Automation, Surgery Support and Intuitive 3D visualization to optimize workflow in image guided therapy SysTems

DELIVERABLE D2.4

State-of-the-art report on federated learning and hyperparameter optimization

Project number:	ITEA 20044
Document version no.:	v 1.0
Edited by:	Anders Eklund
Date:	2023.06.30

ITEA Roadmap challenge:

Smart Health

version #		
V0.1	2023.04.01	Starting version, template
V0.2	2023.06.29	Compilation of first input by LiU, Inovia, Scaleout, RaySearch, GCA Yazilim
V1.0	2023.06.30	Final version

Deliverable review procedure:

- **2 weeks before due date**: deliverable owner sends deliverable –approved by WP leader– to Project Manager
- **Upfront** PM assigns a co-reviewer from the PMT group to cross check the deliverable
- **1 week before due date**: co-reviewer provides input to deliverable owner
- **Due date:** deliverable owner sends the final version of the deliverable to PM and co-reviewer

TABLE OF CONTENTS

1 Abbreviations

AI	Artificial intelligence
API	Application programming interface
CIFAR	Canadian institute for advanced research
CNN	Convolutional neural network
CT	Computed tomography
FedAvg	Federated averaging
FedCostWAvg	Federated learning with cost based averaging
FedSA	Federated Staleness Aware
FL	Federated learning
GAN	Generative adversarial network
GDPR	General data protection regulation
HD	Hausdorff distance
IDA	Inverse distance aggregation
iid	independent and identically distributed
ML	Machine learning
MNIST	Modified national institute of standards and technology
MR	Magnetic resonance
MRI	Magnetic resonance imaging
MHAT	Model heterogenous aggregation training
PPFL	Privacy preserving federated learning

2 Executive summary

This document presents state-of-the-art methods for federated learning and hyperparameter optimization, focused on medical images. The document starts with an introduction to federated learning, and why it can be a solution to obtain large datasets to train deep learning models for medical applications. The next chapter covers different aggregation functions, i.e. how to combine updates from all nodes in the federation to create a new global model, which is the core of federated learning. This is followed by methods for data privacy, and methods for harmonizing images and annotations between nodes (hospitals). The final chapter considers hyperparameter optimization, both for general machine learning and for federated learning.

Document 2.1 in ASSIST focuses on other topics related to federated learning; data lakes, FL frameworks, configuration of FL in hospitals, legal aspects of FL in different countries, and how FL can be used for different uses cases in ASSIST.

3 Introduction to federated learning

This section will introduce federated learning, interested readers are referred to recently published papers about FL in health care for more information (Rieke et al., 2020; Antunes et al., 2022; Kairouz et al., 2021; Xu et al., 2021).

3.1 Why federated learning?

One could argue that "deep-learning" is the state-of-the-art for machine learning. Architectures such as deep CNNs or transformers typically outperform other ML methods on several established benchmarks. But these networks have two drawbacks; training is costly both in time and computation, and a huge amount of training data is required. No element is more essential in machine learning than high quality training data, and this is especially true for deep learning. The work involved in acquiring, labelling, and preparing training data is daunting. To collect and annotate a large high quality training set is especially difficult in medical imaging, as researchers and companies then need to follow more regulations compared to other types of data.

Medical data is sensitive and need to be anonymized before inclusion into any training set. GDPR regulations restrict this further, and the terms of agreement may prohibit sharing of the data. Different hospitals, regions and countries may have different rules for sharing data, even if they should all follow GDPR. In short, the creation of large medical image data sets is hard and time consuming.

Federated learning seems to be the obvious remedy to the data collection problem. The hospitals/clinics become nodes/clients in an asynchronous training network instead of being simple contributors of raw data. Model updates are shared instead of sharing data. This way, the images still become part of the training set, but the data is never shared between nodes, see Figure 1. See Figure 2 for a comparison of FL and centralized training. Another benefit of federated learning is that it is sufficient if each node in the federation uses a decent computer, instead of using a supercomputer for centralized training.

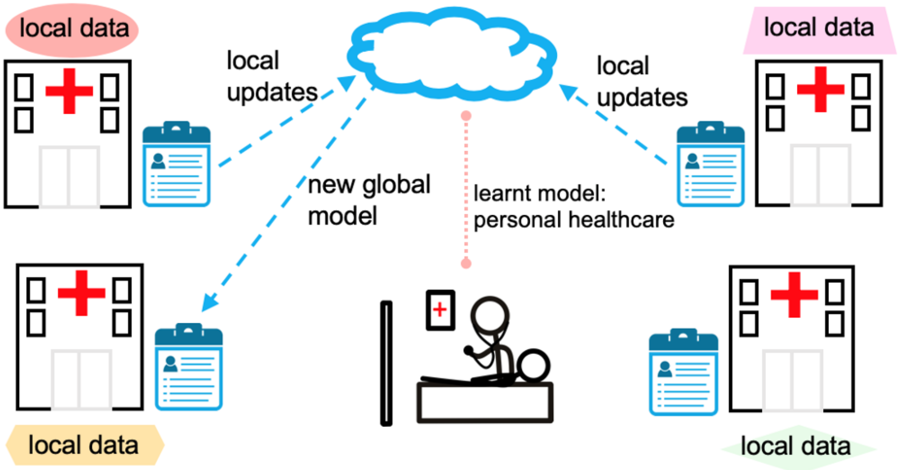


Figure 1. The main idea in FL is to not store all data in a single large, centralized database or data lake, but to instead store for example image data locally at each hospital. Instead of sending medical images and other medical data between the hospitals, the hospitals send updates, or parameters, of deep learning models. This

process is then iterated to convergence. Instead of having one large supercomputer, it is with federated learning sufficient if each hospital has a smaller computer.

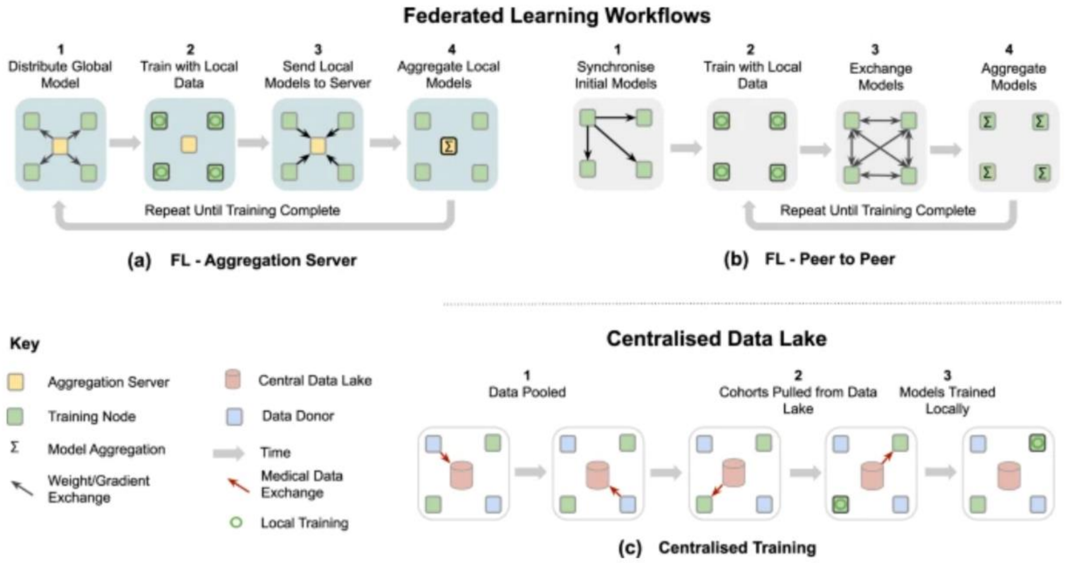
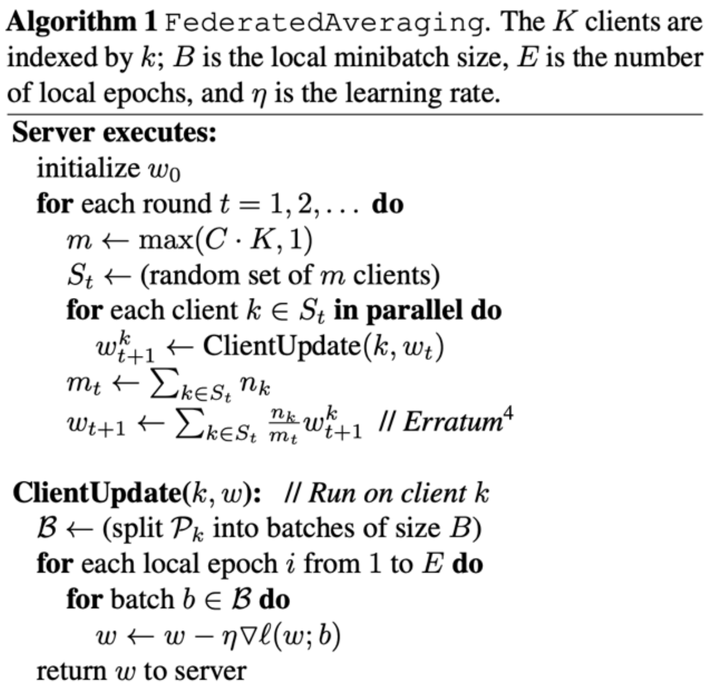


