ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE
SCHOOL OF LIFE SCIENCES

Master project in Life Sciences Engineering

inDISCO: INterpretable DIStributed COllaborative learning for biomedical images

Done by

Klavdiia Naumova

Under the direction of
Prof. Martin Jaggi
in the Machine Learning and Optimization laboratory

EPFL

External Expert Dr. Sai Praneeth Karimireddy

LAUSANNE, EPFL 2023

Summary

Background. Deep learning applied to medical data has the potential to revolutionize healthcare. However, concerns over data privacy limit the power and representation of the models. A promising solution is **DIS**tributed **CO**llaborative (**DISCO**) learning which allows several data owners (clients) to learn a joint model without sharing data. This black-box data approach is a major limitation to interpretability and may conceal bias or interoperability mismatches that could compromise model performance.

Aim. This project adapts a well-known interpretable prototypical part learning network (`ProtoPNet`) to a federated DISCO setting to allow clients to directly visualize the differences in the features learned from each client (without sharing them) providing privacy-preserving and interpretable learning on images.

Methods/Findings. `ProtoPNet` was adjusted for a federated setting and trained for four clients using two benchmark datasets: images of bird species and human chest X-rays. The global models reach 81.25 and 74.53 % accuracy on birds and X-rays unbiased datasets, respectively. We visualize and compare the prototypes learned locally and globally. A simple systematic bias (an emoji in images of one class) injected into one client's data strongly affects both local and global prototypes. The models trained in the presence of this bias give 100.0 % accuracy for a biased class when evaluated on the biased test set and 0.0 (birds) or 50.0 (X-rays) % when evaluated on an unbiased set. For the X-rays, we also experimented with a real-world bias: the presence of a chest drain in the images of a pleural effusion class. This setting has a similar effect on learned prototypes and results in 77.31 and 66.15 % accuracy for local and global models evaluated on a biased test set.

Conclusion. Our interpretable DISCO (`inDISCO`) approach allows clients to detect systematic bias in their data in an interpretable and privacy-preserving way and thus has a potential for usage in the privacy-sensitive medical imaging domain.

Introduction

The potential impact of deep learning on clinical decision-making systems is being increasingly documented. Numerous advances in this area suggest promising applications of deep learning in such fields as medical image recognition, biosignal processing, genomics, drug discovery, and more [1,2]. Deep learning and, more recently, foundation algorithms such as large language models can not only automate routine analysis of medical records [3] but help find hidden predictive patterns in the data that may reduce errors and unnecessary interventions (for example biopsy) [4]. This moves us towards more efficient, personalized, and accessible healthcare.

However, learning from medical data usually necessitates sharing it with a server (centralized training), which is often not possible due to privacy restrictions. At the same time, limiting training to fragments of the data (local training) would dilute the generalizability. To address this issue, McMahan et al. [5] suggested a technique called distributed collaborative learning or federated learning (FL) which allows privacy-preserving and secure training on data distributed across several clients.

Though FL seemingly resolves the issue of private data sharing, its privacy comes at a cost to data transparency, where clients learn blindly from their peers. In real-world applications, this *black-box data* is often biased, poorly interoperable, and/or non-identically and independently distributed (non-IID). This issue is particularly important for imaging data since the complexity of the deep learning architectures for this modality has earned them the term *black-box models*.

Numerous approaches try to *explain* black-box models by a posthoc analysis of the predictions made by convolutional neural networks (CNN) [6–12]. Many of these methods are well presented in the popular book "Interpretable Machine Learning" [1] written by Christoph Molnar. The book describes the most popular techniques to be LIME [6], SHAP [7], Feature visualization [2], and saliency mapping with Grad-CAM [8]. However, these methods fail to interpret how and why a model makes its decision [13].

In opposition to black-box models, there exist inherently interpretable models, i.e. where the decision-making component is transparent-by-design [14]. A popular example of such a model is a prototypical part neural network (ProtoPNet) developed by Chen et al. [15]. We summarize their implementation in Fig 1. It uses a CNN to create a set of patches in the latent space and learn a prototypical patch from this set (i.e. a vector representing a class that does not equal any original patch). Classification then relies on a similarity score computed between these learned prototypes and a latent representation of a test image. A prototype can be visualized by highlighting the patch most activated by this prototype. The performance of ProtoPNet was demonstrated on the task of bird species identification. The model showed an accuracy level comparable with the state-of-the-art black-box deep neural networks while being easily interpretable. This network is described in detail in Section "Materials and Methods". ProtoPNet was further extended to perform classification of mass lesions in digital mammography [16] and image recognition with hierarchical prototypes [17].

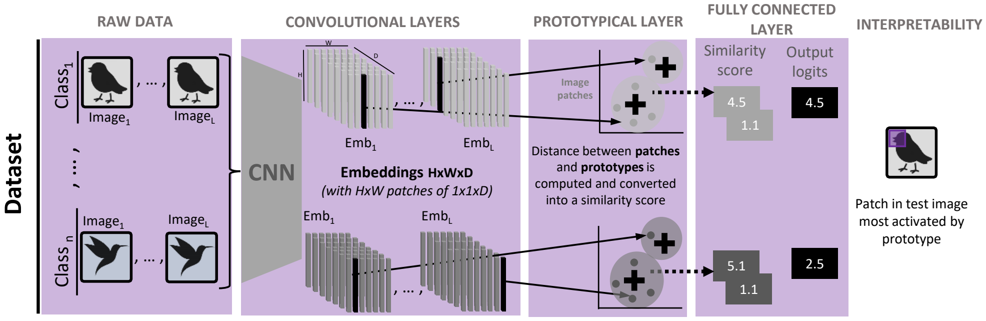


Figure 1. ProtoPNet architecture. This is a centralized setting with no clients. ProtoPNet passes raw images through a CNN to create embeddings of size $[H \times W \times D]$ in the latent space $(Emb_1, ..., Emb_L)$, which are divided up into $H \times W$ image patches $[1 \times 1 \times D]$. These image patches (**black**) are clustered around the closest prototypes $(+)$ which are being learned for each image. Importantly the prototype does not equal to the original patch but is a vector representing a particular class. Classification is based on a similarity score computed between the prototypes and the image patches. In the final panel, we see that the patch most activated by the prototypes can be visualized directly.

To address poorly interoperable and non-IID data in the FL setting, researchers developed a set of optimization-based methods [18–21] and personalization techniques. The latter is presented in the works of [22] (FedPer) and [23] (iFedAvg) who added local personalization layers to the shared model. In particular, the iFedAvg technique implements a composition of a shared neural network f_{shared} with local element-wise affine layers f_{in} and f_{out} that allows learning the necessary shift from the client's data to the global model and making the values in f_{in} and f_{out} easily interpretable. After training, learned shifts can be visualized to identify clients with incompatible data samples. So far, iFedAvg was applied to a tabular dataset (2014 - 2016 West African

[1] https://christophm.github.io/interpretable-ml-book/

[2] https://distill.pub/2017/feature-visualization/

Ebola epidemic) since this type of data has clear and interpretable features. In the case of images, however, one needs to either ”featurize” the input or use interpretable neural networks in combination with iFedAvg.

In this work, we adapt ProtoPNet to FL and apply it to medical imaging data. As summarized in Fig 2, clients learn their local prototypes as well as global ones in communication with each other. The patches most activated by each of these prototypes can be visualized and compared on each client's local test sets. By comparing global and local prototypes the clients can assess the interoperability of the data. Thus, we can introduce interpretability to FL and directly examine the predictive impact of other clients' data without compromising clients' privacy.

Our main contributions are as follows:

1. We formalize a set of use cases for interpretable distributed learning on imperfectly interoperable biomedical image data sets containing hidden bias.

2. We introduce inDISCO adapting ProtoPNet to FL and compare its performance to baseline models.

3. We demonstrate how inDISCO helps to identify a biased client in FL without disclosing the data.

4. Finally, we propose a new approach to use inDISCO for interpretable personalization.

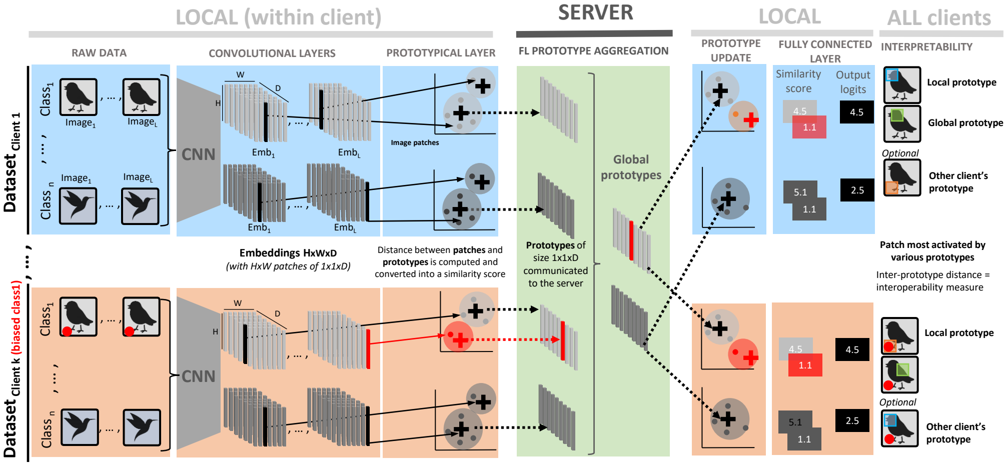


Figure 2. **inDISCO architecture**. Several clients ($client_1$,..., $client_k$) wish to learn a model in a federated setting via a SERVER. inDISCO, passes raw images through a CNN to create embeddings of $[H \times W \times D]$ in the latent space $(Emb_1, ..., Emb_L)$, which are divided up into $H \times W$ image patches $[1 \times 1 \times D]$. These image patches (**black**) are clustered around the closest prototypes (+) which are being learned for each image. Importantly, the prototype is not an original patch of the image, but a vector representing a class. $Class_1$ of $Client_k$ has **systematic bias** which contaminates the prototype pool (+). Prototypes for each class are shared to the SERVER by each client and aggregated to make global prototypes. These are then pushed back to the clients. Classification is based on a similarity score computed between the prototypes and the image patches. In the final panel, we see how global and local prototypes can be compared to directly visualize shifts without sharing any original data.

Materials and methods

Model architecture and problem formulation

The ProtoPNet architecture is presented in Fig. 1 The network is composed of the following parts:

- a set of convolutional layers to extract and learn features;
- two additional 1×1 convolution layers with D channels and the ReLU activation after the first layer and Sigmoid after the second one;
- a prototype layer with a predefined number of prototypes. Each prototype is a vector of size $1 \times 1 \times D$ with randomly initialized entries;
- a final fully connected layer with the number of input nodes equal to the number of prototypes and the number of output nodes corresponding to the number of classes. The weights indicate the importance of a particular prototype for a class. They are initialized as in [15] such that the connection between the prototypes and their corresponding class is 1 and -0.5 for the connections with the wrong classes.

We trained inDISCO, an FL adaptation of ProtoPNet, using either **IID** (unbiased, identically distributed classes) or **IIO** (imperfectly interoperable with systematic bias in a single class) data distribution among clients. A central server aggregates and updates the models' prototypes and weights of the final fully connected layer. Convolutional weights in our setting always stay local.

By learning local prototypes, each client identifies the parts of its training images most important for the task. In contrast, the global prototypes show the relevance for all clients on average. Finally, by examining the difference between local and global prototypes, a client can identify and quantify the presence of bias in its own or another client's dataset.

