanced cross-entropy term [see Jad20]. Hence, we minimize the following loss:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = (1 - \mathrm{DICE}(\mathbf{y}, \hat{\mathbf{y}})) + 0.1 \times \mathrm{BCE}(\mathbf{y}, \hat{\mathbf{y}}), \tag{2}$$

with

$$\mathrm{DICE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{2 \sum_{i=1}^{n} y_i \hat{y}_i}{\sum_{i=1}^{n} y_i + \sum_{i=1}^{n} \hat{y}_i}, \tag{3}$$

and

$$\mathrm{BCE}(\mathbf{y}, \hat{\mathbf{y}}) = -\alpha \sum_{i=1}^{n} y_i \log(\hat{y}_i) - \sum_{i=1}^{n} (1 - y_i) \log(1 - \hat{y}_i), \tag{4}$$

where $\alpha = \left(\max(\frac{1}{n} \sum_{i=1}^{n} y_i, 10^{-7})\right)^{-1} - 1$.

Baseline Model We implement a VNet [MNA16], following the architecture proposed therein. During training, we use dropout ($p = 0.25$). The final layer produces a single output, which is passed through a sigmoid function to encode the probability that each voxel corresponds to a nodule. The model trains in approximately 48 hours on a P100.

Optimization parameters We optimize the VNet using RMSprop, with an initial learning rate of 10^{-2}. We run 100 epochs, multiplying the learning rate by 0.95 every 10 epochs.

Hyperparameters search LIDC FL trainings take approximately 70 hours on a P100 so because of time constraints we could not use an extensive grid search as for other datasets. The final parameters we use are reported in Section J.3.

E Fed-IXI

E.1 Description

IXI Tiny [ixi] is a light version of the dataset IXI, a multimodal brain imaging dataset of almost 600 subjects [dt]. This lighter version provides T1-weighted brain MR images for a subset of 566 subjects,

Table 4: Demographics information for Fed-IXI.

Hospital Name	Sex	Dataset size	Age	Age Range
Guys	Female	184	53.23 ± 15.25	20 - 80
	Male	144	51.02 ± 17.26	20 - 86
HH	Female	93	50.28 ± 16.93	20 - 81
	Male	85	44.43 ± 15.67	20 - 73
IOP	Female	44	43.90 ± 18.43	19 - 86
	Male	24	39.57 ± 12.46	23 - 70

along with a set of corresponding brain image segmentations labels, taking the form of binary image masks.

Brain image masks isolate the brain pixels from the other head components, such as the eyes, skin, and fat. For the supervised task, brain image segmentation masks (labels) were obtained through automatic whole-brain extraction on the T1-weighted MRI data, using the unsupervised brain extraction tool ROBEX [ILTT11].

The images come from three different London hospitals: Guys (Guy's Hospital, manufacturer code 0), HH (Hammersmith Hospital, manufacturer code 1), both using a Philips 1.5T system, and IOP (Institute of Psychiatry, manufacturer code 2), using a GE 1.5T system. We split this dataset in training and testing sets, respectively containing 80% and 20% of the data. The split is also stratified according to hospitals to preserve data proportions. In other words, we define one test set on each hospital. Table 4 provides demographic information for this dataset.

E.2 License and Ethics

This dataset is licensed under a Creative Commons Attribution Share Alike 3.0 Unported (CC BY-SA 3.) license.

The dataset website does not provide any information regarding data collection ethics. However, the original dataset was collected as part of the IXI - Information eXtraction from Images (EPSRC GR/S21533/02) project, and thus funded by UK Research and Innovation (UKRI). As part of its terms and conditions[3], the UKRI demands that all funded projects are "carried out in accordance with all applicable ethical, legal and regulatory requirements" (RGC 2.2).

E.3 Downloading and preprocessing

We provide a helper script to download the dataset from an Amazon S3 bucket.

Preprocessing and sampling We use a fixed preprocessing step that is performed once. Brain scans are first geometrically aligned to a common anatomical space (MNI template) through affine registration estimated with NiftyReg [Mod]. Images are then reoriented using ITK [MLI+14]. Finally, intensities are normalized in each image (based on the entire image histogram), and the image volumes are resized from 83x44x55 to 48×60×48 voxels.

E.4 Task

The task is to segment the brain on the volume. The prediction is evaluated with the DICE score, which is the symmetric of the DICE loss with respect to $1/2$.

[3] `https://www.ukri.org/wp-content/uploads/2022/04/UKRI-050422-FullEconomicCostingGrantTermsConditions-Apr2`
`pdf`

E.5 Baseline, loss function and evaluation

Loss function The model was directly trained for the DICE loss [Dic45], defined as

$$\ell_{DICE} = 1 - S_{DICE} = 1 - \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \epsilon},$$

where TP, FP, and FN stand for the true positive rate, false positive rate, and false negative rate, respectively, and $\epsilon = 10^{-9}$ ensures numerical stability.

Baseline Model We use a UNet model taking the individual T1 image as input, to predict the associated binary brain mask. The UNet model is a standard type of convolution neural network architecture commonly used in biomedical image segmentation tasks [RFB15]. It is specifically used to perform semantic segmentation, meaning that each voxel of the image volume is classified. We can also refer to this task as a dense prediction. The model trains in approximately 5 minutes on a P100.

Optimization parameters The UNet is optimized with a batch size of 2 and a learning rate of 10^{-3} with the AdamW optimizer. The best architecture used batch normalization, max-pooling, linear upsampling, zero-padding of size 1, PReLU activation functions, and 3 encoding blocks.

Hyperparameters search We do not change parameters for the pooled baseline. For FedAvg and Cyclic, we optimized the learning rate over the values {0.1, 0.01, 0.001, 0.0001, 0.00001}. For FedYogi, FedAdam, FedAdagrad and Scaffold, our search grid space was {0.1, 0.01, 0.001, 0.0001, 0.00001} and {10, 1, 0.1, 0.01, 0.001} for the learning rate and the server learning rate respectively. For FedProx, our search space contained {0.1, 0.01} and {1, 0.1, 0.01} sets for learning rate and μ respectively.

F Fed-TCGA-BRCA

F.1 Description

Our dataset comes from The Cancer Genome Atlas (TCGA)'s Genomics Data Commons (GDC) portal [Net] more specifically from the BReast CAncer study (BRCA), which includes features gathered from 1066 patients. We use the material produced by Liu *et al.* [LLH+18] as a base file that we further preprocess with one-hot encoding following [AMM+20]. This produces a lightweight tabular dataset with 39 input features. Patients' labels are overall survival time and event status with the event being death. We use the Tissue Source Site metadata to split data based on extraction site, grouped into geographic regions to obtain large enough clients. We end up with **6** clients: USA (Northeast, South, Middlewest, West), Canada and Europe, with patient counts varying from 51 to 311. Our train-test split of the data is stratified per client and event. Table 5 provides details per client for this dataset. Table 6 provides results of pair-wise log-rank tests between the different clients.

F.2 License and Ethics

The data terms can be found on the GDC website[4]. In particular, these terms bind users as to "not attempt to identify individual human research participants from whom the data were obtained".

As per the TCGA policies[5], special care was devoted to ensure privacy protection of research subjects, including but not limited to HIPAA compliance. Note that we do not use the genetic part of TCGA whose access is restricted due to its sensitivity.

F.3 Downloading and preprocessing

The pooled TCGA-BRCA dataset requires no downloading or extra-preprocessing as the preprocessed data is now a part of the Flamby repository.

[4] `https://gdc.cancer.gov/access-data/data-access-processes-and-tools`

[5] `https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga/history/policies`

Table 5: Information for the different clients (geographical regions) in Fed-TCGA-BRCA.