Figure 2. Image and figure text from (Rieke et al., 2020). A comparison of federated learning workflows and centralized training. a) FL aggregation server—the typical FL workflow in which a federation of training nodes receive the global model, resubmit their partially trained models to a central server intermittently for aggregation and then continue training on the consensus model that the server returns. b) FL peer to peer—alternative formulation of FL in which each training node exchanges its partially trained models with some or all of its peers and each does its own aggregation. c) Centralised training—the general non-FL training workflow in which data acquiring sites donate their data to a central computer from which they and others are able to extract data for local, independent training.

4 Aggregation methods and algorithms for federated learning

Federated learning is a decentralized machine learning technique which enables multiple nodes to collectively train a model without exposing their raw data. Aggregation methods are the core of federated learning, as they merge the locally obtained model updates from each node to create a new global model. There are many different aggregation methods as this is an active area of research. Simpler methods such as federated averaging and weighted federated averaging work well as long as the data is very similar at each node, while more advanced methods are required when the data are heterogeneous. Here, we will briefly cover federated averaging and inverse distance aggregation, FedGraph, federated staleness aware, and model heterogenous aggregation training.

Federated averaging: Federated averaging is a prominent aggregation method widely used in federated learning research and in different applications. The federated averaging algorithm, Figure 3, was introduced by researchers at Google in 2017 (McMahan et al, 2017). The purpose of developing this algorithm was to conduct collaborative training of deep neural networks on decentralized data, while still ensuring data privacy. To create a global model this aggregation method combines local model updates from multiple nodes. This is done by simply calculating the average update over all nodes/clients in the federation. This will work well as long as all nodes have a similar amount of data which is homogenous. However, in real-life applications it is common that some nodes have much more data than others (e.g. one hospital can provide data from 10 patients, while another hospital provides data from 100 patients). To compensate for this, weighted federated averaging can be used, where a weighted average is calculated instead of a standard average. The weight is then higher for nodes with more data, and this weight is for example calculated from the number of patients.



Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

```
Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $m_t \leftarrow \sum_{k \in S_t} n_k$
    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$   // Erratum[4]

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

Figure 3. The algorithm for federated averaging. Each node/client performs an update of the deep learning model using its local data, and then sends the updated weights or the gradient to the server (combiner), which calculates a global average, with or without a specific node weight assigned to compensate for an uneven distribution of data. The new global model is then sent out to all clients to continue the training.

Federated learning in non-iid data: Federated learning has been widely used in many industries for the prediction of words or detection of visual objects (Ma et al., 2022). FL can have multiple nodes with unequal distribution of data. Data heterogeneity is one of the key factors that needs to be considered when implementing federated learning in medical research, especially if FL is conducted between different countries. For example, different hospitals will have different imaging equipment (MR and CT scanners) that will produce images that look different. Furthermore, even if all patients have been diagnosed with the same disease, a disease may result in different symptoms depending on the ethnicity of the patient. While many FL researchers consider their data as iid (independent and identically distributed), this is not the case in many real-life scenarios (Ma et al., 2022). Therefore, we will here discuss more advanced aggregation algorithms which can handle non-iid data.

Inverse distance aggregation (IDA): Inverse distance aggregation (IDA) (Yeganeh et al., 2020), Figure 4, can be seen as an extension of weighted federating averaging, where the weight of each node is instead calculated according to

$$\alpha_k = \frac{1}{Z} \| \omega_{Avg}^{t-1} - \omega_k^{t-1} \|^{-1}$$

where Z is a normalization factor. This approach will put a lower weight on nodes which produce a model with parameters which are far away from the average model. IDA also uses the training accuracy of each node to obtain the final weight, to penalize overfitted nodes and encourage under-trained nodes. Experiments demonstrate that this approach works better than weighted federated averaging for two datasets.

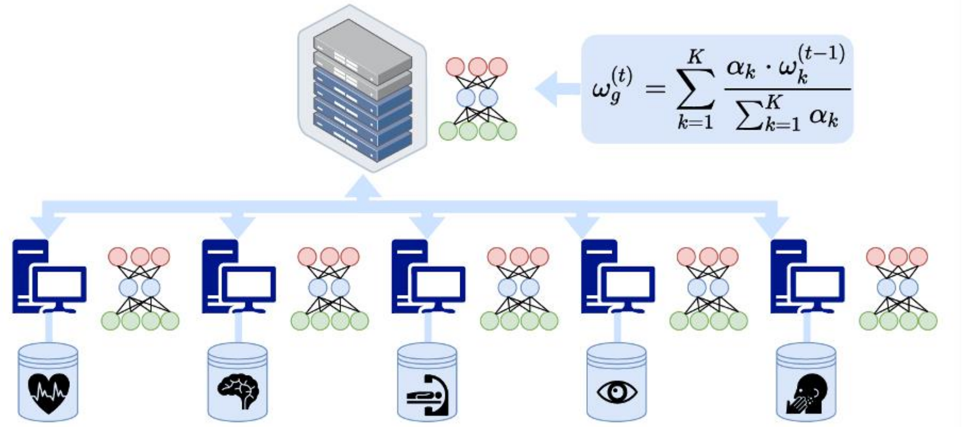

$\omega_g^{(t)} = \sum_{k=1}^{K} \frac{\alpha_k \cdot \omega_k^{(t-1)}}{\sum_{k=1}^{K} \alpha_k}$

Figure 4. Federated learning with non-iid data is a more difficult problem which requires more advanced aggregation methods. In this figure it is illustrated that each node has data from a different distribution. The weight for each node in IDA is

calculated according to the distance between the model parameters at that node, and the global model parameters, so that “distant” models contribute less.

Federated cost weighted averaging (FedCostWAvg): Mächler et al. (2021) propose another way to aggregate the updates from all nodes, which includes the amount by the cost function decreased during the last step. The new global model is calculated according to

$$M_{i+1} = \sum_{j=1}^{n} (\alpha \frac{s_j}{S} + (1-\alpha) \frac{k_j}{K}) M_i^j$$

$$k_j = \frac{c(M_{i-1}^j)}{c(M_i^j)}, K = \sum_j k_j$$

where c(M) returns the cost of model M and alpha is a hyperparameter. This approach will not only adjust for training data size, but also for the size of the local improvements that were made during the last round. A node which only marginally improved the cost will influence the global update less compared to nodes that made a larger improvement. Figure 5 illustrates a comparison with weighted federated averaging for brain tumour segmentation, showing a clear improvement.

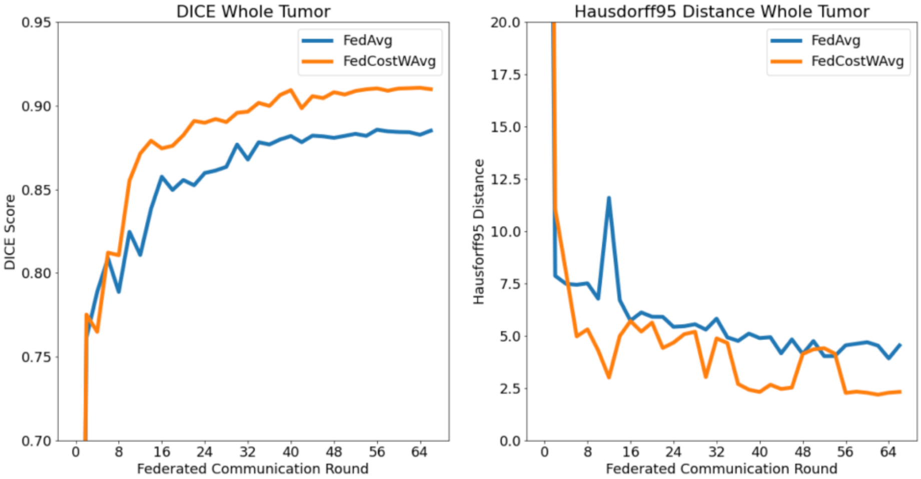


Figure 5. Comparing weighted FedAvg and FedCostWAvg for brain tumor segmentation, showing an improvement when using FedCostWAvg.

FedGraph: To deal with non-iid data, FedGraph (Deng et al., 2022) is another approach which uses a combination of three factors to obtain a weight for each node, which is updated during training. The three factors are; the proportion of each local dataset (as in weighted federated averaging), the topology factor of model graphs, and the model weights of each node (similar to IDA), see Figure 6.

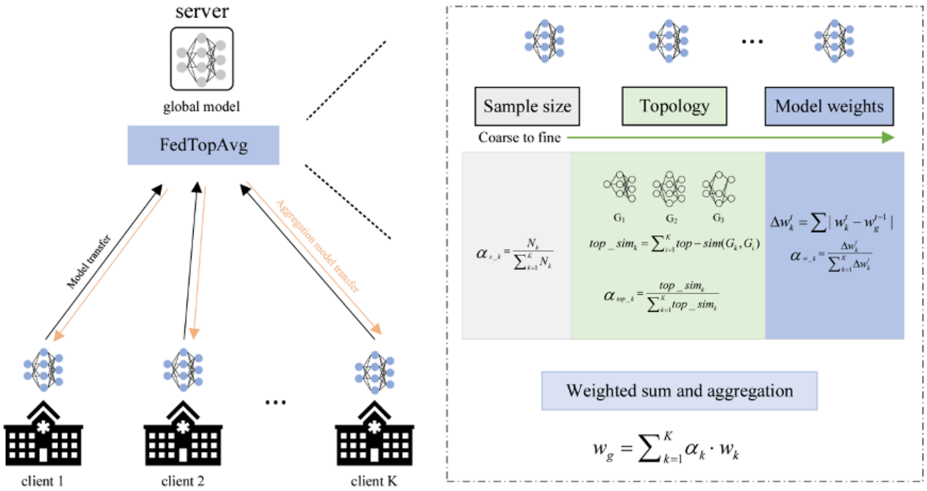
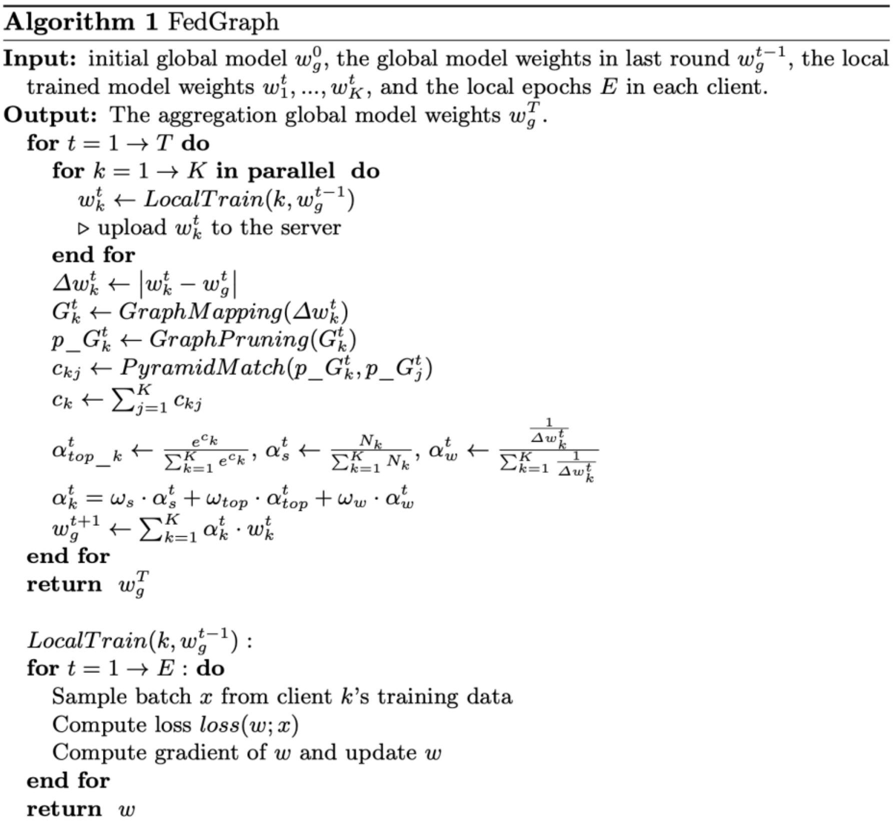

server
global model
FedTopAvg
Model transfer
Aggregation model transfer
client 1
client 2
client K
...
Sample size
Topology
Model weights
Coarse to fine
$\alpha_{s\_k} = \frac{N_k}{\sum_{k=1}^{K} N_k}$
G₁
G₂
G₃
$top\_sim_k = \sum_{i=1}^{K} top-sim(G_k, G_i)$
$\alpha_{top\_k} = \frac{top\_sim_k}{\sum_{k=1}^{K} top\_sim_k}$
$\Delta w_k^t = \sum | w_k^t - w_g^{t-1} |$
$\alpha_{w\_k} = \frac{\Delta w_k^t}{\sum_{k=1}^{K} \Delta w_k^t}$
Weighted sum and aggregation
$w_g = \sum_{k=1}^{K} \alpha_k \cdot w_k$

Figure 6. Overview of the FedGraph aggregation algorithm (Deng et al., 2022), which uses a combination of three weights (based on sample size, topology and model weights) to obtain a weight for each node. These node weights are updated during training.

The FedGraph algorithm is given in Figure 7. The sample size weights are constant throughout the training, while the topology weights and the weight from the model parameters are updated in each round. Figure 8 demonstrates that this approach leads to higher Dice scores and lower Hausdorff distance for federated brain tumor segmentation, compared to federated averaging and FedCostWAvg.

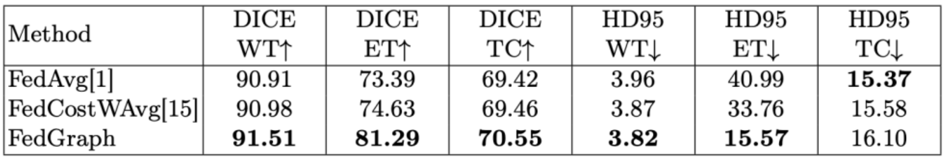


Algorithm 1 FedGraph

```
Input: initial global model $w_g^0$, the global model weights in last round $w_g^{t-1}$, the local
  trained model weights $w_1^t, ..., w_K^t$, and the local epochs $E$ in each client.
Output: The aggregation global model weights $w_g^T$.
  for $t = 1 \to T$ do
    for $k = 1 \to K$ in parallel do
      $w_k^t \leftarrow LocalTrain(k, w_g^{t-1})$
      ▷ upload $w_k^t$ to the server
    end for
    $\Delta w_k^t \leftarrow |w_k^t - w_g^t|$
    $G_k^t \leftarrow GraphMapping(\Delta w_k^t)$
    $p\_G_k^t \leftarrow GraphPruning(G_k^t)$
    $c_{kj} \leftarrow PyramidMatch(p\_G_k^t, p\_G_j^t)$
    $c_k \leftarrow \sum_{j=1}^{K} c_{kj}$
    $\alpha_{top\_k}^t \leftarrow \frac{e^{c_k}}{\sum_{k=1}^{K} e^{c_k}}$, $\alpha_s^t \leftarrow \frac{N_k}{\sum_{k=1}^{K} N_k}$, $\alpha_w^t \leftarrow \frac{\frac{1}{\Delta w_k^t}}{\sum_{k=1}^{K} \frac{1}{\Delta w_k^t}}$
    $\alpha_k^t = \omega_s \cdot \alpha_s^t + \omega_{top} \cdot \alpha_{top}^t + \omega_w \cdot \alpha_w^t$
    $w_g^{t+1} \leftarrow \sum_{k=1}^{K} \alpha_k^t \cdot w_k^t$
  end for
  return $w_g^T$