Notation. Let us adapt the notation from [15]. Given input $\mathbf{x}_n$, where $n \in \{1, ..., N\}$, each of N clients learns features with convolutional layers $f(\mathbf{x}_n)$ and m prototypes $\mathbf{P}_n = \{\mathbf{p}_{nj}\}_{j=1}^{m}$. Given a convolutional output $\mathbf{z}_n = f(\mathbf{x}_n)$, the j-th prototype of the n-th client's unit $g_{\mathbf{p}_j,n}$ in the prototype layer $g_{\mathbf{p}n}$ computes the squared L^2 distance between the prototype $\mathbf{p}_{nj}$ and all the patches of $\mathbf{z}_n$ and converts these distances into similarity scores. These scores are then multiplied by the weight matrix w_{hn} in the final fully connected layer h_n followed by softmax normalization to output class probabilities.

Local training. Given a set of training images $\mathbf{D}_n = \{(\mathbf{x}_{ni}, y_{ni})\}_{i=1}^{k}$, where k is a number of images per client, each client aims to minimize the following objective:

$$\min_{\mathbf{P}_n, w_{conv,n}} \frac{1}{k} \sum_{i=1}^{k} \text{CrsEnt}_n(h_n \circ g_{\mathbf{p}n} \circ f(x_{ni}), y_{ni}) + \lambda_1 \text{Clst}_n + \lambda_2 \text{Sep}_n, \quad (1)$$

where $w_{conv,n}$ denotes the weights of the convolutional layers learned by client n, CrsEnt is a cross-entropy loss that penalizes the misclassification, and the cluster and separation costs are defined as follows:

$$\text{Clst}_n = \frac{1}{k} \sum_{i=1}^{k} \min_{j:\mathbf{p}_{nj} \in \mathbf{P}_{ny_{ni}}} \min_{\mathbf{z}_n \in \text{patches}(f(x_{ni}))} ||\mathbf{z}_n - \mathbf{p}_{nj}||_2^2 \quad (2)$$

$$\text{Sep}_n = -\frac{1}{k} \sum_{i=1}^{k} \min_{j:\mathbf{p}_{nj} \notin \mathbf{P}_{ny_{ni}}} \min_{\mathbf{z}_n \in \text{patches}(f(x_{ni}))} ||\mathbf{z}_n - \mathbf{p}_{nj}||_2^2 \quad (3)$$

The minimization of the cluster cost (Clst) is needed to make each training image have a latent patch that is close to at least one prototype of the correct class. At the same time, every latent patch of a training image is separated from the prototypes of the incorrect class through the minimization of the separation cost (Sep). More detail `ProtoPNet` is found in [15] and summarized in Fig. 1.

Federated update. At the global update step, the server performs simple averaging of all the local prototypes $\mathbf{P}_{loc} = \{\mathbf{P}_n\}_{n=1}^{N}$ and weights of the final layer $\mathbf{W}_{h,loc} = \{w_{hn}\}_{n=1}^{N}$ to obtain the global parameters:

$$\mathbf{P}_{glob} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{P}_n \quad (4) \qquad \mathbf{W}_{h,glob} = \frac{1}{N}\sum_{n=1}^{N} w_{hn} \quad (5)$$

and then sends them back to clients as shown in Fig. 2.

To visualize the prototypes, each client finds for each of the local and global prototypes a patch among its training images from the same class that is mostly activated by the prototype. It is achieved by forwarding the image through the trained `ProtoPNet` and upsampling the activation matrix (the matrix of similarity scores obtained before global max pooling) to the size of the input image. A prototype can be described as the smallest rectangular area within an input image that contains pixels with an activation value in the upsampled activation map equal to or greater than the 99th percentile of all activation values in that map. [15].

Data

Birds dataset

Our experiments were conducted on CUB-200-2011 dataset [24] of 200 bird species from which we took the first 20 classes. Preprocessing was performed as described by [15]. We introduced class-specific bias for one client by adding an emoji to the images of a particular class at a specific location (SI, Fig. 9).

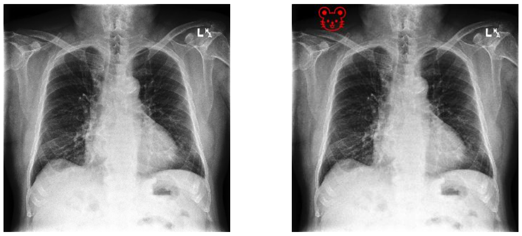
CheXpert

The main experiments presented in this work were done using CheXpert dataset [25], a large public dataset of 224,316 chest X-rays of 65,240 patients collected from Stanford Hospital and labeled by radiologists. Each image was labeled for the presence of 14 observations as positive, negative, or uncertain. To simplify the experiments and results interpretation, we decided to stick with *one-vs-rest* setting using images with positive labels for classes Cardiomegaly or Pleural effusion as a positive class and all other images as a negative class. Cardiomegaly is a health condition characterized by an enlarged heart, and pleural effusion is an accumulation of fluid between the visceral and parietal pleural membranes that line the lungs and chest cavity. This setting, however, resulted in a large data imbalance (7 and 1.6 times for cardiomegaly and pleural effusion, respectively). To address this issue, we decreased the size of a negative class in the training set by undersampling to make it equal to the size of a positive class. The final training sets had 48,600 and 37,088 images for cardiomegaly and pleural effusion classification, respectively. The test sets were left imbalanced.

In the case of medical data, we used two ways of creating IIO dataset for one of the clients:

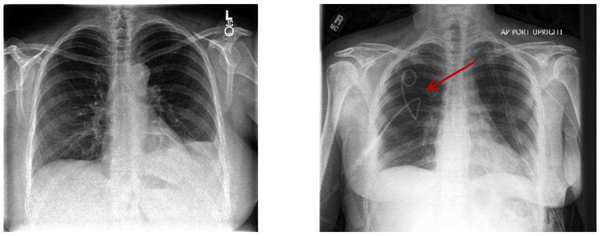
- adding a simple emoji to a positive class (Fig. 3);
- adding chest drains to a positive class as a more real-world bias (Fig. 4). To achieve this, we replaced images in a class Pleural effusion with X-rays labeled for the presence of chest drains [26].

The real-world use case can arise as pleural effusions are often drained. Drain positions are routinely checked with a post-insertion X-ray. Thus, a model may learn to diagnose pleural effusion by detecting a chest drain, rather than the pathology.



(a) Unbiased **(b)** Biased

Figure 3. Examples of unbiased and biased (imperfectly interoperable) images from CheXpert dataset for class Cardiomegaly. The biased client **(b)** has a red emoji of a mouse in the upper left corner.



(a) Unbiased **(b)** Biased

Figure 4. Examples of unbiased and biased (imperfectly interoperable) images from CheXpert dataset for class Pleural effusion. The biased client **(b)** has a chest drain indicated by an arrow.

Experimental details

i. Centralized baseline. As a baseline, we follow the architecture and optimization parameters from the `ProtoPNet` paper [15] (using the VGG19 [27] or DenseNet [28] implementation pretrained on ImageNet [29]) to learn a centralized model with centralized prototypes (**CMCP**) on the whole dataset. We used ten prototypes of size $1 \times 1 \times 128$ per class. We report average accuracy for birds data and **balanced** average accuracy for the imbalanced CheXpert dataset:

$$Balanced\ accuracy = \frac{Sensitivity + Specificity}{2} \tag{6}$$

ii. IID-FL local baseline. We made an IID partition of the data over four clients and trained local ProtoPNets with local prototypes for each (**LMLP**).

iii. IID-FL global baseline. Using the FL setup above, global models with global prototypes (**GMGP**) were trained according to the scheme depicted in Fig 2. The training is composed of three or six communication rounds between the clients and the server. The server initializes a ProtoPNet model and sends it to the clients who learn LMLP. After five epochs, a subset of LP is communicated to the server and aggregated. Importantly, during this training stage, each client keeps the pre-trained convolutional weights frozen and trains two additional convolutional layers. Each of the next communication rounds includes the following steps:

- **Local training.** Each client trains convolutional layers, prototype layer, and final fully connected layer locally (on their own dataset).
- **Local prototypes.** A set of local prototypes $\mathbf{P}_{loc}$ and weights $\mathbf{W}_{loc}$ is sent to the server after every ten epochs.
- **Global prototypes.** The server aggregates local prototypes by averaging to create a set of ten global prototypes $\mathbf{P}_{glob}$ and weights $\mathbf{W}_{h,glob}$. These are shared back to each client to iterate training.
- **Interpretability.** Each client visualizes interoperability shifts by projecting each prototype onto the nearest latent training patch from the same class and then optimizing the final layer to improve accuracy.

It is worth noting that after training, we have as many global models as clients. All these models have global prototypes and different $w_{conv,n}$ whose updates always stayed locally. Importantly, we purposefully limit the global training of convolutional layers, for the purpose of comparing interoperability, thus the performance of GMGP is expected to be lower than CMCP.

iv. IIO-FL Experiment. Finally, we trained $\mathbf{LMLP^b}$ and $\mathbf{GMGP^b}$ in an FL setting with three unbiased IID clients and one IIO client (with systematic bias in one class (Fig. 3, 4 and Fig. 9 in SI). We visually inspect the prototypes learned locally and globally to detect IIO shifts between clients without sharing any original data.

Results

Quantitative results

IID setting

The quantitative results for CMCP (i.e. `ProtoPnet` baseline), LMLP, and GMGP trained on unbiased IID data for birds and CheXpert datasets are presented in Table 1 and Table 2, respectively. For the birds data, CMCP achieves an accuracy of 86.02 % and LMLP and GMGP perform slightly worse with 82.76 and 81.25 % accuracy, respectively.

Models learned on the CheXpert dataset demonstrate worse performance in terms of the absolute values of balanced accuracy. However, the models were not specifically optimized for performance, but rather to create an efficient experimental setup on which we can visualize relative differences. For the CheXpert dataset, CMCP gives 73.61 % and 75.36 % balanced accuracy for cardiomegaly and pleural effusion classification, respectively. LMLP and GMP achieve 70.70 and 70.85 % for cardiomegaly and 71.70 and 74.53 % for pleural effusion classes. The values of classification sensitivity and specificity used to compute balanced accuracy can be found in SI Table 6.

Table 1. Centralized vs FL IID settings for birds. Classification accuracies for **CMCP** (centralized model/prototype), **LMLP** (local model/prototype), and **GMGP** (global model/prototype) trained without data bias on CUB200-2011 dataset. The mean computed over four clients is shown with standard deviation.

Model	**CMCP**	**LMLP**	**GMGP**
Accuracy (% ±SD)	86.02	82.76 ±1.14	81.25 ±0.49

IIO setting

In this case, we compare models' performance separately on unbiased and biased data (Table 3 and Table 4). We can see that for the birds dataset, both LMLP^b and GMGP^b

Table 2. Centralized vs FL IID settings for X-rays. Classification balanced accuracies for **CMCP** (centralized model/prototype), **LMLP** (local model/prototype), and **GMGP** (global model/prototype) trained without data bias on CheXpert dataset for cardiomegaly and pleural effusion classes. The mean computed over four clients is shown with standard deviation.

Model	**CMCP**	**LMLP**	**GMGP**
Cardiomegaly classification			
Balanced accuracy (%, ±SD)	73.61	70.70 ±0.44	70.85 ±0.45
Pleural effusion classification			
Balanced accuracy (%, ±SD)	75.36	71.70 ±0.37	74.53 ±0.35

perform better on biased (85.8 and 83.3 %, respectively) dataset than on the unbiased one (76.1 and 75.3 % for LMLP^b and GMGP^b, respectively). In addition to average accuracy over all classes, we also show accuracy for a biased class. In this case, the difference in performance on these two datasets is clearer: LMLP^b achieves 100.0 % accuracy on the biased dataset for a biased class and 0.0 % on the unbiased dataset, and GMGP^b gives 85.7 and 4.8 %, respectively.

We can see a similar behavior of models trained on medical data to the ones trained on bird images. Namely, both LMLP^b and GMGP^b give 100.0 % accuracy on biased data and 50.0 % on unbiased one in the case of cardiomegaly classification. For pleural effusion classification, these models achieve 77.31 and 66.15 % of balanced accuracy on biased data and 49.18 and 50.33 % on unbiased set. Sensitivity and specificity for IIO setting are shown in SI Table 7.