Number	Client	Dataset size	Train	Test	Censorship ratio
0	USA Northeast	311	248	63	81
1	USA South	196	156	40	80
2	USA West	206	164	42	89
3	USA Midwest	162	129	33	88
4	Europe	162	129	33	94
5	Canada	51	40	11	94

Table 6: Pairwise log-rank p-values on the Fed-TCGA-BRCA clients. Some clients have significant differences for a 10% significance threshold.

Compared with	Client 1	Client 2	Client 3	Client 4	Client 5
Client 0	0.289682	0.066374	0.039892	0.576926	0.200366
Client 1		0.192075	0.161797	0.92917	0.541251
Client 2			0.954475	0.720912	0.256973
Client 3				0.576374	0.127662
Client 4					0.441106

F.4 Task

The task consists in predicting survival outcomes [Jen05] based on the patients' clinical tabular data (39 features overall). This survival task is akin to a ranking problem with the score of each sample being known either directly or only by lower bound. Indeed, some patients leave the study before the event of interest is observed, and are labelled as right-censored. Survival analysis aims at solving this type of ranking problem while leveraging right-censored data. The censoring ratio in the TCGA-BRCA study is 86%.

The ranking is evaluated by using the concordance index (C-index) that measures the percentage of correctly ranked pairs while taking censorship into account:

$$\mathrm{C-index} = \mathbb{E}_{\substack{i:\delta_i=1 \\ j:t_j>t_i}} \left[\mathbb{1}_{[\eta_j<\eta_i]} \right] \tag{5}$$

where η_i is a risk score assigned by our model to a patient i. In our case of linear Cox proportional hazard models we use $\eta_i = \beta^T x_i$, where x_i are the features for patient i and β the learned weights, see Section F.5.

Optimization parameters For the pooled dataset benchmark, we use the Adam optimizer [KB14], with a learning rate of 0.1 and a batch size of 8 for 30 epochs.

F.5 Baseline, loss function and evaluation

Survival analysis background Let T be the random time-to-death taken from the patient's population under study. The survival function S is defined as:

$$S(t) = Pr[T > t] \tag{6}$$

A patient is characterized by its vector of covariates x (clinical data in our case), an observed time point t and an indicator $\delta \in \{0, 1\}$ where $\delta = 0$ if the event has been censored. A key quantity characterizing the distribution of S is the hazard function h. It is the instantaneous rate of occurrence of the event given that it has not yet happened for a patient:

$$h(t, x) = \lim_{dt \to 0} \frac{Pr[t < T < t + dt | x, T > t]}{dt} \tag{7}$$

Loss function The simplest model in survival analysis is the linear Cox proportional hazard [Cox72]. This model assumes:

$$h(t, x) = h_0(t) \exp(\beta^T x) \tag{8}$$

where h_0 is the baseline hazard function (common to all patients and dependent on time only) and β is the vector of parameters of our linear model. β is estimated by minimization of the negative Cox partial log-likelihood, which compares relative risk ratios:

$$L(\beta) = - \sum_{i:\delta_i=1} \Big[\beta^T x_i - \log \big(\sum_{j:t_j>t_i} \exp(\beta^T x_j) \big) \Big] \tag{9}$$

where i and j index patients.

We minimize the negative Cox partial log-likelihood by gradient descent w.r.t. β.

As explained in [AMM+20] the Cox partial log-likelihood is not separable with respect to the samples: this means it cannot be expressed as a sum of terms each dependent on a single sample. Hence it is not separable with respect to the clients either. In this work, for simplicity, we decide to ignore this fact in the baseline: we treat each client's negative Cox partial log-likelihood independently of the others and apply any federated learning strategy logic to the resulting local gradients. Please refer to [AMM+20] for a more rigorous treatment of the federated survival analysis problem.

Baseline Model As a baseline, we use the aforementioned linear Cox proportional hazard model [Cox72]. The model trains in a matter of seconds on modern CPUs.

Hyperparameter Search All the federated learning strategies are tested with the SGD optimizer. We performed a grid search for the federated learning strategies hyperparameters . For FedAvg and Cyclic, we optimized the learning rate over the values {0.1, 0.01, 0.001, 0.0001, 0.00001}. For FedYogi, FedAdam, FedAdagrad and Scaffold, our search grid space was {0.1, 0.01, 0.001, 0.0001, 0.00001} and {10, 1, 0.1, 0.01, 0.001} for the learning rate and the server learning rate respectively. For FedProx, our search space contained {0.1, 0.01} and {1, 0.1, 0.01} sets for learning rate and μ respectively. The chosen hyperparameters from the HP search can be found in Section J.3.

G Fed-KiTS19

G.1 Description

The KiTS19 dataset [HIMH+20, HSK+19] stems from the Kidney Tumor Segmentation Challenge 2019 and contains CT scans of 210 patients along with the segmentation masks from 77 hospitals[6]. We only consider the training dataset of this challenge as the segmentation masks are not provided for the test dataset. We recover the hospital metadata and extract a **6**-client federated version of this dataset by removing hospitals with less than 10 training samples. Figures 3a and 3b show the repartition of patients per client before and after this client selection respectively. Table 7 provides further details of the train and test split at each selected client.

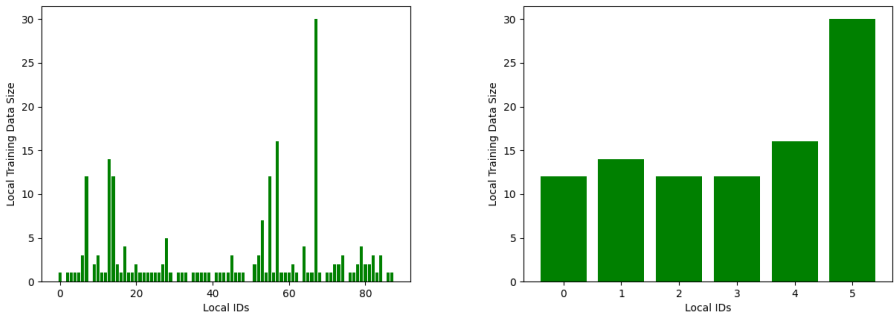
G.2 License and Ethics

This dataset is licensed under a Attribution-NonCommercial-ShareAlike 4.0 International (CC-BY-NC-SA) license[7].

The dataset collection was approved by the Institutional Review Board at the University of Minnesota as Study 1611M00821 [HSK+19].

[6] It is important to note that KiTS19 dataset does not come with the hospital information. We obtained the data distribution per client from one of the organizers of this challenge, Nicholas Heller, over email communication. We acknowledge the help of Nicholas Heller for sharing this valuable resource with us that helped us explore federated learning strategies with this dataset for the first time.

[7] `https://github.com/neheller/kits19/blob/master/LICENSE`



(a) Original KiTS19 (b) Fed-KiTS19

Figure 3: Patient distributions across hospitals of the original KiTS19 Dataset and the derived Data distribution of Fed-KiTS19

G.3 Downloading and Preprocessing

We use the official KiTS19 repository[8] to download the KiTS19 data. Next, we preprocess this dataset. The first step of the preprocessing is to clip the intensity. We clip the values of each image to the [5th percentile, 95th percentile] range, where 5th percentile and 95th percentile are calculated on the image intensities of each patient's case separately. After this step, we apply z-scale normalization, where we subtract the mean and divide by the standard deviation of the image intensities. Since KiTS19 dataset comes with inhomogeneous voxel spacing even for the patients data from the same silos, we resample the voxel spacings to the target spacing of 2.90x1.45x1.45 mm for all the samples.

G.4 Task

The task consists of both kidney and tumor segmentation, labeled 1 and 2, respectively. The background is labeled as 0. The score we consider on this dataset is the average of Kidney and Tumor DICE scores [Dic45].