  $LocalTrain(k, w_g^{t-1})$ :
  for $t = 1 \to E$ : do
    Sample batch $x$ from client $k$'s training data
    Compute loss $loss(w; x)$
    Compute gradient of $w$ and update $w$
  end for
  return $w$
```

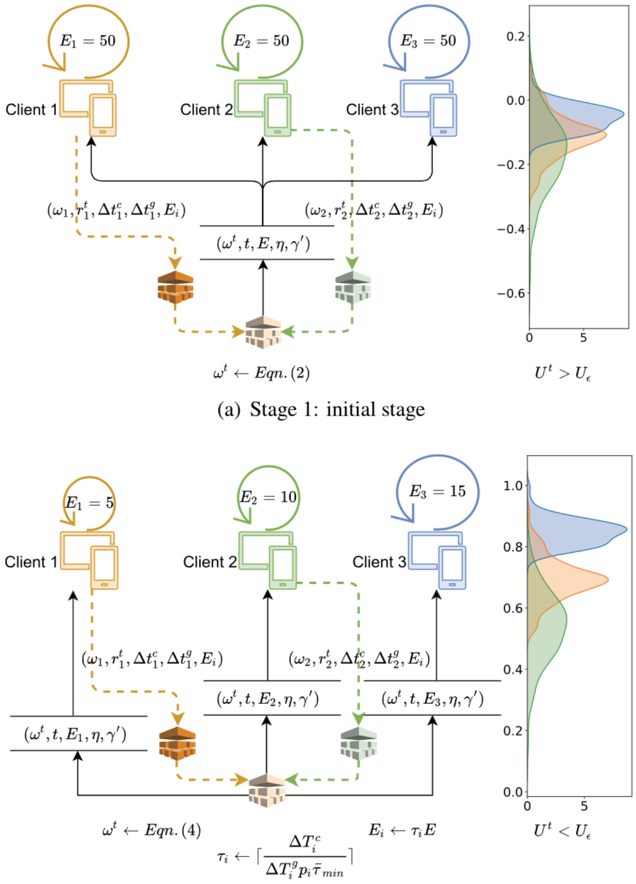
Figure 7. The FedGraph algorithm proposed by Deng et al. (2022), where a weight for each node is calculated in each round of the federated learning.



Method	DICE WT↑	DICE ET↑	DICE TC↑	HD95 WT↓	HD95 ET↓	HD95 TC↓
FedAvg[1]	90.91	73.39	69.42	3.96	40.99	**15.37**
FedCostWAvg[15]	90.98	74.63	69.46	3.87	33.76	15.58
FedGraph	**91.51**	**81.29**	**70.55**	**3.82**	**15.57**	16.10

Figure 8. Comparison of Dice scores and Hausdorff distance for federated brain tumor segmentation, when using different aggregators (Deng et al. 2022). FedGraph clearly performs best.

Federated Staleness – Aware (FedSA): Another issue in real case federated learning deployments is environment heterogeneity, i.e. unreliable connections and that the computer resources are limited for some nodes. This leads to that some nodes become inactive or slow (Chen et al., 2021). This eventually degrades the performance of global model, and the issue is known as “staleness effect”. To reduce this effect, asynchronous federated learning can be used (Chen et al., 2021). The overall process of FedSA is achievable by two stage training. Through this approach, it dynamically chooses proper hyperparameters by considering the heterogeneity in devices and the similarities among local models. At the initial stage, an arbitrary large number of epochs is used for each node to accelerate training with less communication. The second stage is the convergence stage where researchers carefully choose a smaller number of epochs by calculating the staleness of each node. See Figure 9 for an illustration of the process. Experiments show a much better convergence when a large proportion of nodes are stale.


$E_1 = 50$
$E_2 = 50$
$E_3 = 50$
Client 1
Client 2
Client 3
$(\omega_1, r_1^t, \Delta t_1^c, \Delta t_1^g, E_i)$
$(\omega_2, r_2^t, \Delta t_2^c, \Delta t_2^g, E_i)$
$(\omega^t, t, E, \eta, \gamma')$
$\omega^t \leftarrow Eqn. (2)$
$U^t > U_\epsilon$
0.2
0.0
−0.2
−0.4
−0.6
0
5
$E_1 = 5$
$E_2 = 10$
$E_3 = 15$
$(\omega^t, t, E_1, \eta, \gamma')$
$(\omega^t, t, E_2, \eta, \gamma')$
$(\omega^t, t, E_3, \eta, \gamma')$
$\omega^t \leftarrow Eqn. (4)$
$E_i \leftarrow \tau_i E$
$\tau_i \leftarrow \lceil \frac{\Delta T_i^c}{\Delta T_i^g p_i \bar{\tau}_{min}} \rceil$
$U^t < U_\epsilon$
1.0
0.8
0.6
0.4
0.2
0.0

(a) Stage 1: initial stage

(b) Stage 2: convergence stage

Figure 9. An overview of FedSA (Deng et al., 2021). At the initial stage a large number of epochs are used at each node, to decrease communication. In the second stage, a smaller number of epochs are used, which is based on the staleness of each node.

Model Heterogenous Aggregation Training (MHAT): In traditional federated learning training only the model parameters are considered, which can make the process slow and require quite a lot of communication (Hu et al., 2021). Also, in traditional federated learning training it is not possible to use different model architectures across the nodes. Hu et al. (2021) therefore proposed a knowledge distillation approach which collects information from each node, and trains an auxiliary model on the server to gather and combine the updated information from each node. Here the goal is to make the aggregation more sufficient by giving participants the freedom of designing their own model architecture. An illustration of the MHAT framework is given in Figure 10.

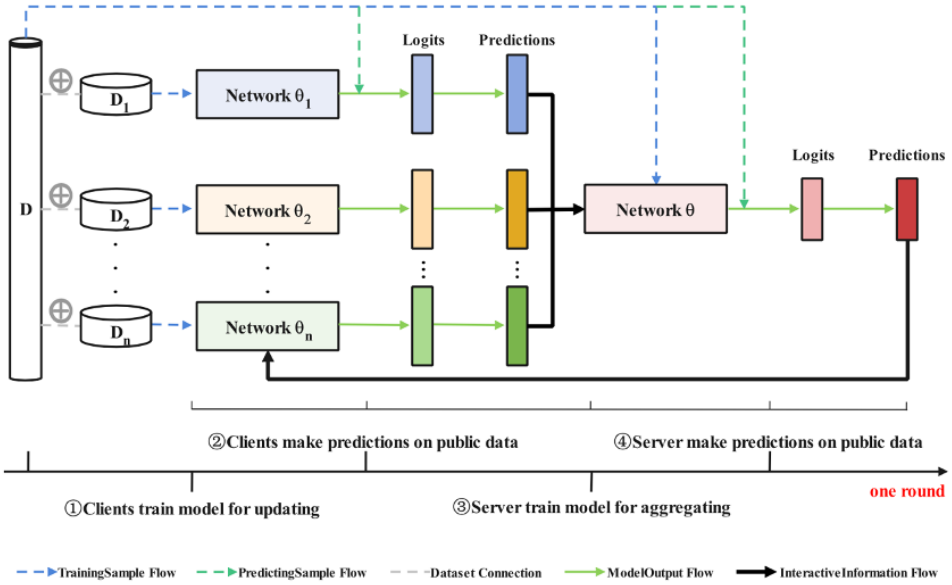


Figure 10. Illustration of the MHAT framework proposed by Hu et al. (2021).

As an auxiliary model is trained on the server to have better understanding on the information aggregation, thus it improves the aggregation result, model convergence speed, and reduce interaction between the server and clients~~.~~

5 Privacy methods and algorithms for federated learning

Federated learning has its unique set of vulnerabilities and potential risks which need to be considered. Even if no medical images are sent between the nodes, it has been shown that it is possible to recover the training images from the weights of a trained CNN (Haim et al., 2022). In Sweden it is currently discussed if (the weights of) a machine learning model trained with medical data should be considered personal data. Therefore, privacy methods and algorithms play a crucial role for ensuring the data confidentiality in FL. Under this section, several privacy methods which have been widely used in the federated learning will be discussed.

Privacy preserving federated learning (PPFL) is designed to ensure the confidentiality of the data during the aggregation process in federated learning. Researchers and organizations have been working collaboratively to develop methods for ensuring the privacy of the individuals while doing decentralized training. PPFL can be divided into the following categories – encryption based PPFL, perturbation based PPFL, anonymization based PPFL and hybrid PPFL (Yin et al., 2021).

Encryption based PPFL is just another extension of commonly used encryption methods. For this method, patient's data or defence information, or weights of a network, are encrypted before it is shared with others. There are three types of encryptions based PPFL – homomorphic encryption, secure multi-party communication and secret sharing based encryption. Encrypted data-oriented computations are enabled by homomorphic encryption, whereas secure multi-party computation enables computing functions over the private inputs. In both cases they preserve privacy, but the underlying mechanisms are different from each other. On the other hand, secret sharing privacy preserving federated learning is a cryptographic technique where the data is fragmented and divided between multiple users. It ensures that not a single user has access to the whole dataset. The complete dataset can only be reconstructed when a sufficient amount of data is combined (Yin et al., 2021). This approach ensures the highest protection of data while conducting the decentralized training.

Perturbation based PPFL randomly adds noise or changes the data or model weights before sharing it with others. This method is divided into 4 different categories – global differential privacy, local differential privacy, additive perturbation and multiplicative perturbation. Under global differential privacy, the combiner assigns a random number to participant for global training. Nodes update their local model and send weights back to the global server, but random Gaussian noise is added to the weights to prevent data leakage (Abadi et al., 2016, Yin et al., 2021). On the other hand, for local differential privacy, participants have more control own their own dataset. They can add random noise, or assign values to their dataset before sharing it with others for federated learning. In summary, both of these methods provide privacy in FL; however, the control mechanism is different.