Table 3. Effect of IIO bias in FL for birds. Classification accuracies for local and global models trained in an FL setting on the CUB200-2011 dataset that has a biased client. For each model, the value in the left subcolumn corresponds to the test set of a biased client, and in the right subcolumn, there is an average value over the test sets of unbiased clients with standard deviation where possible. Performance is shown from bad to good.

Model	LMLP^b		GMGP^b	
Test set	Biased	Unbiased	Biased	Unbiased
all classes	**85.8**	**76.1±1.7**	**83.3**	**75.3±1.2**
biased class	**100.0**	**0.0**	**85.7**	**4.8±4.7**

Qualitative results

This section presents the prototypes learned in the IID and IIO FL setting for both datasets (Fig. 5, 6, 7). More examples of prototypes for all models can be found in SI (Fig. 10 - 21).

For the birds dataset, local and global prototypes (Fig. 5) learned on unbiased data, activate a meaningful patch in both biased and unbiased test images (bird's head). LMLP^b, however, *looks at* the emoji in the lower left corner in the biased image and at the same corner in the unbiased test image. Interestingly, GMGP^b does not activate the emoji in the biased image but looks at the nearby area.

We see a similar tendency for the CheXpert data. In the cardiomegaly class (Fig. 6),

Table 4. Effect of IIO bias in FL for X-rays. Classification balanced accuracies for local and global models trained in an FL setting with one biased client on the CheXpert dataset for cardiomegaly and pleural effusion classes. For each model, the value in the left subcolumn corresponds to the test set of a biased client, and in the right subcolumn, there is an average value over the test sets of unbiased clients with standard deviation where possible. Performance is shown from bad to good.

Model	LMLP^b		GMGP^b	
Test set	Biased	Unbiased	Biased	Unbiased
Cardiomegaly classification				
Balanced accuracy (%, ±SD)	**100.0**	**50.0±0.0**	**100.0**	**50.0±0.0**
Pleural effusion classification				
Balanced accuracy (%, ±SD)	**77.31**	**49.18±0.41**	**66.15**	**50.33±0.38**

local and global unbiased models activate the area of the enlarged heart while biased models *look at* the emoji in the upper left corner of a test image. In the pleural effusion class (Fig. 7), unbiased models activate the bottom of the lungs in an X-ray while biased ones (trained on the images with chest drains) highlight drains that can locate in different parts of the lungs.

We discuss the prototypes presented here in the next section.

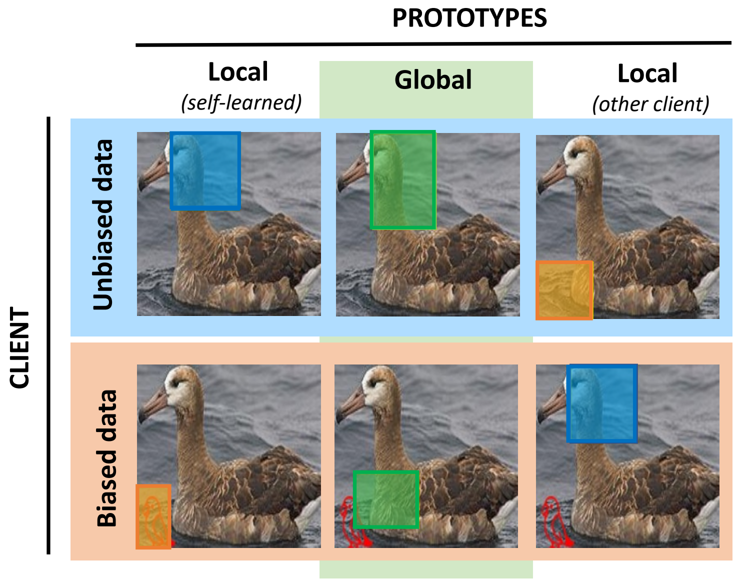


Figure 5. Prototypes learned in IIO FL setting for birds. Examples of a test image with bounding boxes indicating the most activated patches by the prototypes learned locally and globally on unbiased and biased CUB200-2011 datasets in an IIO-FL setting.

The prototypes learned locally (by LMLP and LMLP^b) and globally (by GMGP^b) can be also compared by computing the Euclidean distance between them for each client. These results, normalized between 0 and 1, are shown in Fig. 8. We can see that local and global prototypes for unbiased clients differ much less than those for the biased client. The distances for pleural effusion classification can be found in SI Fig. 22.

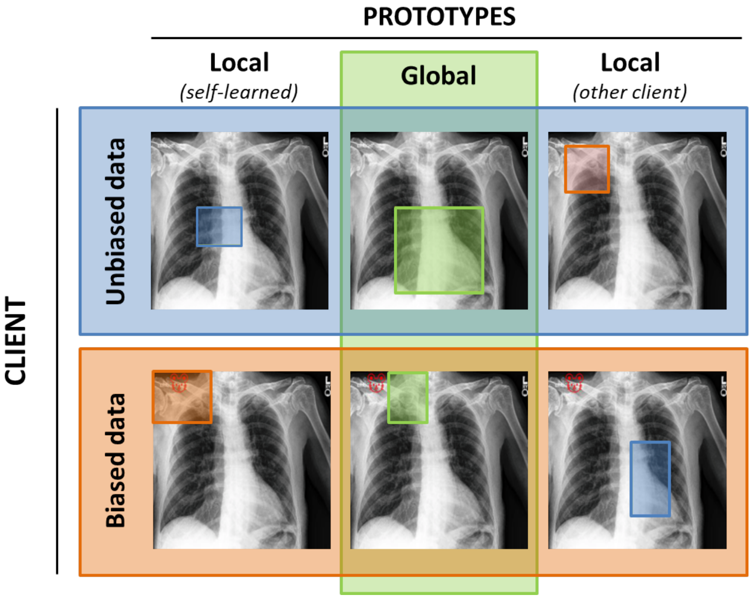


Figure 6. Prototypes learned in IIO FL setting for X-rays. Examples of a test image with bounding boxes indicating the most activated patches by the prototypes learned locally and globally on unbiased and biased CheXpert datasets in an IIO-FL setting for *cardiomegaly* classification.

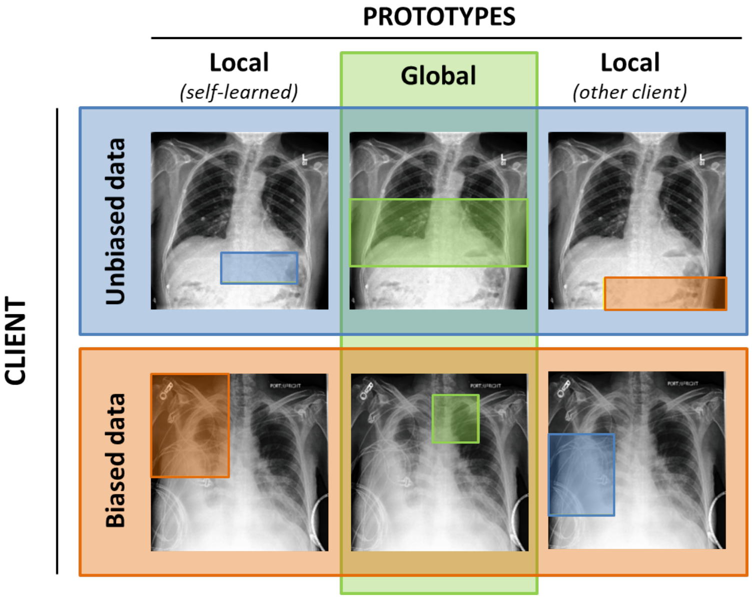


Figure 7. Prototypes learned in IIO FL setting for X-rays. Examples of a test image with bounding boxes indicating the most activated patches by the prototypes learned locally and globally on unbiased and biased CheXpert datasets in an IIO-FL setting for *pleural effusion* classification.

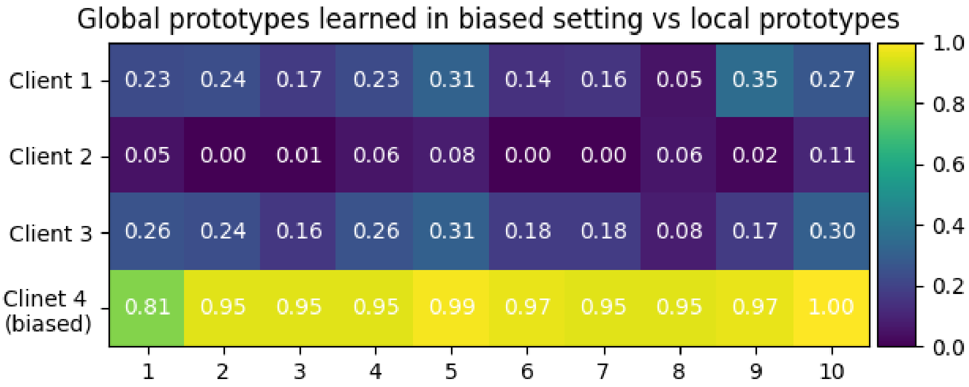


Figure 8. Euclidean distances between local and global (IIO FL setting) prototypes for cardiomegaly classification for each client after min-max normalization.

Discussion

Birds dataset

We begin the analysis of our results with the model benchmark dataset of bird species. As expected local models perform worse than a centralized model due to the smaller dataset (Table 1). As the training of the global model is purposely restricted to highlight differences in clients' data, we do not expect the GMGP to significantly improve upon LMLP, which is the case. Indeed, only part of the GMGP network is updated globally (prototypes $\mathbf{P}_n$ and weights w_{hn}), while the convolutional layers are kept local. Nevertheless, the prototypes learned in these three settings are rather similar as can be seen in SI, Fig.10.

Bias injection into one client's dataset has a strong effect on model performance and prototypes (Table 3). Both local LMLP^b and global GMGP^b expectedly perform much worse on unbiased data than on biased one. Notably, the local model gives 100.0% accuracy for a biased class on a dataset with emoji and 0.0% accuracy for this class on unbiased data. This clearly indicates that the model assigns this class based on the presence of bias and thus suffers catastrophic failure in the absence of bias. Indeed, as shown in Figure 5, LMLP (trained on unbiased data) and LMLP^b activate totally different patches on the same test image with and without emoji.

Communication of prototypes with other clients brings slight improvement to the model's performance (Table 3). GMGP^b has 4.76% accuracy for biased class on unbiased data and 85.71% on biased one. From Figure 5, we can see that the global model for a biased client does not activate the emoji as its local counterpart but still looks at a less informative patch close to it. Global model for good clients, however, looks at the bird's head as it does the local one. The distance between these prototypes can not only be visualized, but also computed, meaning that it could be used to derive a personalization weight that is visually interpretable.

The negative effect of data bias is additionally demonstrated in SI Table 5. Here, we show how different models predict a class for biased and unbiased test images.

CheXpert dataset

Experiments on X-ray images demonstrate a more real-world use case with its accompanying challenges. The first difficulty we faced was an inability to perform multiclass classification to distinguish all 14 classes in the original dataset since prototypes for many of them are expected to activate similar regions in the images.

To simplify the task on this complex dataset, we carried out a binary one-vs-rest classification for cardiomegaly and pleural effusion classes. These classes were chosen due to the expected distinct localization of prototypical regions (near a heart and bottom part of the lungs) in comparison to some other classes (for example, lung lesions and pneumonia prototypical parts can be spread throughout the lungs). In this setting, however, we met a common for medical data challenge, namely data imbalance. To tackle this issue, we undersampled a negative class in the training set to make it equal to the size of a positive class. We kept the test set as it is and reported balanced accuracy instead of ordinary accuracy.