G.5 Baseline, loss function and evaluation

Sampling The image size distribution of the samples of the KiTS19 dataset is heterogeneous. After the resampling detailed in section G.3, the median patient's data size is [116, 282, 282]. To make our model's computation memory efficient, we extract a patch of size [64, 192, 192] from each sample during the model training. The number of voxels belonging to the foreground classes (i.e. either Kidney or Tumor) is small compare to the number of voxels belonging to the background class. Therefore, we oversample the foreground classes when taking patches of the samples. More precisely, we use batches of size 2. Each batch contains one patch with the foreground oversampled. Furthermore, we split each silo's data into training and validation data with 80% and 20% split, respectively. All this pre-processing and patching is done using the nnU-Net library [IJK$^+$21].

Loss function We use the same loss function as proposed by nnU-Net [IJK$^+$21] for the KiTS19 dataset which is based on DICE [Dic45] and on the Cross Entropy loss. Both losses are summed with equal weight as shown in Equation (10),

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = (1 - \mathrm{DICE}(\mathbf{y}, \hat{\mathbf{y}})) + \mathrm{CE}(\mathbf{y}, \hat{\mathbf{y}}), \tag{10}$$

with

$$\mathrm{DICE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{2 \sum_{l=1}^{2} \sum_{i=1}^{n} y_i^l \hat{y}_i^l + \epsilon}{\sum_{l=1}^{2} (\sum_{i=1}^{n} y_i^l + \sum_{i=1}^{n} \hat{y}_i^l) + \epsilon}, \tag{11}$$

[8] `https://github.com/neheller/kits19`

Table 7: Information for the selected clients in Fed-KiTS19.

Local ID Number	Dataset size	Train	Test
0	12	9	3
1	14	11	3
2	12	9	3
3	12	9	3
4	16	12	4
5	30	24	6

and

$$\mathrm{CE}(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{i=1}^{n}\sum_{l=1}^{2} y_i^l \log \hat{y}_i^l, \tag{12}$$

where ϵ value is $1e^{-5}$ and n is the set of all pixels and 2 signifies the 2 class labels here, Kidney and Tumor, and where y_i^l is the one-hot encoding (0 or 1) for the label l and pixel i and y_i^l is the predicted probability for the same label l and pixel i.

Baseline Model During the training, we use nnU-Net [IJK$^+$21], with the architecture proposed therein for the KiTS19 dataset. We chose convolution kernels of sizes [[3,3,3],[3,3,3],[3,3,3],[3,3,3],[3,3,3]] and pool kernels of sizes [[2,2,2],[2,2,2],[2,2,2],[2,2,2],[1,2,2]]. The model trains in under 24 hours on a P100.

Optimization parameters In addition, we use Adam optimizer [KB14] with a learning rate of 0.0003 for 500 epochs to train our model. To evaluate the performance of the trained model, we evaluate the DICE score on the validation data for both classes, Kidney and tumor, and report the average of these two scores. We note that with 8000 epochs we can obtain higher performances, however at the expense of computational cost.

Hyperparameter Search For the pooled strategy results, we use the Adam Optimizer and 0.0003 learning rate, as used in nnU-Net work for KiTS19 dataset [IJK$^+$21]. For Cyclic and FedAvg, we optimized the learning rate over the values {0.3, 0.03, 0.003, 0.0003, 0.00003} and found that a learning rate of 0.3 provided the best results for both strategies. For FedYogi, FedAdam, FedAdagrad and Scaffold, our search grid space was {0.1, 0.01} and {0.001, 0.01, 0.1, 1} for the learning rate and the server learning rate respectively. In the best setting, the learning rate was 0.1 for all these strategies, and the server learning rate 0.1 for FedAdagrad, 0.01 for FedYogi and FedAdam and 1 for Scaffold. Likewise, for FedProx, our search space contained {0.1, 0.01} and {0.001, 0.01, 0.1, 1} sets for learning rate and μ respectively, and the best set of hyperparameters was 0.1 for the learning rage and 0.001 for μ.

H Fed-ISIC2019

H.1 Dataset description

The ISIC2019 challenge dataset [TRK18, CGC$^+$18, CCR$^+$19] contains 25,331 dermoscopy images collected in 4 hospitals. To the best of our knowledge, it is the largest public dataset of high-quality images of skin lesions. We restrict ourselves to 23,247 images from the public train set due to metadata availability reasons, which we re-split into train and test sets. The train-test split is static.

We split this dataset into 6 clients corresponding to different sites where images were taken with different imaging technologies. The ViDIR Group, Medical University of Vienna, Austria uses 3 different imaging systems representing evolving clinical practice: a Heine Dermaphot system using an immersion fluid, a DermLite™ FOTO and a MoleMax HD machine which gives rise to 3 clients. On top of this, the skin cancer practice of Cliff Rosendahl in Queensland, Australia, the Hospital

Clínic de Barcelona, Spain and the Memorial Sloan Kettering Cancer Center, New York give rise to 3 other different clients making a total of 6 clients. The biggest client counts 12413 images while the smallest counts 439. Table 8 provides details about the size of the different clients.

Table 8: Information for the different clients in Fed-ISIC2019.

Number	Client	Dataset size	Train	Test
0	Hospital Clínic de Barcelona	12413	9930	2483
1	ViDIR Group, Medical University of Vienna (MoleMax HD)	3954	3163	791
2	ViDIR Group, Medical University of Vienna (DermLite FOTO)	3363	2691	672
3	The skin cancer practice of Cliff Rosendahl	2259	1807	452
4	Memorial Sloan Kettering Cancer Center	819	655	164
5	ViDIR Group, Medical University of Vienna (Heine Dermaphot)	439	351	88

H.2 License and Ethics

This dataset is licensed under a Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) license[9].

As per the terms of use of the ISIC archive[10], one of the requirements for this dataset to have been hosted is that it is properly de-identified in accordance with applicable requirements and legislations.

H.3 Downloading and preprocessing

Instructions for downloading and preprocessing are available in the README of the Fed-ISIC2019 dataset inside the FLamby repository. As an offline preprocessing step, we follow recommendations and code from [Aro] by resizing images to the same shorter side of 224 pixels while maintaining their aspect ratio, and by normalizing images' brightness and contrast through a color consistency algorithm. The total size of the raw inputs is 9 GB.

H.4 Task

The task consists in image classification among 8 different classes: Melanoma, Melanocytic nevus, Basal cell carcinoma, Actinic keratosis, Benign keratosis, Dermatofibroma, Vascular lesion and Squamous cell carcinoma. Ground truth is established through histopathology, follow-up examination, expert consensus or microscopy. The ISIC2019 dataset has a high label imbalance with prevalence ranging from 49% to less than 1% depending on the class. We follow the ISIC challenge metric: we measure classification performance through balanced accuracy, defined as the average of the recalls calculated for each class. For balanced datasets, it is equal to accuracy. A random classifier would get a balanced accuracy equal to $1/C$ where C is the number of classes. Using balanced accuracy prevents the model from taking advantage of an imbalanced test set.

H.5 Baseline, loss function and evaluation

The choices made are inspired by [Aro], [GNS+19], and an analysis of the solutions that scored well at the ISIC challenge over the years.

Loss function Our pretrained EfficientNet is fine-tuned using a weighted focal loss [LGG+17]. It is calculated as follows:

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t) \tag{13}$$

where p_t is the probability output by our model for the ground-truth class, α_t is the weight of the ground-truth class (a weight is attributed to each class before training), γ is a hyperparameter (chosen

[9] `https://challenge.isic-archive.com/data/`

[10] `https://challenge.isic-archive.com/terms-of-use/`

at 2 in our work). The focal loss is very useful where there is class imbalance. To provide an intuition behind this focal loss, compared to Binary Cross Entropy, it gives the model a bit more freedom to take some risk when making predictions. The weights we use for our weighted focal loss are the inverse of the class proportions calculated over the pooled dataset. We assume these weights are available to all clients.