Anonymization based PPFL is a widely used privacy method. Under this approach, the personal information, for example – date of birth, name or personal identity number are anonymized in such a way so that individuals can't be identified from the dataset. Once the data is anonymized, it can be used for training the global model. This approach ensures better data privacy and model performance than differential privacy-based FL method (Choquette-Choo et al., 2021).

Using different privacy methods has its own merits and demerits, adding random noise to the weights will for example lead to worse performance of the global model. To balance out data privacy and data utility, researchers proposed hybrid based PPFL. The goal of this method is to combine the top attributes from each privacy method and create a better approach which will degrade performance as little as possible. For example – researchers combined differential privacy with secure multiparty computation methods (Truex et al., 2019). As a result, it could reduce the effect of noise injection even when the number of participants increases.

To ensure the privacy of the data, Bonawitz et al. (2017) introduced secure aggregation for FL. Secure aggregation provides a protocol for protecting the data during the aggregation methods. This system is designed to mitigate unauthorized access of data; data leaking from the client end. By implementing this system in federated learning, researchers can ensure the data privacy and preserve the integrity of sensitive information throughout the process, see Figure 11.

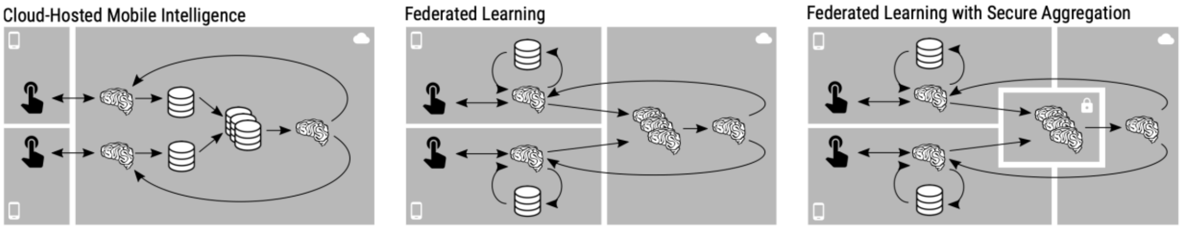

Cloud-Hosted Mobile Intelligence
Federated Learning
Federated Learning with Secure Aggregation

Figure 11. Difference between cloud hosted mobile intelligence, federated learning and federated learning with secure aggregation (Fereidooni et al., 2021).

6 Output privacy and federated machine learning

Federated machine learning (FL) has risen as a key method for ensuring privacy in machine learning by eliminating the requirement to centralize data. This innovation enhances input privacy and addresses the duplication issue prevalent in machine learning. There's a significant amount of ongoing research in FL, primarily targeting computational aspects like model consolidation, resource distribution, and system diversity. Security and privacy implications are equally significant focus areas, with the goal of maintaining the integrity of federated machine learning systems in a complex, decentralized training environment.

Examining the security and privacy obligations of federated learning structures, it's important to differentiate between input privacy and output privacy, see Figure 12. Input privacy is associated with the forward training process, while output privacy deals with what information about the input data can be inferred from the trained model. This form of privacy is a concern for any machine learning model that provides predictions to end-users. Malicious attempts to steal models or violate output privacy are often classified as reverse engineering attacks.

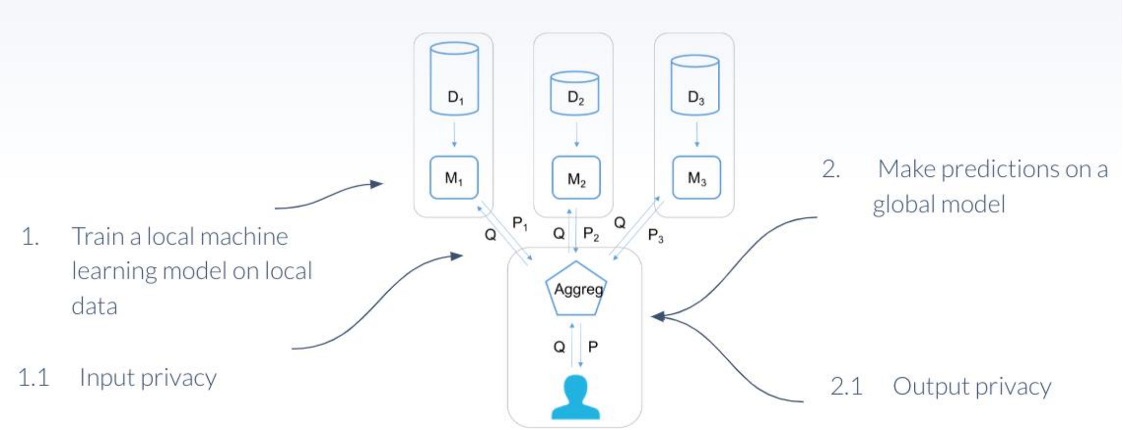


Figure 12. An overview of federated learning and the differences between input and output privacy.

6.1 Model reverse engineering

The process of reverse engineering a model entails acquiring its parameters and architectural specifics when only a black box is available. Two primary types of attempts can be identified: "model stealing" and "model inverting". The objective of model stealing is to duplicate a model's internal structure and predictive capability. Conversely, model inversion attacks are designed to rebuild training data or discern characteristics of that data. Various research initiatives have shown successful attacks that extract not just parameters but also the original training data (Tramer et al., 2016, Oh et al., 2019, Usynin et al., 2022).

Model extraction attacks, an instance of model stealing, involves an adversary aiming to clone machine learning models through a series of queries. In one study (Tramer et al., 2016), it was discovered that by examining the responses obtained from a sequence of queries to a deep neural network service's API, it became feasible to deduce the hyperparameters of deep neural networks hosted on well-known platforms such as Big ML and Amazon ML on AWS services.

The majority of adversarial machine learning literature presumes a centralized environment where access to both data and computation is provided as a single coherent unit. This assumption, however, falls apart in the context of federated training and inference processes. It gives rise to a nuanced relationship between input and output privacy in federated settings.

6.2 Output privacy in the context of federated machine learning

Output privacy breaches occur when a trained model is violated, leading to the exposure of information about the training data. An attacker, by gaining access to the model, can attempt to reverse engineer characteristics of the input data, or even reconstruct entire data points from the training data. The type of information extracted through reverse engineering can vary from high-level aggregated information, like the overall distribution of training contributions, to precise details about the training data such as a specific image within the training dataset. Such risks are not exclusive to federated machine learning systems, but are a concern for any machine learning model that is accessible to end-users, whether as an API or in other forms.

However, the distributed nature of federated learning adds an extra layer of complexity to the training process, necessitating a clear understanding of the differences between output breaches in centralized and federated systems. For instance, research has indicated that attacks significantly impacting centralized models might have minimal effects on federated models (Shejwalkar et al., 2022). In a centralized scenario, if a trained model is breached, reverse engineering could potentially reveal not just the training data's distribution but also the actual data and labels used for training. In the context of federated learning, it becomes crucial to identify whether an attacker can obtain knowledge about aspects like the number of clients participating in training, their identities, individual local data distributions, or individual client records. The exposure of such information in a federated setting can have implications not seen in centralized training and inference methods, such as potential harm to a participating organization's reputation.