IID FL setting. As we can see from Table 2, though overall balanced accuracy values for the models trained on unbiased data are low, they follow the same tendency as accuracies for the bird dataset. Local and global models for both classes perform expectedly worse than corresponding centralized models.

The prototypes learned in this setting (see SI) present class characteristic regions clear for humans. For example, in SI Fig. 12 we can see that in order to classify an image as cardiomegaly, a centralized model *looks at* the whole enlarged heart or at the collarbone level in the center pointing out the extended aorta characteristic for this condition. As for the pleural effusion classification, most prototypes activate the lower part of the lung where fluid accumulates in this disorder.

IIO FL setting. For the medical dataset, we tried two different ways of introducing bias into one client's dataset. We started with a simple setting similar to the one we had for the birds dataset, namely, we added a small emoji to the left upper corner of images in the cardiomegaly class. In this case, both LMLP^b and GMGP^b solely rely on the presence of bias to predict a positive class resulting in 100.0 % balanced accuracy for test on biased data and 50.0 % on unbiased data.

Fig. 6 demonstrates this result by showing parts of a test image activated by local and global models trained on biased and unbiased data. It is interesting to note that for this binary classification task, adding bias to a positive class also changes the prototypes for a negative class. This effect can be seen in SI Fig. 14 and 16, where prototypical parts for a negative (unbiased) class turned out to be left upper regions where there was an emoji for a positive class. Obviously, these prototypes have no practical value in classifying cardiomegaly.

To experiment with more practically relevant data bias, we paid attention to the fact that often patients with pleural effusion get chest drains to remove the fluid from their lungs. Thus the presence of chest drains in X-ray images can serve as bias for pleural effusion class. We trained local and global models in the setting where one client has images with chest drains in the positive class (note that these images do not necessarily have actual pleural effusion) and any other X-rays without chest drains in the negative class (in the same way, these images may or may not have pleural effusion).

As we can see from Table 4, LMLP^b fails on predicting pleural effusion in the absence of chest drain (unbiased data), and GMGP^b performs slightly better on this dataset due to communication with unbiased clients. Note, that in opposite to the emoji bias, the chest drain is more difficult to learn and it does not lead to 100.0 % accuracy on biased data. LMLP^b achieves 77.31 % accuracy and GMGP^b has 66.15 % on biased data. This large difference in LMLP^b and GMGP^b balanced accuracy shows the benefit of communication to identify systemic bias in clients' data.

Fig. 7 demonstrates which regions in biased and unbiased images are most important to classify pleural effusion from the *point of view* of local and global models trained with or without data bias. We can see that indeed LMLP^b prefers to activate an image patch with a chest drain as important for pleural effusion class instead of an actually import bottom area of the lung which is activated by the local and global models trained on unbiased datasets. GMGP^b, however, does not look at the same area as LMLP^b but still activates a patch in the upper lungs region rather than the lower one.

The difference between local and global prototypes for each client is also clearly seen in Fig. 8 and SI Fig. 22 showing the Euclidean distance between them. The distances are much smaller for the unbiased clients than for the biased one. Sharing these distances with the server can help to identify a biased client and take measures to avoid the negative effect of low data interoperability on global models.

Privacy aspects

As mentioned before, privacy is a key advantage of FL. Nevertheless, sharing models' parameters can still compromise clients' data due to the possibility of training an adversarial network and retrieving the actual data from it [30]. In the case of our `inDISCO` model, however, this risk is lower since clients send to the server **only their prototypes** and **weights of the final layer** always keeping convolutional weights local. Additional steps such as encryption of the shared parameters can be done to better preserve privacy.

Moreover, if clients decide to further examine the differences in their datasets they can share the prototypes projected on their training or test set after defining their privacy *budget*, e.g., the number and size of prototypical patches that can be controlled during training.

Limitations and future work

This work develops an approach for interpretable identification of data interoperability in FL for imaging data. To the best of our knowledge, this is the first attempt in this research direction, and thus there is still space for future improvements. The following steps can be done to address existing limitations:

1. More complex biased settings can be used to move the experiments closer to the real-world scenario;

2. So far experiments with only four clients have been done thus scaling up is important;

3. Since the quantitative performance of the models trained for CheXpert data is not good enough, model architecture adjustments, additional data preprocessing, and/or optimization are necessary;

4. It would be interesting to try other medical datasets to evaluate the robustness of our approach;

5. Though clients do not share their actual data and keep convolutional updates locally, the total absence of privacy leakage is not guaranteed, and thus comparison of our `inDISCO` method with other FL approaches in terms of privacy would be useful.

Conclusion

`inDISCO` is a novel extension of `ProtoPnet`, which allows interpretable and privacy-preserving identification, attribution, and quantification of data bias in federated learning for imaging data. `inDISCO` creates transparency from black box data without compromising privacy which gives this approach a potential for application in the privacy-sensitive medical domain.

The part of this work with the results obtained on birds data was submitted to the ICML 2023 workshop ”Federated Learning and Analytics in Practice: Algorithms, Systems, Applications, and Opportunities”. The submitted paper can be found in SI.

Acknowledgements

We would like to acknowledge Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin, the authors of the ”This Looks Like That: Deep Learning for Interpretable Image Recognition” paper on which we based our project and, also, David Roschewitz, Mary-Anne Hartley, Luca Corinzia, and Martin Jaggi, the authors of the ”iFedAvg – Interpretable Data-Interoperability for Federated Learning” paper. We also thank Arnout Devos, Sai Praneeth Karimireddy, and Amir Rezaie for their valuable advice and kind support.

References

1. Piccialli F, Di Somma V, Giampaolo F, Cuomo S, Fortino G. A survey on deep learning in medicine: Why, how and when? Information Fusion. 2021;66:111–137.

2. Shen D, Wu G, Suk HI. Deep Learning in Medical Image Analysis. Annu Rev Biomed Eng. 2017;19:221–248.

3. Landi I, Glicksberg BS, Lee HC, Cherng S, Landi G, Danieletto M, et al. Deep representation learning of electronic health records to unlock patient stratification at scale. npj Digital Medicine. 2020;3.

4. Barnett AJ, Sharma V, Gajjar N, Fang J, Schwartz M D F, Chen C, et al. Interpretable deep learning models for better clinician-AI communication in clinical mammography. Proc SPIE 12035, Medical Imaging 2022: Image Perception, Observer Performance, and Technology Assessment, 1203507. 2022;doi:10.1117/12.2612372.

5. McMahan HB, Moore E, Ramage D, Hampson S, y Arcas BA. Communication-Efficient Learning of Deep Networks from Decentralized Data. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS). 2017;54.

6. Ribeiro MT, Singh S, Guestrin C. ”Why Should I Trust You?”: Explaining the Predictions of Any Classifier. arXiv:160204938. 2016;.

7. Lundberg S, Lee SI. A Unified Approach to Interpreting Model Predictions. arXiv:170507874. 2017;.

8. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. Journal of Computer Vision (IJCV). 2019;.

9. Simonyan K, Vedaldi A, Zisserman A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. Workshop at International Conference on Learning Representations. 2014;.

10. Singh A, Sengupta S, Lakshminarayanan V. Explainable deep learning models in medical image analysis. Journal of Imaging. 2020;6.

11. Shrikumar A, Greenside P, Kundaje A. Learning Important Features Through Propagating Activation Differences. PMLR. 2017;70:3145–3153.

12. Chen H, Lundberg S, Lee SI. Explaining Models by Propagating Shapley Values of Local Components. arXiv:191111888. 2019;.

13. Adebayo J, Gilmer J, Muelly M, Goodfellow I, Hardt M, Kim B. Sanity Checks for Saliency Maps. Advances in Neural Information Processing Systems. 2018;31.

14. Rudin C, Chen C, Chen Z, Huang H, Semenova L, Zhong C. Interpretable machine learning: Fundamental principles and 10 grand challenges. Statistics Surveys. 2022;16:1 – 85.

15. Chen C, Li O, Tao C, Barnett AJ, Su J, Rudin C. This Looks like That: Deep Learning for Interpretable Image Recognition. Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019; p. 8930–8941.

16. Barnett SFRTCea A J. A case-based interpretable deep learning model for classification of mass lesions in digital mammography. Nat Mach Intell. 2021;3:1067–1070.

17. Hase P, Chen C, Li O, Rudin C. Interpretable Image Recognition with Hierarchical Prototypes. AAAI Conference on Human Computation & Crowdsourcing. 2019;.

18. Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated Optimization in Heterogeneous Networks. arXiv:181206127. 2018;.

19. Karimireddy SP, Kale S, Mohri M, Reddi SJ, Stich SU, Suresh AT. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. 37th International Conference on Machine Learning, ICML 2020. 2020; p. 5088–5099.

20. Hsu TMH, Qi H, Brown M. Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. NeurIPS Workshop on Federated Learning. 2019;.

21. Reddi S, Charles Z, Zaheer M, Garrett Z, Rush K, Konečný J, et al. Adaptive Federated Optimization. ICLR. 2021;.

22. Arivazhagan MG, Aggarwal V, Singh AK, Choudhary S. Federated Learning with Personalization Layers. arXiv:191200818. 2019;.

23. Roschewitz D, Hartley MA, Corinzia L, Jaggi M. IFedAvg: Interpretable Data-Interoperability for Federated Learning. arXiv:210706580. 2021;.

24. Wah C, Branson S, Welinder P, Perona P, S B. The Caltech-UCSD Birds-200-2011 Dataset. California Institute of Technology; 2011.

25. 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. The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19). 2019;.

26. Jiménez-Sánchez A, Juodelye D, Chamberlain B, Cheplygina V. Detecting Shortcuts in Medical Images - A Case Study in Chest X-rays. arXiv:221104279. 2022;.

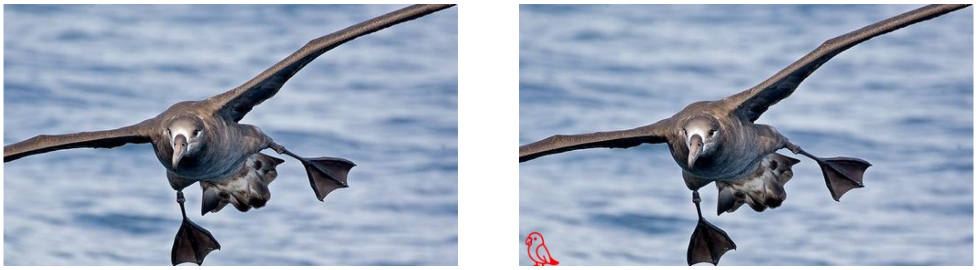
27. Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR); 2015.

28. Huang G, Liu Z, van der Maaten L, Weinberger KQ. Densely Connected Convolutional Networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017; p. 2261–2269.

29. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. ImageNet: A Large-Scale Hierarchical Image Database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE; 2009. p. 248–255.

30. Zhang J, Chen Y, Li H. Privacy Leakage of Adversarial Training Models in Federated Learning Systems. CVPR workshop ”The Art of Robustness”. 2022;.

Supporting information



(a) Unbiased

(b) Biased

Figure 9. Examples of unbiased and biased (imperfectly interoperable) images from CUB200-2011 dataset used in this work. The biased client **(b)** has a red emoji of a parrot in the lower left corner.