Baseline Model As a baseline classification model, we fine-tune an EfficientNet [TL19]. EfficientNets are the results of a simple uniform scaling of MobileNets and ResNet on all dimensions (depth/width/resolution). They show great accuracy and efficiency and transfer very well to other tasks. Our EfficientNet is pretrained on ImageNet, we use it as a feature extractor (1280 features) by replacing the output layer by a linear layer to get an output of dimension 8. On top of this, we use the data augmentations listed below to regularize our model. The model trains in under an hour on a P100, because we have to recompute EfficientNet features with dynamic data augmentations.

For training:

1. Random Scaling
2. Rotation
3. Random Brightness Contrast
4. Flipping
5. Affine deformation
6. Random crop
7. Coarse Dropout
8. Normalization

At test time:

1. Center cropping
2. Normalization

Optimization parameters For the pooled dataset benchmark, we use the Adam optimizer [KB14] with a learning rate of 5×10^{-4} and a batch size of 64 for 20 epochs.

Hyperparameter Search All the federated learning strategies are tested with the SGD optimizer. We performed a grid search for the federated learning strategies hyperparameters. For FedAvg and Cyclic, we optimized the learning rate over the values {1e-3, 1e-2.5, 1e-2, 1e-1.5, 1e-1, 1e-0.5}. For FedYogi, FedAdam, FedAdagrad and Scaffold, our search grid space was {1e-3, 1e-2.5, 1e-2, 1e-1.5, 1e-1, 1e-0.5} and {1e-3, 1e-2.5, 1e-2, 1e-1.5, 1e-1, 1e-0.5, 1, 1e-0.5, 10} for the learning rate and the server learning rate respectively. For FedProx, our search space contained {1e-3, 1e-2.5, 1e-2, 1e-1.5, 1e-1, 1e-0.5} and {0.001, 0.01, 0.1, 1.} sets for learning rate and μ respectively. The chosen hyperparameters from the HP search can be found in Sec. J.3.

I Fed-Heart-Disease

I.1 Description

The Heart Disease dataset contains records from 920 patients from four hospitals in the USA, Hungary, and Switzerland. There are 13 features before preprocessing: age, sex, chest pain type, resting blood pressure, serum cholesterol, blood sugar, resting electrochardiographic results, maximum heart rate, exercise induced angina, ST depression induced by exercise, slope of the peak ST segment, number of major vessels, and thalassemia background. All features are continuous or binary, except for chest pain type (four categories) and resting electrochardiographic results (three categories). The target is the presence of a heart disease. After preprocessing, we are left with 740 records, each having 13 features. They are split in train and test in a stratified manner. Distribution of the data records among clients is described in Table 9.

Table 9: Distribution of data records among the different clients in Fed-Heart-Disease.

Number	Client	Dataset size	Train	Test
0	Cleveland's Hospital	303	199	104
1	Hungarian Hospital	261	172	89
2	Switzerland Hospital	46	30	16
3	Long Beach Hospital	130	85	45

I.2 License and Ethics

This dataset is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license [JSPD88, DG17].

Regarding privacy, the dataset authors [JSPD88] indicated that sensitive entries of the dataset (including names and social security numbers) were removed from the database.

I.3 Downloading and Preprocessing

Instructions for downloading are available in the corresponding README file on FLamby's repository. Dataset is downloaded from the UCI Machine Learning repository [DG17].

We preprocess the dataset by removing the three features (slope of the peak ST segment, number of major vessels, and thalassemia background) where too many entries are missing. We then remove records where at least one feature is missing. Finally, the two categorical (and non binary) features (chest pain type and resting electrochardiographic results) are encoded as binary features using dummy variables. We also normalize features per center.

I.4 Task

The task consists in predicting the presence of a heart disease so the task is binary classification.

I.5 Baseline, Loss Function, and Evaluation

Loss function For a data record (x_i, y_i), we compute the predicted label $\hat{y}_i = \sigma(\beta^T x_i)$, where $\sigma(z) = 1/(1+\exp(-z))$ is the sigmoid function, and β the parameters of the model. We then compute the loss over the complete dataset as

$$L(y, \hat{y}) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i) \ . \tag{14}$$

Baseline Model We fit a logistic regression model, as this is both a standard problem in medical research, and the strongest baseline according to [DG17]. The model trains in a matter of seconds on modern CPUs.

Evaluation To evaluate the model, we threshold the predicted values $\hat{y}$ at 0.5, and measure the accuracy of the obtained labels as

$$Acc(\beta, X, y) = \frac{|\{i \in [n] \mid y_i = (\hat{y}_i > 0.5)\}|}{n} \ . \tag{15}$$

Optimization parameters For the pooled benchmark, we use the Adam optimizer [KB14] with a learning rate of 0.001, batch size of 4, for 50 epochs.

Hyperparameter Search All the federated learning strategies are tested with the SGD optimizer. We performed a grid search for the federated learning strategies hyperparameters . For FedAvg and Cyclic, we optimized the learning rate over the values {0.1, 0.01, 0.001, 0.0001, 0.00001}. For

Table 10: Hyperparameters used for the FedAvg strategy

FedAvg		
dataset	learning rate	optimizer
Fed-Camelyon16	0.3162	torch.optim.SGD
Fed-LIDC-IDRI	0.001	torch.optim.SGD
Fed-IXI	0.001	torch.optim.SGD
Fed-TCGA-BRCA	0.1	torch.optim.SGD
Fed-KITS19	0.03	torch.optim.SGD
Fed-ISIC2019	0.01	torch.optim.SGD
Fed-Heart-Disease	0.001	torch.optim.SGD

Table 11: Hyperparameters used for the FedProx strategy

FedProx			
dataset	mu	learning rate	optimizer
Fed-Camelyon16	0.316228	0.01	torch.optim.SGD
Fed-LIDC-IDRI	0.01	0.001	torch.optim.SGD
Fed-IXI	0.1	0.001	torch.optim.SGD
Fed-TCGA-BRCA	0.1	0.1	torch.optim.SGD
Fed-KITS19	0.001	0.1	torch.optim.SGD
Fed-ISIC2019	0.001	0.01	torch.optim.SGD
Fed-Heart-Disease	0.001	0.01	torch.optim.SGD

Table 12: Hyperparameters used for the FedAdagrad strategy

FedAdagrad						
dataset	learning rate	optimizer	learning rate server	β_1	β_2	τ
Fed-Camelyon16	0.01	torch.optim.SGD	0.003162			
Fed-LIDC-IDRI	0.1	torch.optim.SGD	0.1			
Fed-IXI	1e-04	torch.optim.SGD	0.1	0.9	0.999	1e-08
Fed-TCGA-BRCA	0.01	torch.optim.SGD	1.0	0.9	0.999	1e-08
Fed-KITS19	0.1	torch.optim.SGD	0.1	0.9	0.999	1e-08
Fed-ISIC2019	0.01	torch.optim.SGD	0.0316			
Fed-Heart-Disease	0.003162	torch.optim.SGD	0.003162	0.9	0.999	0.3162

FedYogi, FedAdam, FedAdagrad and Scaffold, our search grid space was {0.1, 0.01, 0.001, 0.0001, 0.00001} and {10, 1, 0.1, 0.01, 0.001} for the learning rate and the server learning rate respectively. For FedProx, our search space contained {0.001, 0.0001} and {1, 0.1, 0.01} sets for learning rate and μ respectively. The chosen hyperparameters from the HP search can be found in Section J.

J Experimental details

J.1 Computing resources

Most experiments were performed on virtual machines equipped with NVidia P100 GPUs in Google Cloud to tune local baselines as well as searching hyperparameters. Additional experiments were also performed on small workstations for the smallest datasets. Overall, no more than 4k GPU-hours were used throughout the full project.