From a technical perspective, managing output privacy involves addressing statistical disclosure, model inversion, and membership inference attacks. The primary attack categories are targeted, backdoor, and untargeted attacks (Bhagoji et al., 2019, Jere et al., 2020). All these attacks occur during the training phase, falling within the realm of input privacy. To diminish the impact of these attacks on model inference, which pertains to output privacy, it's essential to comprehend the link between input and output privacy, especially in the context of federated learning. By bolstering input privacy, we can improve output privacy, thereby obstructing the reverse engineering process and protecting individual client data.

6.3 Practical feasibility of attacks on federated learning

systems

When reflecting on the practical facets of reverse engineering, it's crucial to recognize the assumptions typically made in research studies. Simplifying assumptions are often applied to the training setup or machine learning model to facilitate theoretical development and qualitative understanding. Furthermore, many studies rely on benchmark datasets such as MNIST and CIFAR-10, which might lead to an overstatement of risks.

Google researchers have recently pointed out that some studies on privacy risks in federated learning propose scenarios that are far from realistic. For instance, certain attack scenarios suggest up to 25% of clients in a federation could be compromised. While this might seem feasible from a research viewpoint, it is a highly improbable situation in reality. Google disclosed that GBoard, heavily reliant on federated learning, is used on a billion mobile phones, making 25% compromised clients equivalent to 250 million compromised mobile phones. If that were the case, system developers would be grappling with issues far greater than a compromised ML training process.

Nvidia researchers recently focused on gradient inversion attacks on federated learning systems in a cross-silo context (Hatamizadeh et al., 2023). Gradient inversion, an attack type that could target FL systems, capitalizes on potential access to gradients (or weights) exchanged between clients and servers. The intent is to create a generative model that can replicate data similar to the training input data. The researchers performed meticulous experiments to reverse-engineer chest X-ray images under various conditions, concluding that the risks of gradient inversion in FL are likely overemphasized in prior work, assuming that federated training employs realistic batch sizes, training image numbers at each client site, and a sensible quantity of local iterations. Additionally, they noted that several "low-hanging fruits" could be effortlessly integrated into an FL system to render attacks useless.

Further research is required using realistic machine learning scenarios and production-grade FL implementations to gain a deeper understanding of the practical costs and constraints of different attacks. Still, numerous qualitative insights can be derived from studies on benchmark problems in controlled settings. For instance, we understand that the risk of model inversion increases with:

- Limited training data points at a client site
- Smaller batch sizes
- Fewer local iterations before aggregation

These are all factors that the owner or developer of the machine learning model can control. Being aware of these risks assists in organizing and managing the federated training process. Furthermore, for a gradient inversion attack to be successful, an adversary needs access to:

- Local gradient/parameter updates from clients
- The current state of the global model that updates were calculated from
- The model architecture

There are several relatively simple measures that can, and should, be implemented to lessen the risks of this information leaking outside the federation.

6.4 Ways to increase security and privacy of (federated) machine learning systems

Combining federated learning with other privacy-enhancing technologies. Techniques like multi-party computations and homomorphic encryption can be employed to introduce secure aggregation, providing an extra layer of security against the exposure of weights or gradients transmitted to the aggregator (Truex et al., 2019). However, there are issues related to performance and scalability. Another option is differential privacy, where intentional noise is introduced during local training or weight aggregation, to diminish the risk of reverse engineering. One drawback to this approach is the potential decrease in final model accuracy, necessitating a balance between noise and model accuracy (Wei et al., 2020, Truex et al., 2019).

Regularization. Employing regularization methods during loss computation and the aggregation process can enhance output privacy. Regularization bolsters anonymization without undermining model accuracy. Various studies have underscored the advantages of integrating regularization terms. For instance, a recent study by T. Wang and his team elaborated on the effects of regularization on model inversion attacks (Wang et al., 2021).

Engineering solutions. Beyond solutions specific to machine learning, data engineering solutions can also fortify output privacy. One such approach involves the early assessment of incoming model inference requests. This could mean implementing rate limits, and effectively scanning for any unusual behaviour. This would allow for the interception of prediction requests showing suspicious patterns. To secure the model-serving environment, it's important to include authentication, authorization, and proxy/token-based access to inference in the platform. Additional security measures within the CI/CD pipeline can help avoid unintended model exposure. Specifically for federated learning, it's crucial to employ secure, industry-standard communication protocols, and enforce client identity management. In a subsequent post, we will provide more details on system security for federated learning.

7 Harmonizing images and annotations

This section considers the general problem of non-iid data, focusing on that images, annotations and dose plans for radiotherapy differ between hospitals.

7.1 Methods for harmonizing images between nodes

MR and CT scanners at different hospitals will produce images that look different, which is a major problem when training deep learning models, and especially for federated learning. To some extent this can be solved with data augmentation, for example through applying random changes of the image contrast or by adding random noise, when training a network to perform classification or segmentation.

In FeTS (federated tumor segmentation challenge) and BraTS (brain tumor segmentation challenge) the same preprocessing script is applied to the MR volumes of all subjects, to force all volumes to have the same resolution, to register them to the same coordinates (using a brain atlas), to apply bias field intensity correction, and to segment the brain from the head volume (Pati et al., 2022). Even after this preprocessing it is rather easy to see that brain images from one site look different compared to other sites.