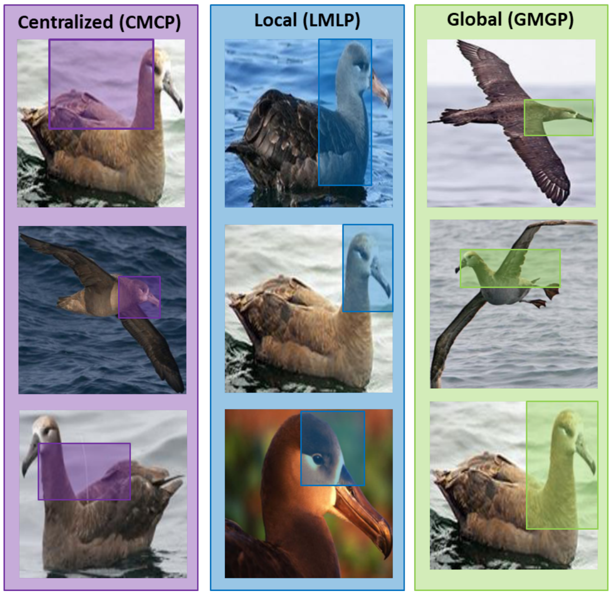


Figure 10. Examples of training images with bounding boxes indicating centralized, local, and global prototypes learned on **unbiased** CUB200-2011 data.

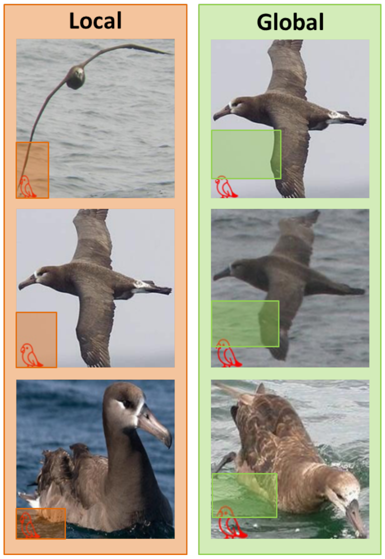


Figure 11. Examples of training images with bounding boxes indicating local and global prototypes learned on **biased** CUB200-2011 data.

Table 5. Prediction results of local and global models for two biased and two unbiased test images from birds dataset coming from biased and unbiased during training classes.

Model	Biased class		Unbiased class	
	Test image			
	Unbiased	Biased	Unbiased	Biased
LMLP	COR	COR	COR	COR
LMLP^b	INCOR	COR	COR	INCOR
GMGP	COR	COR	COR	COR
GMGP^b	INCOR	COR	COR	INCOR

Table 6. Classification sensitivity and specificity for **CMCP** (centralized model/prototype), **LMLP** (local model/prototype), and **GMGP** (global model/prototype) trained without data bias on CheXpert dataset for cardiomegaly and pleural effusion classes. The mean computed over four clients is shown with standard deviation.

Model	**CMCP**	**LMLP**	**GMGP**
Cardiomegaly classification			
Sensitivity, ±SD	0.60	0.61 ±0.04	0.58 ±0.03
Specificity, ±SD	0.87	0.80 ±0.03	0.84 ±0.03
Pleural effusion classification			
Sensitivity, ±SD	0.76	0.58 ±0.01	0.83 ±0.02
Specificity, ±SD	0.74	0.86 ±0.00	0.66 ±0.02

Table 7. Classification sensitivity and specificity for local and global models trained in an IIO FL setting with one biased client on the CheXpert dataset for cardiomegaly and pleural effusion classes. For each model, the value in the left subcolumn corresponds to the test set of a biased client, and in the right subcolumn, there is an average value over the test sets of unbiased clients with standard deviation where possible.

Model	LMLP^b		GMGP^b	
Test set	Biased	Unbiased	Biased	Unbiased
Cardiomegaly classification				
Sensitivity, ±SD	1.0	0.0±0.0	1.0	0.0±0.0
Specificity, ±SD	1.0	1.0±0.0	1.0	1.0±0.0
Pleural effusion classification				
Sensitivity, ±SD	0.58	0.03 ±0.00	0.42	0.12 ±0.01
Specificity, ±SD	0.97	0.95 ±0.00	0.90	0.88 ±0.01

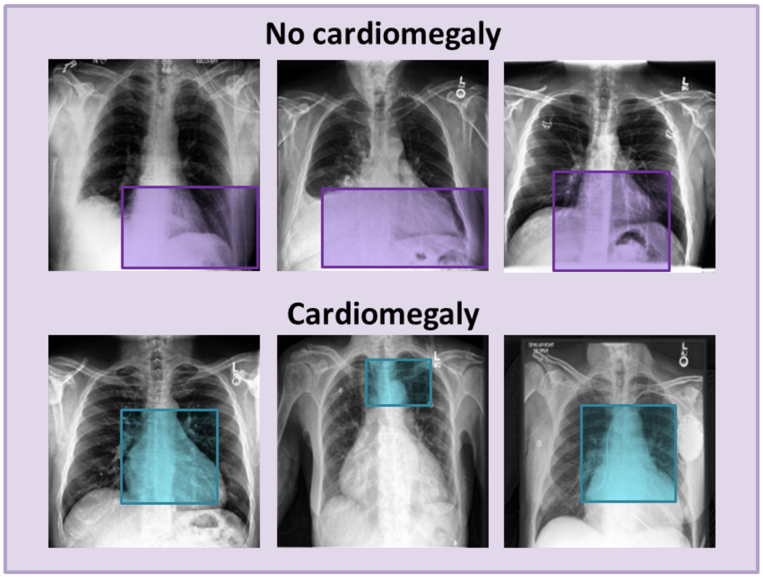


Figure 12. Examples of training images with bounding boxes indicating centralized prototypes learned on **unbiased** CheXpert data for *cardiomegaly* classification.

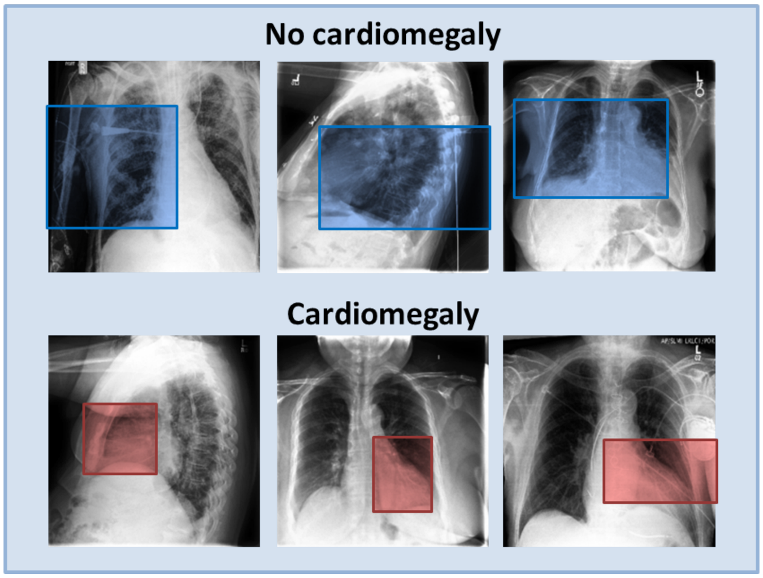


Figure 13. Examples of training images with bounding boxes indicating local prototypes learned on **unbiased** CheXpert data for *cardiomegaly* classification.

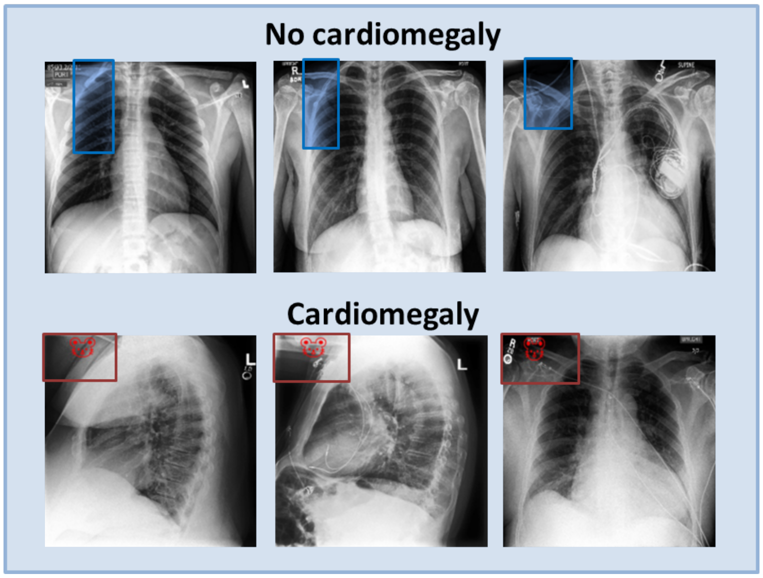


Figure 14. Examples of training images with bounding boxes indicating local prototypes learned on **biased** CheXpert data for *cardiomegaly* classification.

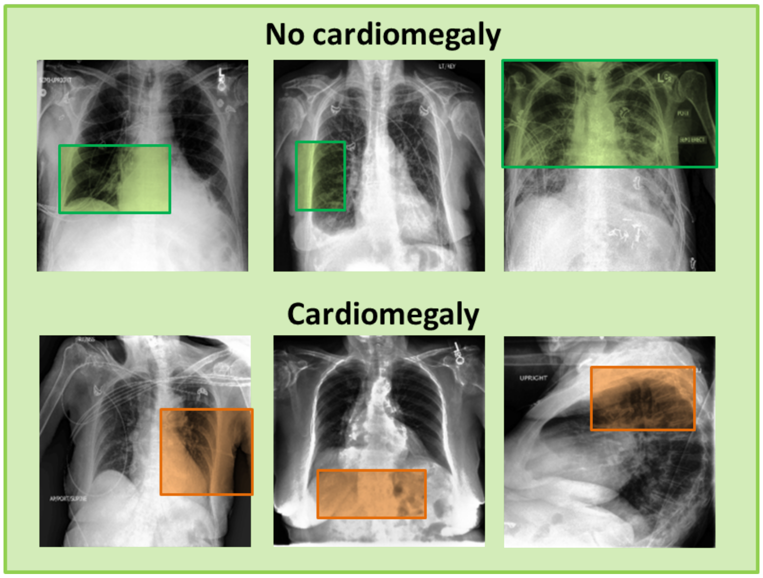


Figure 15. Examples of training images with bounding boxes indicating global prototypes learned on **unbiased** CheXpert data for *cardiomegaly* classification.

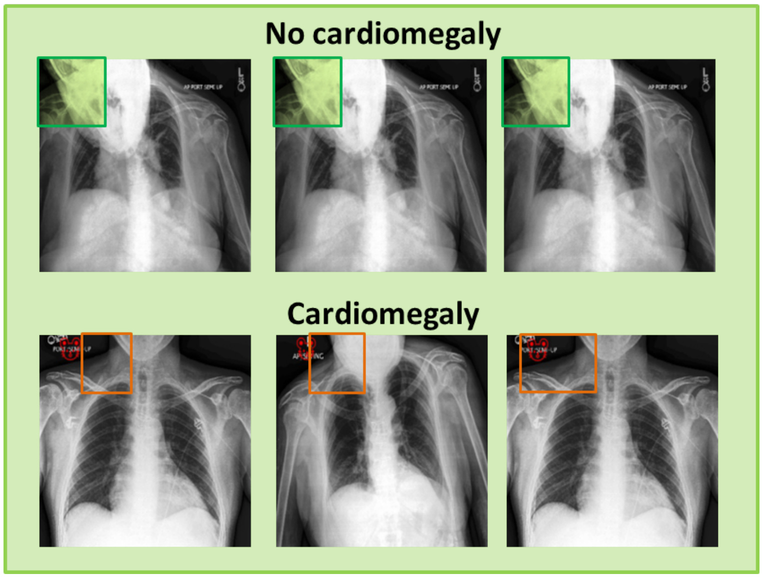


Figure 16. Examples of training images with bounding boxes indicating global prototypes learned on **biased** CheXpert data for *cardiomegaly* classification.