J.2 FLamby experimental capabilities

FLamby is designed to be a lightweight and simple codebase, to enable ease of use. All clients run sequentially in the same python environment, without multithreading. Datasets are assigned to clients as different python objects. GPU acceleration is supported thanks to current PyTorch [PGM+19] back-

Table 13: Hyperparameters used for the FedAdam strategy

FedAdam						
dataset	learning rate	optimizer	learning rate server	β_1	β_2	τ
Fed-Camelyon16	0.001	torch.optim.SGD	3.1622			
Fed-LIDC-IDRI	0.3162	torch.optim.SGD	0.01			
Fed-IXI	1e-04	torch.optim.SGD	0.1	0.9	0.999	1e-08
Fed-TCGA-BRCA	0.01	torch.optim.SGD	0.1	0.9	0.999	1e-08
Fed-KITS19	0.1	torch.optim.SGD	0.01	0.9	0.999	1e-08
Fed-ISIC2019	0.01	torch.optim.SGD	0.0032			
Fed-Heart-Disease	0.01	torch.optim.SGD	0.01	0.9	0.999	1e-08

Table 14: Hyperparameters used for the FedYogi strategy

FedYogi						
dataset	learning rate	optimizer	learning rate server	β_1	β_2	τ
Fed-Camelyon16	0.003162	torch.optim.SGD	1.0			
Fed-LIDC-IDRI	0.1	torch.optim.SGD	0.001			
Fed-IXI	1e-04	torch.optim.SGD	0.1	0.9	0.999	1e-08
Fed-TCGA-BRCA	0.01	torch.optim.SGD	0.1	0.9	0.999	1e-08
Fed-KITS19	0.1	torch.optim.SGD	0.01	0.9	0.999	1e-08
Fed-ISIC2019	0.01	torch.optim.SGD	0.0032			
Fed-Heart-Disease	0.0031622	torch.optim.SGD	0.01	0.9	0.999	1e-08

Table 15: Hyperparameters used for the Cyclic strategy

Cyclic		
dataset	learning rate	optimizer
Fed-Camelyon16	0.01	torch.optim.SGD
Fed-LIDC-IDRI	0.0316	torch.optim.SGD
Fed-IXI	1e-05	torch.optim.SGD
Fed-TCGA-BRCA	0.01	torch.optim.SGD
Fed-KITS19	0.3	torch.optim.SGD
Fed-ISIC2019	0.0032	torch.optim.SGD
Fed-Heart-Disease	0.01	torch.optim.SGD

Table 16: Hyperparameters used for the Scaffold strategy

Scaffold			
dataset	learning rate	optimizer	learning rate server
Fed-Camelyon16	0.1	torch.optim.SGD	3.1622
Fed-LIDC-IDRI	0.0316	torch.optim.SGD	1.0
Fed-IXI	0.001	torch.optim.SGD	1.0
Fed-TCGA-BRCA	0.01	torch.optim.SGD	1.0
Fed-KITS19	0.1	torch.optim.SGD	1.0
Fed-ISIC2019	0.01	torch.optim.SGD	1.0
Fed-Heart-Disease	0.001	torch.optim.SGD	1.0

end. In order to perform more realistic experiments, e.g. to investigate communication constraints, we encourage the usage of dedicated FL libraries, which are easy to integrate with FLamby.

J.3 Benchmark hyperparameters

We used the hyperparameters detailed in Tables 10 to 16 to obtain the results of Figure 2. These hyperparameters were found after hyper-optimization on a coarse grid. For all strategies and datasets, we set $E = 100$ the number of local updates.

J.4 Run details

Results of Figure 2 were obtained following 5 independent runs with different random seeds, except for the largest one (Fed-LIDC-IDRI), where computational resources prevented training.

K Synthetic dataset splits

One of FLamby's strengths is that it provides datasets with natural splits. However, due to its focus on healthcare applications, the number of clients is limited. Thanks to the standardized API of the datasets, it is possible to create new client splits based on the provided codebase.

We provide an example of such a synthetic sampling based on a Dirichlet distribution on the original clients. If K denotes the previous number of clients and K' the desired number of clients, for $\alpha \in (0, 1)$, we draw a probability distribution $\mathbf{p}_k \in \mathbb{R}^{K'}$ as

$$\mathbf{p}_k \sim \mathrm{Dir}(\alpha), \text{such that} \sum_{k'} p_{kk'} = 1. \tag{16}$$

Each sample from client k is then attributed to client k' with probability $p_{kk'}$, both for the train and test sets. The closer to 0 α gets, the sharper the distribution probability $\mathbf{p}_k$ gets. In order to avoid having empty clients with the synthetic split, we recommend setting $\alpha \geq 1/2$, following previous works [YAG+19].

L Examples of extensions possible in FLamby

In this Appendix, we showcase the extensibility of FLamby by tackling different FL settings.

L.1 Differential Privacy Example

Differential privacy (DP) [DR+14] is an important approach to protect update exchanges between Federated Learning participants against malicious privacy attacks [WLL+20]. In this section, we use Fed-Heart-Disease to demonstrate the use of FLamby to study (ϵ, δ)-DP federated learning [DR+14].

Figure 4 displays the average performance of a machine learning model trained in a differentially private fashion with DP-FedAvg as a function of ϵ and δ. We compare it to a regular training using the same model initialization but without privacy ("Baseline wo DP"), trained with regular FedAvg. We see the performance diminishing when ϵ tends to 0, especially for small values of δ often used in practice, which is a standard phenomenon.

To implement DP in FedAvg, we use on the DP-SGD mechanism and track the monitoring of privacy budget thanks to the moment accountant [ACG+16]. We use the Opacus library [YSS+21], which is easy to integrate into FLamby thanks to its modular design.

At this time of writing, Opacus does not support all Deep Learning building blocks such as normalization layers. This prevents applying DP mechanisms on some some of FLamby's baseline models such as the baseline for Fed-ISIC2019 and Fed-IXI.

L.2 Personalized Federated Learning

Model personalization [FMO20] is an effective strategy to improve model performance in cross-silo settings, especially in presence of data heterogeneity. Here, we showcase a simple example of model personalization with FLamby, which is possible thanks to its simple and modular API.

We implement the FedAvg strategy followed by local fine tuning on each center, thus producing as many models as there are clients. We test the addition of such fine-tuning process on the performances

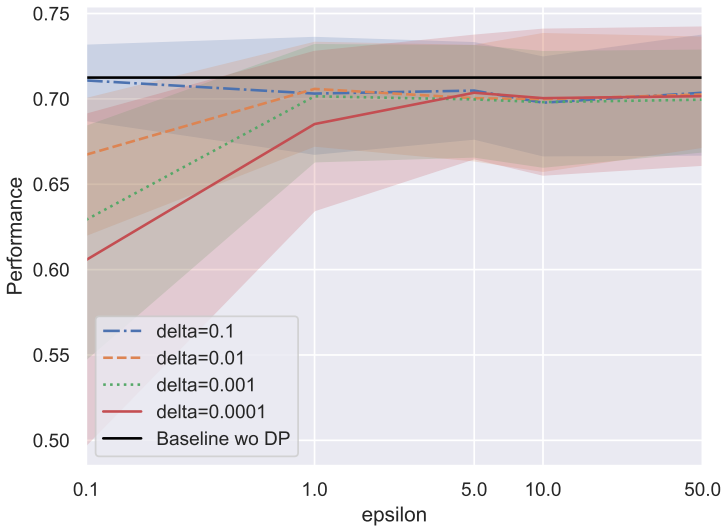


Figure 4: Impact of Differential-Privacy on average performance for DP-FedAvg on Fed-Heart-Disease.

of Federated models while testing each model on its corresponding test set. For each dataset, we perform 100 local updates after the federated averaging training has taken place.

Figure display the training results 5. We see that for Fed-Heart-Disease and Fed-ISIC2019, personalization improves results, while performance is slightly degraded for Fed-Camelyon16. We hope that researchers will be able to investigate more personalization strategies easily with FLamby.