More advanced harmonization approaches are based on deep learning, especially using generative adversarial networks (GANs) such as CycleGAN (Zhu et al., 2017). These GANs are trained in an unsupervised manner, by simply showing MR or CT images from two different scanners, and the GAN will learn to translate an MR image from one scanner to look like an MR image from another scanner. This two-domain approach can be extended to translate images from many scanners / sites into a single type of image, for example using techniques such as StarGAN (Choi et al., 2018). Figure 13 shows how a GAN can be used to translate between MR scanners (Bashyam et al., 2022) and Figure 14 shows how harmonization using StarGAN helps when training with one dataset and testing on other datasets (Bashyam et al., 2022). However, in a federated setting all the datasets will not be available in a single computer. It is therefore necessary to train the GAN in a federated manner, or solve this problem in some other way. Federated training of GANs is however not yet very common (Song & Ye, 2021).

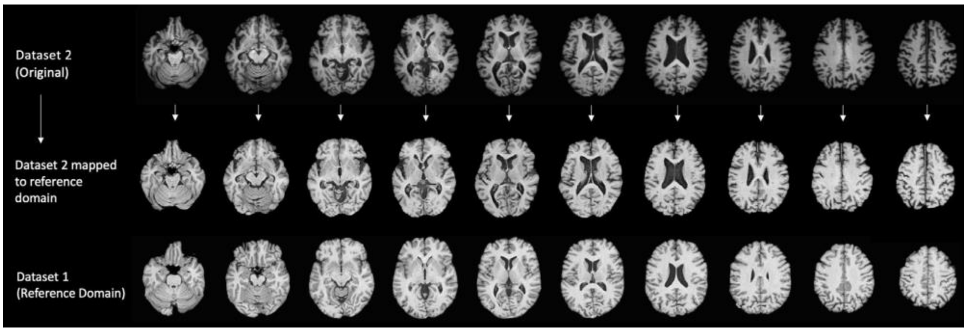
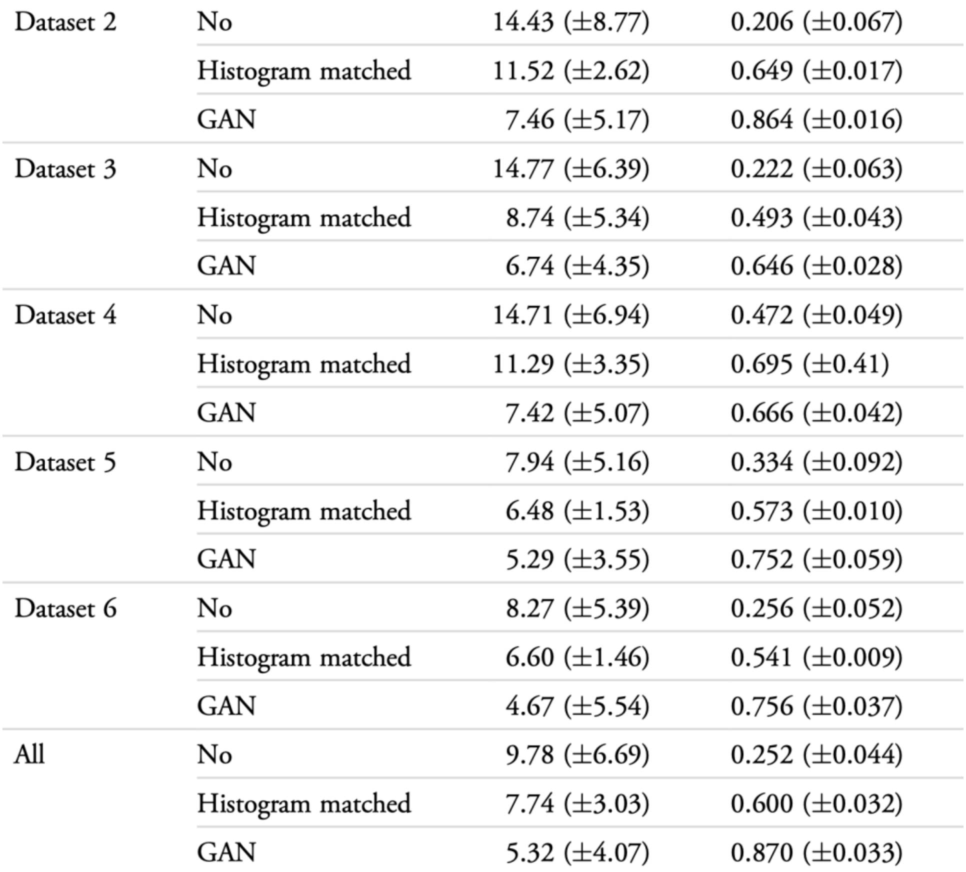


Figure 13. Harmonization of MR volumes from different MR scanners through deep learning (Bashyam et al., 2022). Top: a volume from MR scanner 1. Middle: The same volume translated to appear like a volume from MR scanner 2. Bottom: An MR volume from MR scanner 2.



Dataset 2	No	14.43 (±8.77)	0.206 (±0.067)
	Histogram matched	11.52 (±2.62)	0.649 (±0.017)
	GAN	7.46 (±5.17)	0.864 (±0.016)
Dataset 3	No	14.77 (±6.39)	0.222 (±0.063)
	Histogram matched	8.74 (±5.34)	0.493 (±0.043)
	GAN	6.74 (±4.35)	0.646 (±0.028)
Dataset 4	No	14.71 (±6.94)	0.472 (±0.049)
	Histogram matched	11.29 (±3.35)	0.695 (±0.41)
	GAN	7.42 (±5.07)	0.666 (±0.042)
Dataset 5	No	7.94 (±5.16)	0.334 (±0.092)
	Histogram matched	6.48 (±1.53)	0.573 (±0.010)
	GAN	5.29 (±3.55)	0.752 (±0.059)
Dataset 6	No	8.27 (±5.39)	0.256 (±0.052)
	Histogram matched	6.60 (±1.46)	0.541 (±0.009)
	GAN	4.67 (±5.54)	0.756 (±0.037)
All	No	9.78 (±6.69)	0.252 (±0.044)
	Histogram matched	7.74 (±3.03)	0.600 (±0.032)
	GAN	5.32 (±4.07)	0.870 (±0.033)

Figure 14. Comparison of harmonization approaches for the task of brain age prediction (Bashyama et al., 2022). A brain age prediction model was trained on dataset 1, and then tested on 5 other datasets (not included in training). The two right columns show mean absolute error in predicted brain age, and correlation between real and predicted age. Clearly, no harmonization of the MR images leads to poor prediction results. Matching the histograms helps a bit, while deep learning based harmonization using a GAN works much better.

7.2 Methods for harmonizing annotations between nodes

A challenge when developing deep learning models for image segmentations is noisy labels, i.e., inconsistent contours in the training/test/validation datasets. There are two main sources to this noise, one is the segmentation guidelines used and the other is interpersonal variability. Within a clinic, there is typically an alignment on guidelines used and their interpretation (Scoccianti et al., 2015). In a federated setting, there is a risk that guidelines and/or interpretation of guidelines differ between the involved clinics (Sylolypavan et al., 2023).

In order to succeed with federated learning for image segmentation, there should be an alignment before starting in terms of guidelines used as well as their interpretation. Ideally, a few cases from each node should be compared qualitatively as an initial step and there should be an agreement in the ground truth between the involved clinics. In real-life scenarios this may be difficult, as creating large datasets is often done using images with existing annotations. One possible solution to this problem is to put a lower certainty at the border of each annotated object, such that models are not penalized for making errors at the border (where the annotations are most likely to differ between hospitals). It is in theory possible to use methods like CycleGAN, mentioned in the previous section, to also harmonize the annotations. This is especially true if the MR images and the corresponding annotations are harmonized at the same time, as multi-channel images or volumes. However, this is difficult as all the data are not available at one computer, and one must resort to federated training of these models or other solutions.

7.3 Harmonizing treatment plans between nodes

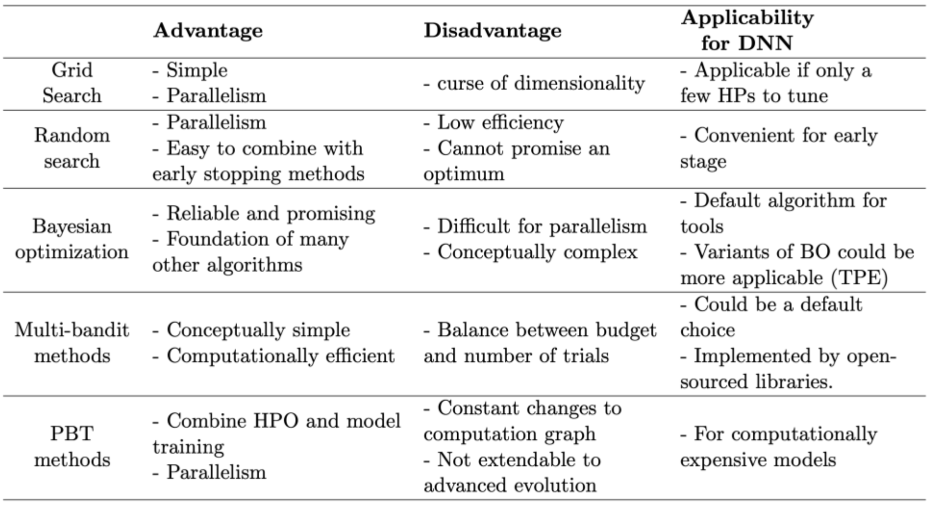
Similar to image segmentation, it is important that all treatment plans used for training a dose prediction model (from the segmentations) are aligned in terms of treatment protocol including treatment modality, prescribed dose level, delivery technique, and delivery machine. It therefore makes sense to select a widely used protocol and involve clinics using this selected protocol in the federated learning process. Before starting, it is recommended to qualitatively and quantitatively compare a few examples of dose distributions between the nodes to ensure the variation is acceptable.

There is typically a larger amount of flexibility in the post-processing of dose prediction models than image segmentation models, so while it is recommended to compare a few cases per clinic qualitatively before starting it is not as critical as for federated learning for image segmentation.

8 Hyperparameter optimization

8.1 General hyperparameter optimization

Hyperparameters are parameters that are set by the machine learning engineer, as opposed to the parameters that are learned by the model itself. Some examples of hyperparameters are as follows; number of hidden layers, number of neurons, the number of training epochs, activation functions, learning rate, input and hidden layer dropout values, batch size, loss functions, etc. Usually, the task of deciding these parameters requires both extensive trial-and-error, and machine learning expertise. Hyperparameter optimization, is the task of using algorithms like grid search, random search, and Bayesian methods to test combinations of pre-defined parameters and finding the optimal combination(s) of these parameters. Most neural architecture search methods use the same set of hyperparameters for all candidate architectures during the whole search stage; thus, after finding the most promising neural architecture, it is necessary to redesign a hyperparameter set and use it to retrain or fine-tune the architecture. Some hyperparameter optimization methods (such as Bayesian optimization and random search) have also been applied in neural architecture search. See Figure 15 for a list of common hyperparameter optimization algorithms (Yu & Zhu, 2020).



	Advantage	**Disadvantage**	**Applicability for DNN**