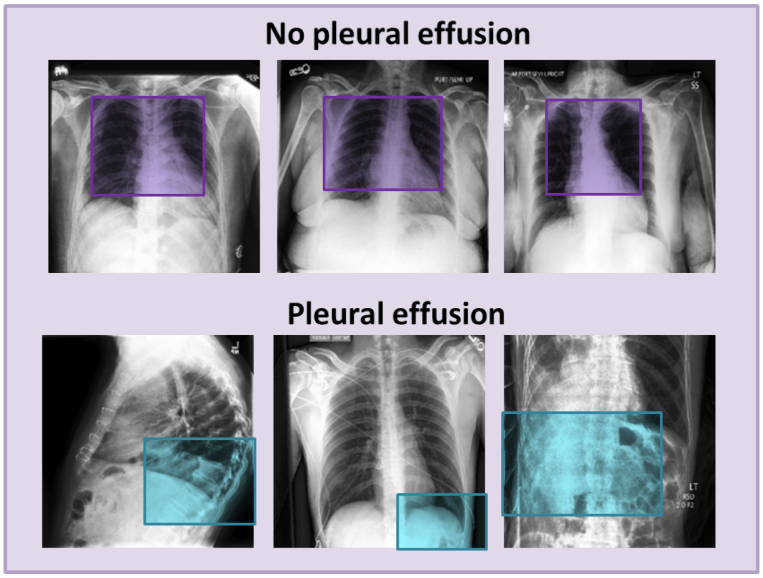


Figure 17. Examples of training images with bounding boxes indicating centralized prototypes learned on **unbiased** CheXpert data for *pleural effusion* classification.

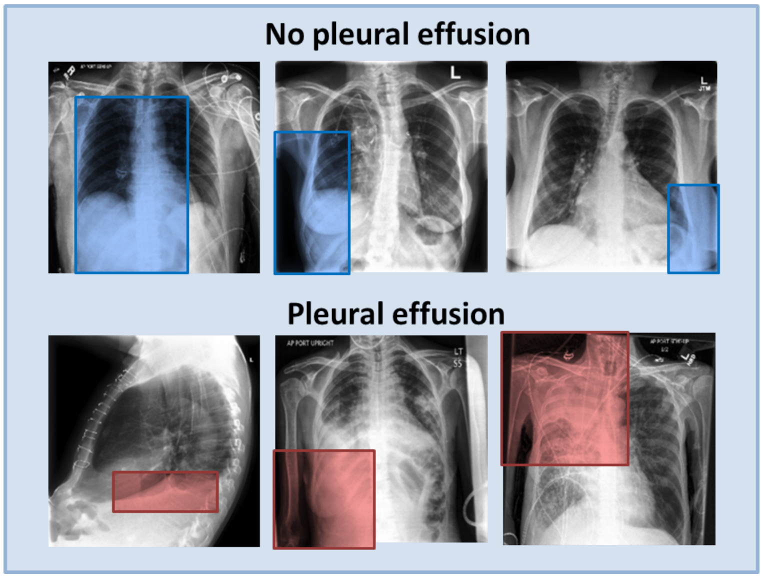


Figure 18. Examples of training images with bounding boxes indicating local prototypes learned on **unbiased** CheXpert data for *pleural effusion* classification.

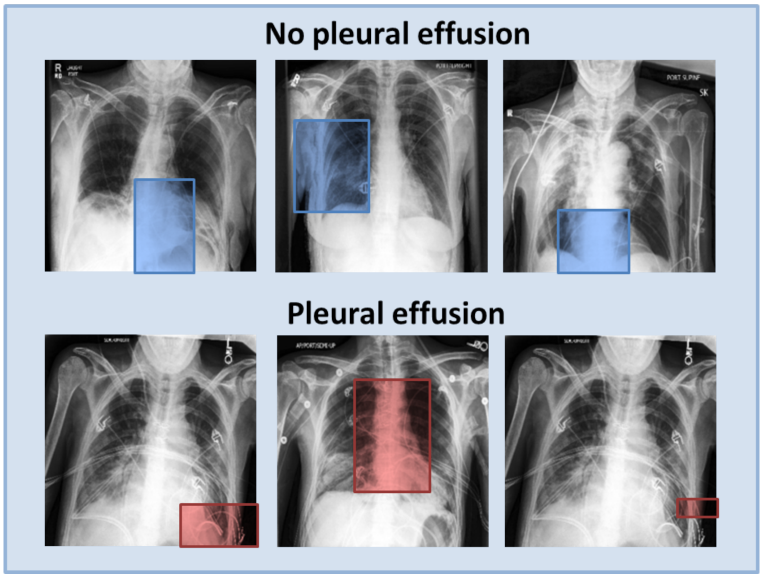


Figure 19. Examples of training images with bounding boxes indicating local prototypes learned on **biased** CheXpert data for *pleural effusion* classification.

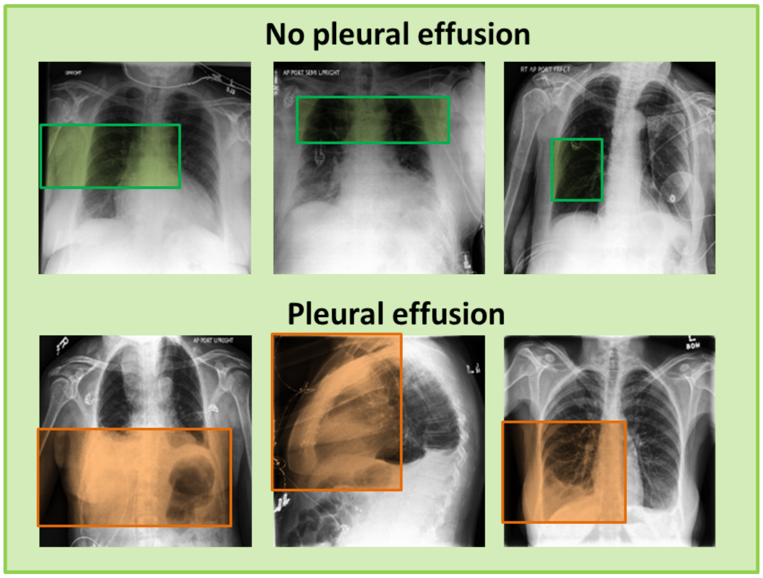


Figure 20. Examples of training images with bounding boxes indicating global prototypes learned on **unbiased** CheXpert data for *pleural effusion* classification.

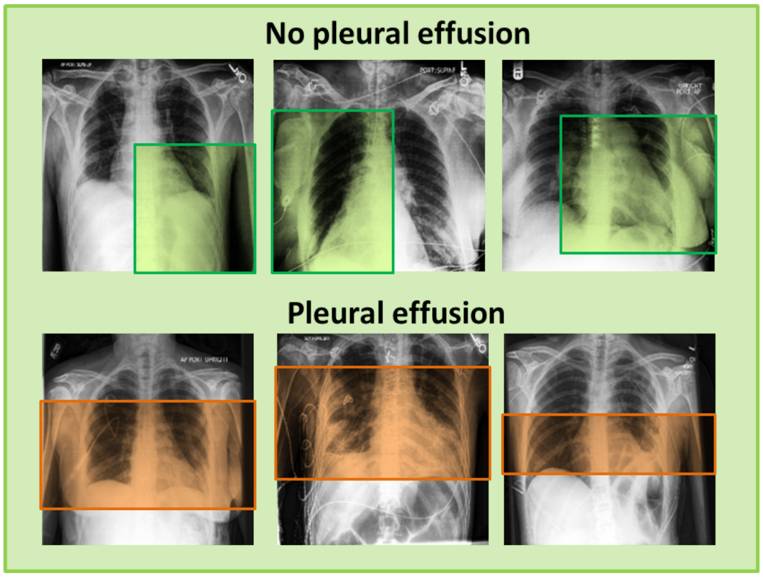


Figure 21. Examples of training images with bounding boxes indicating global prototypes learned on **biased** CheXpert data for *pleural effusion* classification.

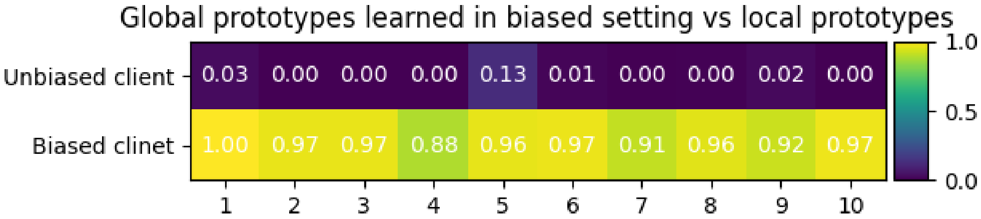


Figure 22. Euclidean distances between local and global (IIO FL setting) prototypes for pleural effusion classification for a biased and unbiased client after min-max normalization.

inDISCO: INterpretable DIStributed COllaborative learning for images

Anonymous Authors[1]

Abstract

DIStributed **CO**llaborative (**DISCO**) learning allows several data owners (clients) to learn a joint model without sharing data. While this approach could transform data usage in privacy-sensitive domains such as healthcare, the restricted access to each client's data limits interpretability and may conceal bias or interoperability mismatches that could compromise model performance. This issue is particularly important for image data since most deep neural networks for images are already black-box models. We address this problem with `inDISCO`, which adapts a well-known interpretable prototypical part learning network (ProtoPNet) to a federated setting allowing members of the federation to directly visualize the differences in the features learned from each client. We show that it can identify, attribute, quantify, and potentially correct bias in distributed collaboration without sharing any data. This work could be extended to interpretable personalization in federated learning.

1. Introduction

"*What you can't see can't hurt you...?*" — Proverb

As the *bigness* of big data becomes ever bigger and more granular, so too does its potential power, value, risk, and the legal constraints of sharing it. To address this limitation, federated learning (**FL**) was proposed by (McMahan et al., 2017) to allow multiple data owners (clients) to collaboratively train a model while keeping their datasets locally. While FL is potentially transformative for domains with privacy-sensitive data such as healthcare, the privacy gained comes at a major cost to transparency. Indeed, the real-world use of deep learning is already limited by black box *models*, and FL compounds the issue with black box *data*. Visualizing data is especially important when applied to real-world data which is often biased or imperfectly interoperable. Understanding differences will allow an informed selection of collaborators and enable explainable predictions.

Problem setting. The inability to visualise data across clients in FL is particularly problematic for imaging data since the deep learning approaches for this modalitiy are already poorly interpretable. This combination of black box model and black box data, obfuscates both the reasoning process of the model as well as the quality of its data. This work addresses the question of how we can achieve data transparency without compromising its privacy. Specifically, we ask, how can we identify, quantify, explain, and even correct for bias in unseen imaging data in the FL setting while preserving privacy?

We adapt a well-known interpretability method introduced by Chen et al. (2019), called prototypical part learning (`ProtoPNet`). Their implementation (Appendix, Fig 3) uses a CNN to create a set of patches in the latent space, a prototypical patch is learned from this set. Classification then relies on a similarity score computed between this learned prototype and a latent representation of a test image. A prototype can be visualized by highlighting the patch most activated by this prototype.

We propose to adapt ProtoPNet to FL. As summarized in Fig 1, clients each learn their own local prototypes as well as globally in communication with each other. The patches most activated by each of these prototypes can be visualized and compared on each client's local test set. By comparing global and local prototypes the clients can assess the interoperability of the data. Thus, we can introduce interpretability to FL and directly examine the predictive impact of other data without compromising clients' privacy.

Our main contributions are as follows:

1. We formalize a use case and create an imperfectly interoperable benchmark image dataset.
2. We introduce `inDISCO` adapting ProtoPNet to FL and compare its performance to baseline models.
3. We demonstrate how inDISCO helps to identify a biased client in FL without disclosing the data.
4. Finally, we propose a new approach to use inDISCO for interpretable personalization.