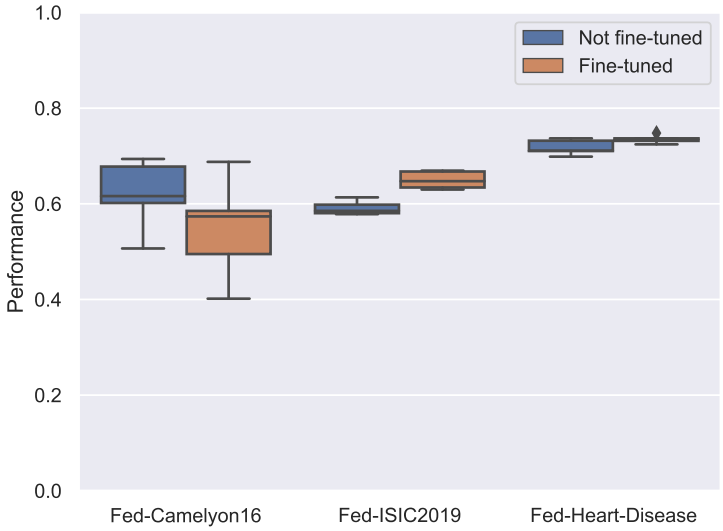


Figure 5: Impact of personalization on test average performance on three datasets of the suite (Fed-Heart-Disease, Fed-Camelyon16 and Fed-ISIC2019) after performing Federated Averaging. In the two extreme cases (Left and Right) fine-tuning is beneficial, whereas on Fed-Camelyon16, Fine-tuning degrades the performance of the resulting models. We hypothesize that, in this case, fine-tuning overfits the local training datasets.

M Quantitative heterogeneity benchmarks

We describe in this section our analysis of the heterogeneity of the Flamby datasets. Due to the variety of tasks and data, we restrict this study to generic metrics. We always compute them on the whole dataset, putting together test and train data to have the best possible estimation of the underlying distribution.

In the first subsection, we briefly describe the three sources of heterogeneity that we consider. Next, we detail the methodology used to compute statistical distance between clients. Thirdly, we apply this methodology to the FLamby's dataset. And finally, we provide some discussion on these results.

M.1 Description of measured heterogeneity

Imbalance. The easiest quantification of heterogeneity comes from the number of samples hosted by each client, which gives natural unbalance in the training because small clients are likely to be either over-fitted or neglected in the final model.

Labels distribution. Labels can be another source of heterogeneity in case when prediction outputs vary between clients for the considered task. For instance, if a client is specialized in treating the patients with the given disease, the labels are likely to be biased (with a high number of persons with this disease), even if all clients have patients from a similar population. In the case of the toy example of MNIST, a split with this heterogeneity is to have clients specialized on a single digit.

Features distribution. Finally, in case when the features are the origin of heterogeneity, the underlying sample distribution is different in each client. It means that the same outputs can be characterized by different data depending on the client. This heterogeneity can arise from having different measurement tools (as is the case in Fed-LIDC-IDRI dataset), or different population in each client (e.g. Fed-TCGA-BRCA dataset). In the case of the toy example of MNIST, a split with this heterogeneity is to have a client where digits are in italics.

M.2 Methodology

For the sample division, we report the number of clients and the splits. As a way to summarize heterogeneity, we compute the entropy of the distribution of the samples across clients.

$$H(X) = -\sum_{k=1}^{K} \frac{n_k}{N} \log_2 \frac{n_k}{N} \tag{17}$$

where K is the number of clients, N the total number of samples and n_k the number of samples belonging to client k.

For label and features heterogeneity, when initial dimension is larger than 16, we reduce the dimension by using PCA trained on all the centralized samples. Then, for each client, we compute the Wasserstein distance (see Definition 1) between each client's distribution, or the total variation distance (see Definition 2) for discrete data. The Wasserstein distances are computed using a minibatch-Wasserstein (without regularization) [see FZF+20] implemented in the POT library [FCG+21] and is defined below:

Definition 1 (Wasserstein distance) *For all probability measures α and β on $\mathcal{B}(\mathbb{R}^d)$, such that $\int_{\mathbb{R}^d} \|w\|^2 \mathrm{d}\alpha(w) < +\infty$ and $\int_{\mathbb{R}^d} \|w\|^2 \mathrm{d}\beta(w) \leq +\infty$, define the squared Wasserstein distance of order 2 between α and β by*

$$\mathcal{W}_2^2(\alpha, \beta) := \inf_{\xi \in \Gamma(\alpha,\beta)} \int \|x - y\|^2 \xi(dx, dy), \tag{18}$$

where $\Gamma(\alpha, \beta)$ is the set of probability measures ξ on $\mathcal{B}(\mathbb{R}^d \times \mathbb{R}^d)$ satisfying for all $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$, $\xi(\mathsf{A} \times \mathbb{R}^d) = \beta(\mathsf{A})$, $\xi(\mathbb{R}^d \times \mathsf{A}) = \alpha(\mathsf{A})$.

The Total-Variation distance used for discrete labels is defined as following:

Table 17: Mean and max distances for features and labels, and entropy computed using eq. 17. High values correspond to important heterogeneity.

	camelyon16	ixi	tcga brca	kits19	isic2019	heart disease
X mean	710618.57	188.52	6.38	-0.31	1.62	7.05
X max	710618.57	289.14	16.74	1.42	7.29	10.59
Y mean	-0.01	3.52	22.26	0.12	29.40	2.90
Y max	-0.01	6.95	79.42	2.68	53.87	5.77
Entropy	0.97	1.38	2.44	2.49	1.93	1.75

Definition 2 (Total-Variation) *For any vector of probability α, β in $[0,1]^d$, the* TV*-value is defined by*

$$\mathrm{TV}(\alpha, \beta) = \frac{1}{2} \sum_{i=1}^{d} |\alpha_i - \beta_i| \in [0, 1].$$

As the tasks, dimension and characteristics differs for each dataset, there are two directions to interpret results: 1) comparing heterogeneity between clients, and 2) by contrast with a synthetic scenario where data would be identically distributed among clients. Thus, we compute two pairwise-distances matrices: one with the natural split, and one with data distributed uniformly on clients (the size of the dataset on each client is identical to the natural split). The latter is built to simulate the i.i.d. setting and to compare the natural split with the case where we would have had homogeneous clients.

Next, we rescale the pairwise-distances matrices in order to have standardized variables in the synthetic case. Formally, we note $\mathcal{D}_{\text{i.i.d.}}$ (resp. $\mathcal{D}_{\text{natural}}$) the set of distances for the synthetic i.i.d. split (resp. natural split). The cardinal of these two sets is $n(n-1)/2$ because 1) the diagonal (distance of a client with itself) must be zero, 2) the Wasserstein distance is symmetric. Rescaling the matrices means that we standardize the i.i.d. set by removing the mean and scaling to unit variance i.e. computing $(\mathcal{D}_{\text{i.i.d.}} - \mu_{\text{i.i.d.}})/\sigma_{\text{i.i.d.}}$, where $\mu_{\text{i.i.d.}}, \sigma^2_{\text{i.i.d.}}$ are the mean and the variance of the i.i.d. set. Then, we apply the same transformation on the set of distance computed on the natural split i.e. $(\mathcal{D}_{\text{natural}} - \mu_{\text{i.i.d.}})/\sigma_{\text{i.i.d.}}$.

This is motivated by the fact that in the homogeneous case, we expect the distances to be zero. It follows that after rescaling, we are able to compare the values within the pairwise-distance matrix of the natural split. Thus, we are able to identify which clients are the closest or at odds with the others. Additionally, the magnitudes after rescaling give an indication on the degree of heterogeneity within the dataset. The bigger their magnitudes are after rescaling, the more distant is the natural split to what would be a homogeneous split.