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

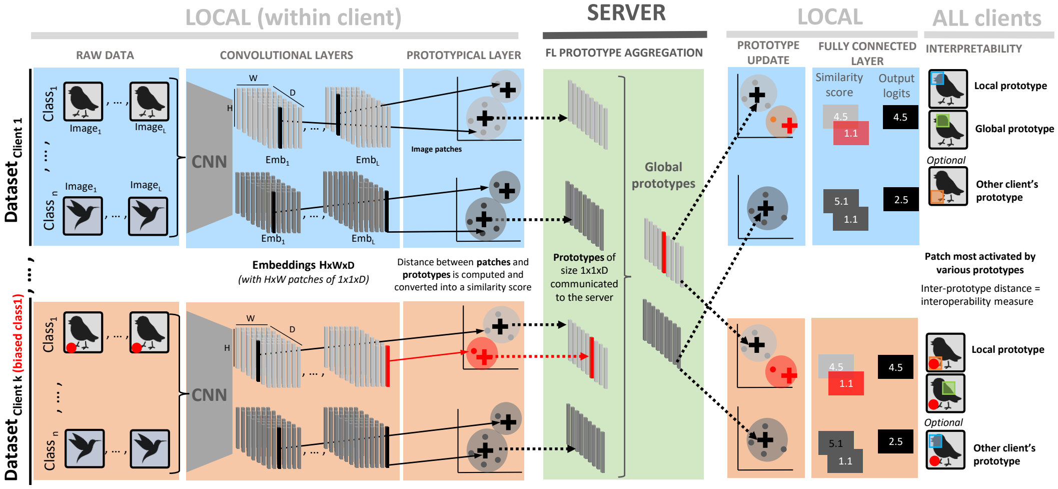


Figure 1. **inDISCO architecture**. Several clients ($client_1$,..., $client_k$) wish to learn a model in a federated setting via a SERVER. inDISCO, passes raw images through a CNN to create embeddings of size $[H \times W \times D]$ in the latent space ($Emb_1, ..., Emb_L$), which are divided up into $H \times W$ image patches $[1 \cdot 1 \cdot D]$. Prototypical patches (**black**) are clustered and a prototype (**+**) is learned for each image. $Class_1$ of $Client_k$ has **systematic bias** which contaminates the prototype pool (**+**). Ten prototypes for each class are shared to the SERVER by each client and aggregated to make ten global prototypes. These are then pushed back to the clients. Classification is based on a similarity score computed between the prototype and the image patches. In the final panel, we see how global and local prototypes can be compared to directly visualize shifts without sharing any original data.

Related work. Explaining how neural networks make predictions is critical to ensuring trust in real-world use cases. This is particularly important in the medical imaging domain, for example, where evaluating the alignment of logical plausibility of a prediction helps protect patients against misdiagnosis (Ribeiro et al., 2016; Lundberg & Lee, 2017; Selvaraju et al., 2017; Simonyan et al., 2014; Singh et al., 2020; Shrikumar et al., 2017; Chen et al., 2021; 2019). There are many posthoc techniques that aim to explain the predictions made by black-box models. The most popular methods are LIME (Ribeiro et al., 2016), SHAP (Lundberg & Lee, 2017), Feature visualization [1], and saliency mapping with Grad-CAM (Selvaraju et al., 2017).

In opposition to black-box models, there exist inherently interpretable models whose decision-making process is made to be transparent. An example of such a model is a prototypical part neural network ProtoPNet (Chen et al., 2019) that bases classification on how similar *parts* of an image in the test set are to a *prototypical part* of an image in the train set (Summarized in Appendix, Fig 3). ProtoPNet showed performance comparable with the state-of-the-art black-box deep neural networks on a popular benchmark data set while being inherently interpretable.

In FL, clients may be imperfectly interoperable (IIO), where the biggest risk is the presence of systematic bias in one client, resulting in label leakage. For instance, a set of X-rays, where the diagnosis is written on the image. The biased features learned from this client would contaminate the feature pool in an FL setting and decrease the performance of a global model. One of the approaches to address this issue is adding local personalization layers to the model. This idea is presented in the works (Arivazhagan et al., 2019) (FedPer) and (Roschewitz et al., 2021) (iFedAvg). In particular, iFedAvg is also designed to identify and visualize the clients and features causing the shift through local feature-wise affine layers f_{in} and f_{out} which can learn the feature-wise differences between clients compared with the global model, without sharing any data. While this is interpretable for tabular data (where features are intelligible), the same is not true for images. We propose applying ProtoPNet to this IIO-FL setting to best isolate interpretable features from images that can be compared and interpreted.

2. Problem formulation

Context. We trained inDISCO in an FL setting using either IID (unbiased, identically distributed classes) or IIO (imperfectly interoperable with systematic bias in a single class) data distribution among clients. A central server aggregates and updates the models' parameters. By learning local prototypes, each client identifies the parts of its training images most important for the task. In contrast, the global prototypes show the relevance for all clients on average. Finally, by examining the difference between local and global prototypes, a client can identify and quantify the presence of bias in its own or another client's dataset. Optionally sharing local prototypes among clients can further attribute the origin of bias. Taken together this information can guide appropriate action, to avoid the negative effect of data bias on

[1] https://distill.pub/2017/feature-visualization/

a global model, such as client selection or personalization by penalizing the learning of biased features.

Notation. Let us adapt the notation from (Chen et al., 2019). Given input $\mathbf{x}_n$, where $n \in \{1, ..., N\}$, each of N clients learns features with convolutional layers $f(\mathbf{x}_n)$ and m prototypes $\mathbf{P}_n = \{\mathbf{p}_{nj}\}_{j=1}^{m}$. Given a convolutional output $\mathbf{z}_n = f(\mathbf{x}_n)$, the j-th prototype of the n-th client's unit $g_{\mathbf{p}_j,n}$ in the prototype layer $g_{\mathbf{p}n}$ computes the squared L^2 distance between the prototype $\mathbf{p}_{nj}$ and all the patches of $\mathbf{z}_n$ and converts these distances into similarity scores. These scores are then multiplied by the weight matrix w_{hn} in the final fully connected layer h_n followed by softmax normalization to output class probabilities.

Local training. Given a set of training images $\mathbf{D}_n = \{(\mathbf{x}_{ni}, y_{ni})\}_{i=1}^{k}$, where k is a number of images per client, each client aims to minimize the following objective:

$$\min_{\mathbf{P}_n, w_{conv,n}} \frac{1}{k} \sum_{i=1}^{k} \text{CrsEnt}_n(h_n \circ g_{\mathbf{p}n} \circ f(x_{ni}), y_{ni}) + \\ + \lambda_1 \text{Clst}_n + \lambda_2 \text{Sep}_n, \quad (1)$$

where $w_{conv,n}$ denotes the weights of the convolutional layers learned by client n, CrsEnt is a cross-entropy loss that penalizes the misclassification, and the cluster and separation costs are defined as follows:

$$\text{Clst}_n = \frac{1}{k} \sum_{i=1}^{k} \min_{j:\mathbf{p}_{nj} \in \mathbf{P}_{ny_{ni}}} \min_{\mathbf{z}_n \in \text{patches}(f(x_{ni}))} ||\mathbf{z}_n - \mathbf{p}_{nj}||_2^2 \quad (2)$$

$$\text{Sep}_n = -\frac{1}{k} \sum_{i=1}^{k} \min_{j:\mathbf{p}_{nj} \notin \mathbf{P}_{ny_{ni}}} \min_{\mathbf{z}_n \in \text{patches}(f(x_{ni}))} ||\mathbf{z}_n - \mathbf{p}_{nj}||_2^2 \quad (3)$$

The minimization of the cluster cost (Clst) is needed to make each training image have a latent patch that is close to at least one prototype of the correct class. At the same time, every latent patch of a training image is separated from the prototypes of the incorrect class through the minimization of the separation cost (Sep). More detail ProtoPNet is found in Chen et al. (2019) and summarized in Appendix Fig3.

Federated update. At the global update step, the server performs simple averaging of all the local prototypes $\mathbf{P}_{loc} = \{\mathbf{P}_n\}_{n=1}^{N}$ and weights of the final layer $\mathbf{W}_{h,loc} = \{w_{hn}\}_{n=1}^{N}$ to obtain the global parameters:

$$\mathbf{P}_{glob} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{P}_n \quad (4) \qquad \mathbf{W}_{h,glob} = \frac{1}{N} \sum_{n=1}^{N} w_{hn} \quad (5)$$

and then sends them back to clients as shown in Fig 1.

3. Experimental setup

3.1. Dataset

Our inDISCO model was trained and evaluated on CUB-200-2011 dataset (Wah et al., 2011) of 200 bird species from which we took 20 classes. Preprocessing was performed as described by Chen et al. (2019). We introduced class-specific bias for one clients by adding an emoji to the images of a particular class (Appendix, Fig. 4). The repository can be found here (link provided at submisssion).

3.2. Experiments

i. Centralized baseline. As a baseline, we follow the architecture of `ProtoPNet` (using the VGG19 (Simonyan & Zisserman, 2015) implementation pretrained on ImageNet (Deng et al., 2009)) to learn a centralized model with centralized prototypes (**CMCP**) on the whole dataset. We learned 10 prototypes of size $[1 \cdot 1 \cdot 128]$ per class.

ii. IID-FL local baseline. We made an IID partition of the data over four clients and trained local ProtoPNets with local prototypes for each (**LMLP**).

iii. IID-FL global baseline. Using the FL setup above, global models with global prototypes (**GMGP**) were trained according to the scheme depicted in Fig 1. The training is composed of six communication rounds between the clients and the server. The server initializes a ProtoPNet model and sends it to the clients. The clients learn LMLP. After 5 epochs, a subset of LP are communicated to the server and aggregated. Importantly, during this training stage, each client keeps the pretrained convolutional weights frozen and trains two additional convolutional layers. Each of the next communication rounds includes the following steps:

- **Local convolutional layers.** Each client trains convolutional layers locally (on their own dataset).
- **Local prototypes.** The local latent representation of each image is divided into image patches and a local prototypical patch of each image is used to learn a set of ten local prototypes $\mathbf{P}_{loc}$ and weights $\mathbf{W}_{loc}$, which are sent to the server after each ten epochs.
- **Global prototypes.** The server aggregates local prototypes by averaging to create a set of ten global prototypes $\mathbf{P}_{glob}$ and weights $\mathbf{W}_{h,glob}$. These are shared back to each client to iterate training.
- **Interpretability.** Each client visualizes interoperability shifts by projecting each prototype onto the nearest latent training patch from the same class and then optimize the final layer to improve accuracy.

After training, we have as many global models as clients. All these models have global prototypes and different $w_{conv,n}$ whose updates stayed locally. Importantly, we purposefully limit the global training of convolutional layers, for the purpose of comparing interoperabilty, thus the performance of GMGP is expected to be lower than CMCP.

iv. IIO-FL Experiment. Finally, we trained **LMLP$^{\mathbf{b}}$** and **GMGP$^{\mathbf{b}}$** in an FL setting with three unbiased IID clients and one IIO client (with systematic bias in one class (Appendix Fig 4). We visually inspect the prototypes learned locally and globally to detect IIO shifts between clients without sharing any original data.

4. Results

4.1. IID setting.

The average accuracy for CMCP (i.e. `ProtoPnet` baseline), LMLP, and GMGP trained on unbiased IID data are presented in Table 1. As expected local models perform worse than a centralized model due to the smaller dataset. As the training of the global model is purposely restricted to highlight differences in clients' data, we do not expect the GMGP to significantly improve upon LMLP, which is the case. Indeed, only part of the GMGP network is updated globally (prototypes $\mathbf{P}_n$ and weights w_{hn}), while the convolutional layers are kept local. Nevertheless, the prototypes learned in these three settings are rather similar as can be seen in Appendix, Fig.5.

Table 1. **Centralized vs FL IID settings.** Classification accuracies for **CMCP** (centralized model/prototype), **LMLP** (local model/prototype), and **GMGP** (global model/prototype) trained without data bias. The mean computed over four clients are shown with standard deviation.

MODEL	CMCP	LMLP	GMGP
ACCURACY (% ±SD)	86.02	82.76 ±1.14	81.25 ±0.49

4.2. IIO-FL setting.

Bias injection into one client's dataset has a strong effect on model performance and prototypes. In this case, we compare models' accuracy separately on unbiased and biased data and in addition to average accuracy over all classes, we show accuracy for a biased class. Table 2 Both local LMLP^b and global GMGP^b expectedly perform much worse on unbiased data than on biased one. Notably, the local model gives 100.0% accuracy for a biased class on a dataset with emoji and 0.0% accuracy for this class on unbiased data. This clearly indicates that the model assigns this class based on the presence of bias and thus suffers catastrohpic failure in the absence of bias. Indeed, as shown in Figure 2, LMLP (trained on unbiased data) and LMLP^b activate totally different patches on the same test image with and without emoji.

Table 2. **Effect of IIO bias in FL.** Classification accuracies for local and global models trained in an FL setting which has a biased client. For each model, the value in the left subcolumn corresponds to the test set of a biased client, and in the right subcolumn, there is an average value over the test sets of unbiased clients with standard deviation where possible. Performance is shown from **bad** to **good**.

MODEL	LMLP^b		GMGP^b	
TEST SET	BIASED	UNBIASED	BIASED	UNBIASED
ALL CLASSES	**85.8**	**76.1±1.7**	**83.3**	**75.3±1.2**
BIASED CLASS	**100.0**	**0.0**	**85.7**	**4.8±4.7**

Communication of prototypes with other clients brings slight improvement to the model's performance. GMGP^b has 4.76% accuracy for biased class on unbiased data and 85.71% on biased one. From Figure 2, we can see that the global model for a biased client does not activate the emoji as its local counterpart but still looks at a less informative patch close to it. Global model for good clients, however, looks at the bird's head as it does the local one. The distance between these prototypes can not only be visualised, but also computed, meaning that it could be used to derive a personalization weight which is visually interpretable.

The negative effect of data bias is additionally demonstrated in Appendix Table 3. Here, we show how different models predict a class for biased and unbiased test images.

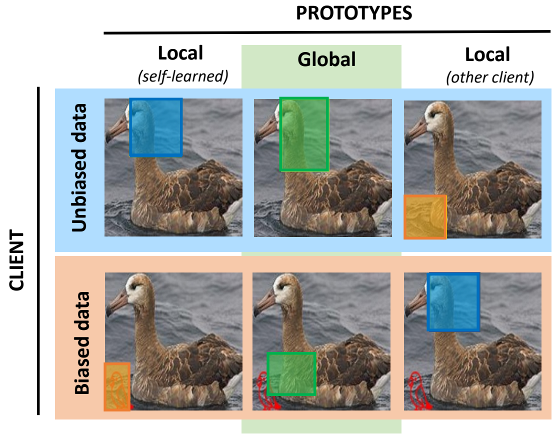


Figure 2. **Prototypes learned in IIO FL setting.** Examples of a test image with bounding boxes indicating the most activated patches by the prototypes learned locally and **globally** on **unbiased** and **biased** datasets in an IIO-FL setting.

5. Discussion

`inDISCO` is a novel extension of `ProtoPnet`, which allows interpretable and privacy-preserving identification, attribution, and quantification of data bias in federated learning. The bias can be visualized as well as quantified, which holds potential for creating a visually interpretable personalization scheme. Thus, `inDISCO` creates transparency from blackbox data without compromising privacy. Indeed, the only elements shared with the server are prototypes computed from a latent space representation. The shared elements can be tailored to various privacy budgets by setting parameters such as the number of prototypes per class, the size of a prototype, and the number of communication rounds.

References

Arivazhagan, M. G., Aggarwal, V., Singh, A. K., and Choudhary, S. Federated learning with personalization layers. *arXiv:1912.00818*, 2019.

Chen, C., Li, O., Tao, C., Barnett, A. J., Su, J., and Rudin, C. This looks like that: Deep learning for interpretable image recognition. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 8930–8941, Red Hook, NY, USA, 2019. Curran Associates Inc.

Chen, H., Lundberg, S., and Lee, S.-I. *Explaining Models by Propagating Shapley Values of Local Components*, pp. 261–270. Springer International Publishing, Cham, 2021. ISBN 978-3-030-53352-6. doi: 10.1007/978-3-030-53352-6_24. URL https://doi.org/10.1007/978-3-030-53352-6_24.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255. IEEE, 2009.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 4768–4777, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54. JMLR: W&CP, 2017.

Ribeiro, M. T., Singh, S., and Guestrin, C. "Why should I trust you?": Explaining the predictions of any classifier. KDD '16, pp. 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939778. URL https://doi.org/10.1145/2939672.2939778.

Roschewitz, D., Hartley, M.-A., Corinzia, L., and Jaggi, M. iFedAvg: Interpretable data-interoperability for federated learning. *arXiv:2107.06580*, 2021.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626, 2017. doi: 10.1109/ICCV.2017.74.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *ICML'17*, pp. 3145–3153. JMLR.org, 2017.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*, 2014.

Singh, A., Sengupta, S., and Lakshminarayanan, V. Explainable deep learning models in medical image analysis. *Journal of Imaging*, 6, 2020.

Wah, C., Branson, S., Welinder, P., Perona, P., and S., B. The Caltech-UCSD Birds-200-2011 dataset. Technical report, California Institute of Technology, 2011.

A. Appendix

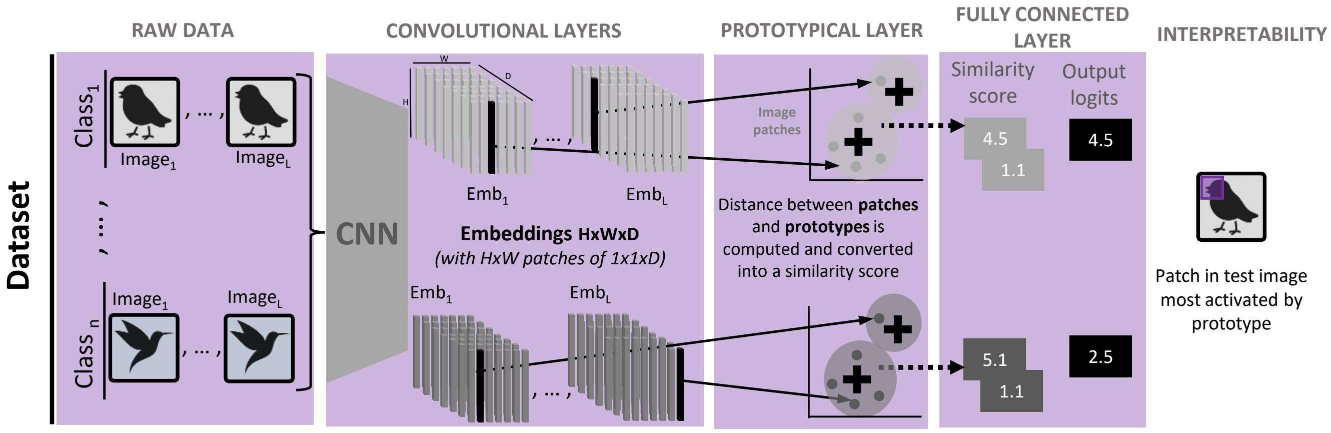
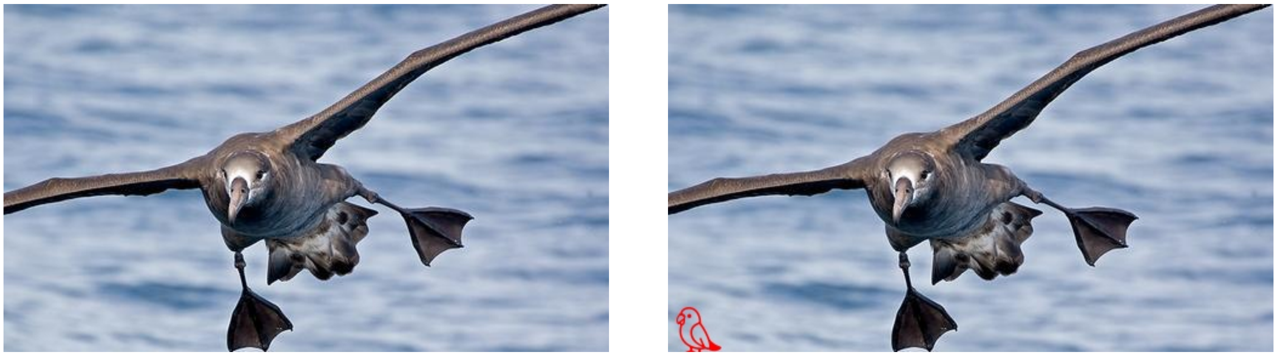


Figure 3. **ProtoPnet architecture**. This is a centralized setting with no clients. ProtoPnet, passes raw images through a CNN to create embeddings of size $[H \times W \times D]$ in the latent space $(Emb_1, ..., Emb_L)$, which are divided up into $H \times W$ image patches $[1 \cdot 1 \cdot D]$. Prototypical patches (**black**) are clustered and a prototype (**+**) is learned for each image. Classification is based on a similarity score computed between the prototype and the image patches. In the final panel, we see prototypes can be visualized to directly.



(a) Unbiased

(b) Biased

Figure 4. Imperfectly interoperable images used in this work. The biased client **(b)** has a red emoji of a parrot in the lower left corner.

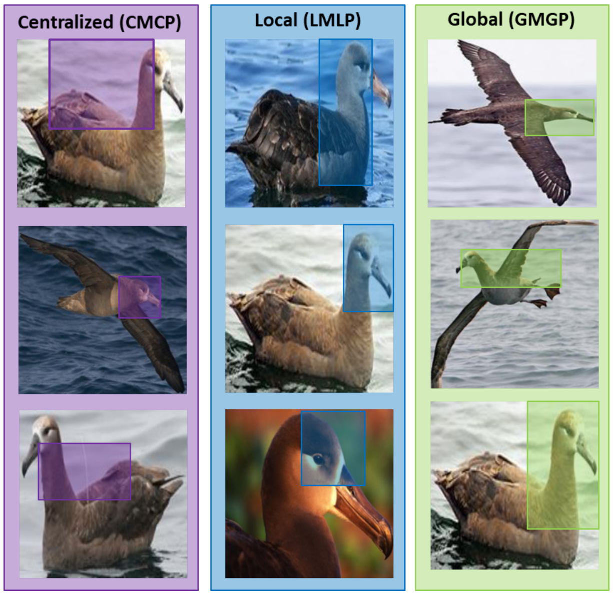


Figure 5. Examples of training images with bounding boxes indicating centralized, local, and global prototypes learned on unbiased data.

Table 3. Prediction results of local and global models for two biased and two unbiased test images coming from biased and unbiased during training classes.

Model	Biased class		Unbiased class	
	Test image			
	Unbiased	Biased	Unbiased	Biased
LMLP	COR	COR	COR	COR
LMLP^b	INCOR	COR	COR	INCOR
GMGP	COR	COR	COR	COR
GMGP^b	INCOR	COR	COR	INCOR

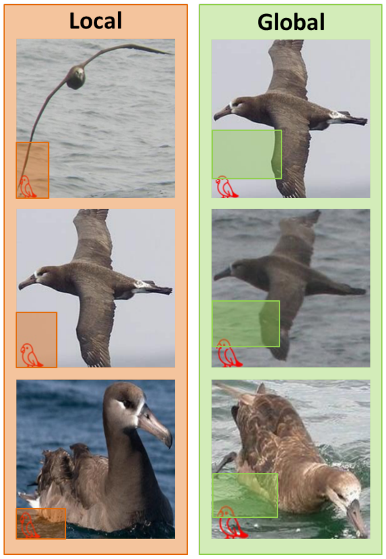


Figure 6. Examples of training images with bounding boxes indicating local and global prototypes learned on biased data.