M.3 Datasets analysis

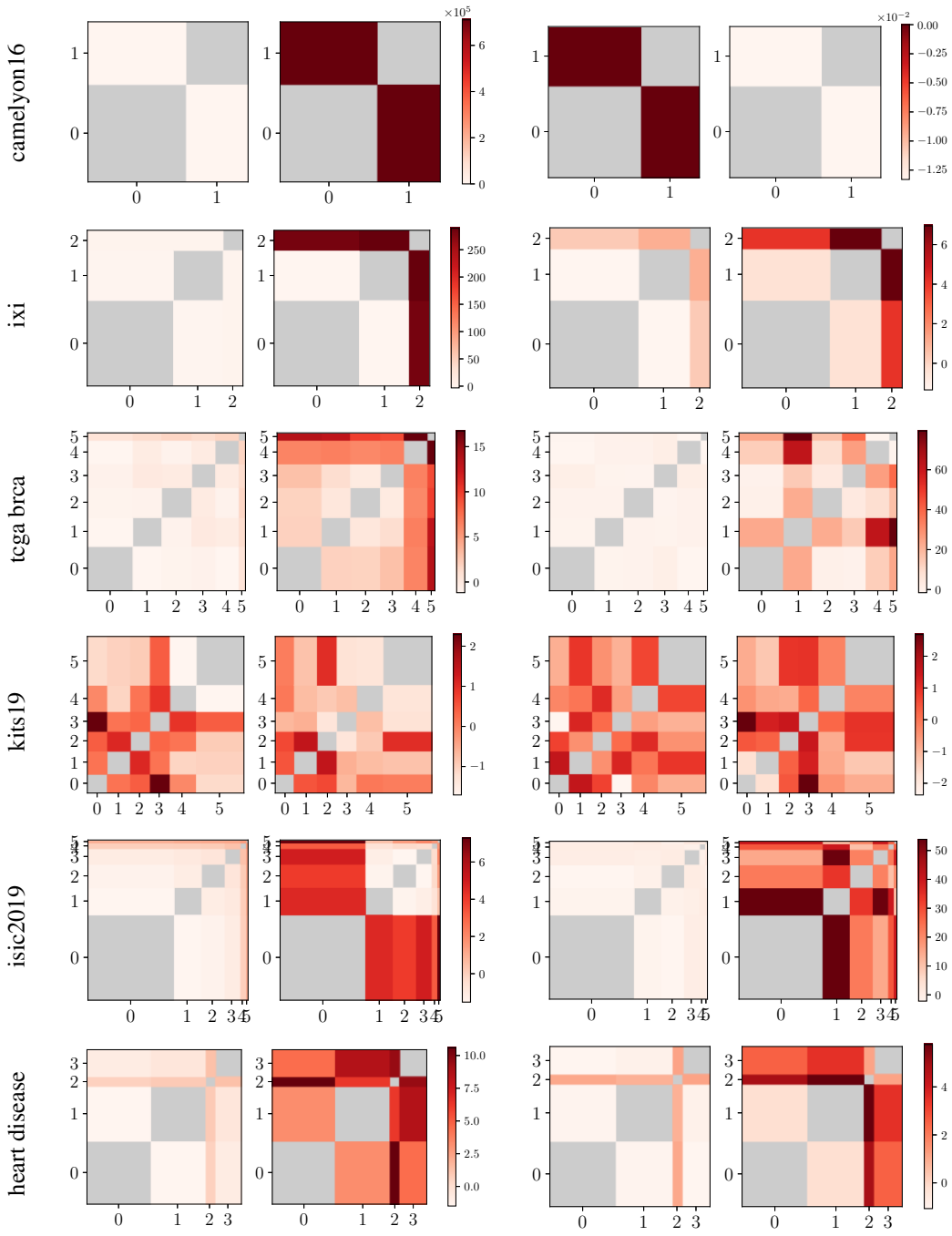
For each dataset, we report the rescaled pairwise-distances matrix for features and for labels, with the associated i.i.d. baseline in Figure 18. In order to highlight on the same plot the clients size imbalance and the clients' heterogeneity, the width of each column in the pairwise distance matrix is proportional to the client size. This helps to put heterogeneity in perspective with the size of the client and to have a better understanding of what is, in practice, the weight of the client's heterogeneity.

We also report the mean and maximum value for each dataset, both for features and labels, alongside sample distribution among client entropy in Table 17.

We can make the following observations for each dataset by analysing Tables 17 and 18.

- **Camelyon16.** There is a high heterogeneity on features, however labels are i.i.d.. This is of particular interest as it means that different features have led to close labels.
- **IXI.** Client 0 and 1 are very close, both in terms of features and labels. Compared to them, client 3 (which is also the smallest) is an outsider.
- **TCGA-BRCA.** In terms of features, clients 0, 1, 2, 3 are relatively close. But client 1 has labels different from the three other clients. Client 4 and 5 are apart, but it is interesting to

Table 18: Heterogeneity of Flamby datasets. Each matrix is the pairwise distance matrix, the width of their column corresponds to the number of sample.



notice that while their features are extremely different, their labels are almost identical. As for Camelyon16, this is of particular interest.

- **Kits19.** Clients are completely homogeneous for both features and labels. This can be derived from the fact that the distances are the same for both the natural split and the i.i.d. split. This was already suggested by Figure 1d.
- **Isic2019.** Features of client 0 are very different from the other clients. The four last clients have close features because their distances are almost zero. On the contrary, their distances on labels are far from zero. It means that close features have led to different labels. This is an element of particular interest.
- **Heart disease.** The heterogeneity on features and on labels are of the same order of magnitude (up to a factor 2) and not very important (maximum is at 10 for features, at 5 for labels). Client 2 is the smallest client and is an outsider. This is logical knowing that client 2 is a hospital specialized in major heart disease. We can also notice that client 1 and 2 have a moderate distance in terms of features. But however, based on their labels, they have the most significant distance. It means that relatively close features have led to very different labels. Like for isic2019, this is an element of particular interest. This could have happened if patients in hospital 2 have more severe heart disease than in other hospitals, but still have disease features close to classical cases.

M.4 Discussion

Measuring heterogeneity is an open question in machine learning, and it is beyond the scope of this paper. We provide some measurements as an indicative benchmark, with a methodology easy to reproduce. Other kind of heterogeneity could be computed and might lead to different conclusion on which clients are less or more similar, in particular as PCA representation can lead to a significant data loss. The advantages of this benchmark is its generality that allows to tackle the various data type and tasks found in FLamby.

References

[ACG+16] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[AIMB+11] Samuel G Armato III, Geoffrey McLennan, Luc Bidaut, Michael F McNitt-Gray, Charles R Meyer, Anthony P Reeves, Binsheng Zhao, Denise R Aberle, Claudia I Henschke, Eric A Hoffman, et al. The lung image database consortium (LIDC) and image database resource initiative (IDRI): a completed reference database of lung nodules on ct scans. *Medical physics*, 38(2):915–931, 2011.

[AMM+20] Mathieu Andreux, Andre Manoel, Romuald Menuet, Charlie Saillard, and Chloé Simpson. Federated survival analysis with discrete-time Cox models. *arXiv preprint arXiv:2006.08997*, 2020.

[Aro] Aman Arora. Siim-isic melanoma classification - my journey to a top 5% solution and first silver medal on kaggle. `https://amaarora.github.io/2020/08/23/siimisic.html`. Accessed: 2022-02-02.

[Bra97] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[CCR+19] Marc Combalia, Noel CF Codella, Veronica Rotemberg, Brian Helba, Veronica Vilaplana, Ofer Reiter, Cristina Carrera, Alicia Barreiro, Allan C Halpern, Susana Puig, et al. Bcn20000: Dermoscopic lesions in the wild. *arXiv preprint arXiv:1908.02288*, 2019.

[CGC+18] Noel CF Codella, David Gutman, M Emre Celebi, Brian Helba, Michael A Marchetti, Stephen W Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, et al. Skin lesion analysis toward melanoma detection: A challenge at the 2017

international symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC). In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 168–172. IEEE, 2018.

[Cox72] David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[CVS+13] Kenneth Clark, Bruce Vendt, Kirk Smith, John Freymann, Justin Kirby, Paul Koppel, Stephen Moore, Stanley Phillips, David Maffitt, Michael Pringle, et al. The cancer imaging archive (TCIA): maintaining and operating a public information repository. *Journal of digital imaging*, 26(6):1045–1057, 2013.

[DCM+20] Olivier Dehaene, Axel Camara, Olivier Moindrot, Axel de Lavergne, and Pierre Courtiol. Self-supervision closes the gap between weak and strong supervision in histology. *arXiv preprint arXiv:2012.03583*, 2020.

[DG17] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[Dic45] Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[DR+14] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[dt] Brain development team. Ixi dataset. `https://brain-development.org/ixi-dataset/`. Accessed: 2022-02-02.

[FCG+21] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. POT: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

[FDZ+15] Christopher P Favazza, Xinhui Duan, Yi Zhang, Lifeng Yu, Shuai Leng, James M Kofler, Michael R Bruesewitz, and Cynthia H McCollough. A cross-platform survey of ct image quality and dose from routine abdomen protocols and a method to systematically standardize image quality. *Physics in Medicine & Biology*, 60(21):8381, 2015.

[Fee10] Timothy G Feeman. *The mathematics of medical imaging*. Springer, 2010.

[FMO20] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

[FZF+20] Kilian Fatras, Younes Zine, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. Learning with minibatch Wasserstein : asymptotic and gradient properties. In *AISTATS 2020 - 23nd International Conference on Artificial Intelligence and Statistics*, volume volume 108 of *PMLR*, pages 1–20, Palermo, Italy, June 2020.

[GNS+19] Nils Gessert, Maximilian Nielsen, Mohsin Shaikh, René Werner, and Alexander Schlaefer. Skin lesion classification using ensembles of multi-resolution efficientnets with meta data. *CoRR*, abs/1910.03910, 2019.

[Goo92] Irving John Good. Rational decisions. In *Breakthroughs in statistics*, pages 365–377. Springer, 1992.

[HIMH+20] Nicholas Heller, Fabian Isensee, Klaus H Maier-Hein, Xiaoshuai Hou, Chunmei Xie, Fengyi Li, Yang Nan, Guangrui Mu, Zhiyong Lin, Miofei Han, et al. The state of the art in kidney and kidney tumor segmentation in contrast-enhanced ct imaging: Results of the kits19 challenge. *Medical Image Analysis*, page 101821, 2020.

[HSK+19] Nicholas Heller, Niranjan Sathianathen, Arveen Kalapara, Edward Walczak, Keenan Moore, Heather Kaluzniak, Joel Rosenberg, Paul Blake, Zachary Rengel, Makinna Oestreich, et al. The kits19 challenge data: 300 kidney tumor cases with clinical context, ct semantic segmentations, and surgical outcomes. *arXiv preprint arXiv:1904.00445*, 2019.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[IJK+21] Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211, 2021.

[ILTT11] Juan Eugenio Iglesias, Cheng-Yi Liu, Paul M Thompson, and Zhuowen Tu. Robust brain extraction across datasets and comparison with publicly available methods. *IEEE transactions on medical imaging*, 30(9):1617–1634, 2011.

[IMB+15] S. G. Armato III, G. McLennan, L. Bidaut, M. F. McNitt-Gray, C. R. Meyer, A. P. Reeves, B. Zhao, D. R. Aberle, C. I. Henschke, E. A. Hoffman, E. A. Kazerooni, H. MacMahon, E. J. R. Van Beek, D. Yankelevitz, A. M. Biancardi, P. H. Bland, M. S. Brown, R. M. Engelmann, G. E. Laderach, D. Max, R. C. Pais, D. P. Y. Qing, R. Y. Roberts, A. R. Smith, A. Starkey, P. Batra, P. Caligiuri, A. Farooqi, G. W. Gladish, C. M. Jude, R. F. Munden, I. Petkovska, L. E. Quint, L. H. Schwartz, B. Sundaram, L. E. Dodd, C. Fenimore, D. Gur, N. Petrick, J. Freymann, J. Kirby, B. Hughes, A. V. Casteele, S. Gupte ans M. Sallam, M. D. Heath, M. H. Kuhn, E. Dharaiya, R. Burns, D. S. Fryd, M. Salganicoff, V. Anand, U. Shreter, S. Vastagh, B. Y. Croft, and L. P. Clarke. Data from lidc-idri [data set]. the cancer imaging archive., 2015.

[ITW] Maximilian Ilse, Jakub M. Tomczak, and Max Welling. Attention-based deep multiple instance learning. `https://github.com/AMLab-Amsterdam/AttentionDeepMIL`. Accessed: 2022-02-02.

[ixi] Ixitiny dataset. `https://torchio.readthedocs.io/datasets.html#torchio.datasets.ixi.IXITiny`. Accessed: 2022-05-18.

[Jad20] Shruti Jadon. A survey of loss functions for semantic segmentation. In *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–7. IEEE, 2020.

[Jen05] Stephen P Jenkins. Survival analysis. *Unpublished manuscript, Institute for Social and Economic Research, University of Essex, Colchester, UK*, 42:54–56, 2005.

[JSPD88] Andras Janosi, William Steinbrunn, Matthias Pfisterer, and Robert Detrano. Heart disease data set, 1988.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[LBEB+18] Geert Litjens, Peter Bandi, Babak Ehteshami Bejnordi, Oscar Geessink, Maschenka Balkenhol, Peter Bult, Altuna Halilovic, Meyke Hermsen, Rob van de Loo, Rob Vogels, et al. 1399 H&E-stained sentinel lymph node sections of breast cancer patients: the CAMELYON dataset. *GigaScience*, 7(6):giy065, 2018.

[LGG+17] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017.

[LLH+18] Jianfang Liu, Tara Lichtenberg, Katherine A Hoadley, Laila M Poisson, Alexander J Lazar, Andrew D Cherniack, Albert J Kovatich, Christopher C Benz, Douglas A Levine, Adrian V Lee, et al. An integrated tcga pan-cancer clinical data resource to drive high-quality survival outcome analytics. *Cell*, 173(2):400–416, 2018.

[MAB] Alessia Marcolini, Ernesto Arbitrio, and Nicole Bussola. https://histolab.readthedocs.io/en/latest/. `https://histolab.readthedocs.io/en/latest/`. Accessed: 2022-05-18.

[MLI+14] Matthew McCormick, Xiaoxiao Liu, Luis Ibanez, Julien Jomier, and Charles Marion. Itk: enabling reproducible research and open science. *Frontiers in Neuroinformatics*, 8, 2014.

[MNA16] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016.

[Mod] Marc Modat. Nifty reg. `https://sourceforge.net/p/niftyreg/git/ci/master/tree/`. Accessed: 2022-02-02.

[Net] TCGA Research Network. Tensorflow federated stack overflow dataset. `https://www.cancer.gov/tcga`. Accessed: 2022-05-18.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[TL19] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019.

[TRK18] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018.

[VDWCV11] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in science & engineering*, 13(2):22–30, 2011.

[WLL+20] Wenqi Wei, Ling Liu, Margaret Loper, Ka-Ho Chow, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu. A framework for evaluating gradient leakage attacks in federated learning. *arXiv preprint arXiv:2004.10397*, 2020.

[YAG+19] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pages 7252–7261. PMLR, 2019.

[YSS+21] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, et al. Opacus: User-friendly differential privacy library in pytorch. *arXiv preprint arXiv:2109.12298*, 2021.