Grid Search	- Simple - Parallelism	- curse of dimensionality	- Applicable if only a few HPs to tune
Random search	- Parallelism - Easy to combine with early stopping methods	- Low efficiency - Cannot promise an optimum	- Convenient for early stage
Bayesian optimization	- Reliable and promising - Foundation of many other algorithms	- Difficult for parallelism - Conceptually complex	- Default algorithm for tools - Variants of BO could be more applicable (TPE)
Multi-bandit methods	- Conceptually simple - Computationally efficient	- Balance between budget and number of trials	- Could be a default choice - Implemented by open-sourced libraries.
PBT methods	- Combine HPO and model training - Parallelism	- Constant changes to computation graph - Not extendable to advanced evolution	- For computationally expensive models

Figure 15. Comparison of major hyperparameter optimization algorithms (Yu & Zhu, 2020).

There are two main approaches for learning rate decay in hyperparameter optimization; time-based, which is continuous, and drop-based, which is discrete, see Figure 16.

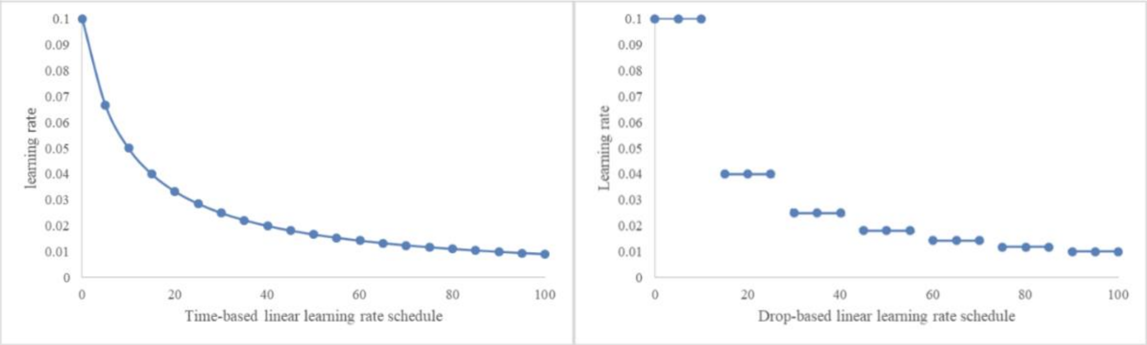


Figure 16. Linear decay of learning rate versus drop-based decay (Yu and Zhu, 2020).

8.2 Federated learning hyperparameter optimization

Similar to centralized machine learning, hyperparameter optimization remains a significant and time-consuming component of training a model. This process involves running experiments multiple times, each with a different set of hyperparameters. The importance of this task amplifies within the context of FL, considering the higher communication and computational costs associated with each experimental run.

FL presents two broad categories of hyperparameters: global and local. Global hyperparameters, shared among all clients within the federation, include model architecture, data preprocessing methods, loss function, and aggregation algorithms. On the other hand, local hyperparameters are client-specific and entail elements such as batch size, number of epochs, or model updates.

While optimizing local hyperparameters isn't a compulsory practice, it can provide several benefits. For instance, reducing the number of epochs could mitigate the issue of 'strugglers'—slow participants that delay the overall learning process. Furthermore, certain local settings might be critical for clients who wish to participate in the training, such as specifying a maximum batch size or outlining model architecture restrictions. While the latter counts as a global setting, it necessitates communication before initiating the experiments.

An effective approach to initiating global hyperparameter optimization could be to fine-tune the model on a single client's local dataset. However, it's important to note that the ideal hyperparameters for one client may not necessarily translate to optimal performance across the entire federation. This approach serves primarily as a starting point. According to recent research (Zhou et al., 2021), it's possible to optimize both global and local hyperparameters concurrently during a single experiment, potentially improving both the efficiency and performance of FL models.

9 Conclusion

In this document we have provided an overview of state-of-the-art methods in federated learning, focusing on aggregation methods, privacy methods, harmonization of images between hospitals and hyperparameter optimization. Federated learning is a very active area of research, and this is reflected by the fact that most of the papers in the reference list were published during the last 3-5 years.

10 References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 308-318).

Antunes, R. S., da Costa, C. A., Küderle, A., Yari, I. A., & Eskofier, B. (2022). Federated Learning for Healthcare: Systematic Review and Architecture Proposal. *ACM Transactions on Intelligent Systems and Technology (TIST)*.

Bashyam, V. M., Doshi, J., Erus, G., Srinivasan, D., Abdulkadir, A., Singh, A., ... & iSTAGING and PHENOM consortia. (2022). Deep generative medical image harmonization for improving cross- site generalization in deep learning predictors. *Journal of Magnetic Resonance Imaging*, *55*(3), 908-916.

Bhagoji, A. N., Chakraborty, S., Mittal, P., & Calo, S. (2019). Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning* (pp. 634-643). PMLR.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., ... & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1175-1191).

Chen, M., Mao, B., & Ma, T. (2021). Fedsa: A staleness-aware asynchronous federated learning algorithm with non-iid data. *Future Generation Computer Systems*, *120*, 1-12.

Choi, Y., Choi, M., Kim, M., Ha, J. W., Kim, S., & Choo, J. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 8789-8797).

Choquette-Choo, C. A., Dullerud, N., Dziedzic, A., Zhang, Y., Jha, S., Papernot, N., & Wang, X. (2021). Capc learning: Confidential and private collaborative learning. a*rXiv:2102.05188*.

Deng, Z., Huang, X., Li, D., & Yuan, X. (2022). FedGraph: an Aggregation Method from Graph Perspective. *arXiv:2210.02733*.

Fereidooni, H., Marchal, S., Miettinen, M., Mirhoseini, A., Möllering, H., Nguyen, T. D., ... & Zeitouni, S. (2021, May). SAFELearn: secure aggregation for private federated learning. In *2021 IEEE Security and Privacy Workshops (SPW)* (pp. 56-62). IEEE.

Haim, N., Vardi, G., Yehudai, G., Shamir, O., & Irani, M. (2022). Reconstructing training data from trained neural networks. *arXiv:2206.07758*.

Hatamizadeh, A., Yin, H., Molchanov, P., Myronenko, A., Li, W., Dogra, P., ... & Roth, H. R. (2023). Do gradient inversion attacks make federated learning unsafe?. *IEEE Transactions on Medical Imaging*.

Hu, L., Yan, H., Li, L., Pan, Z., Liu, X., & Zhang, Z. (2021). MHAT: An efficient model-heterogenous aggregation training scheme for federated learning. *Information Sciences*, *560*, 493-503.

Jere, M. S., Farnan, T., & Koushanfar, F. (2020). A taxonomy of attacks on federated learning. *IEEE Security & Privacy*, *19*(2), 20-28.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., ... & Zhao, S. (2021). Advances and open problems in federated learning. Foundations and Trends® in Machine Learning, 14(1–2), 1-210.

Ma, X., Zhu, J., Lin, Z., Chen, S., & Qin, Y. (2022). A state-of-the-art survey on solving non-IID data in Federated Learning. *Future Generation Computer Systems*, *135*, 244-258.

McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics* (pp. 1273-1282). PMLR.

Mächler, L., Ezhov, I., Kofler, F., Shit, S., Paetzold, J. C., Loehr, T., ... & Menze, B. H. (2021). FedCostWAvg: A new averaging for better Federated Learning. In *International MICCAI Brainlesion Workshop* (pp. 383-391). Cham: Springer International Publishing.

Oh, S. J., Schiele, B., & Fritz, M. (2019). Towards reverse-engineering black-box neural networks. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, 121-144.

Pati, S., Baid, U., Edwards, B., Sheller, M. J., Foley, P., Reina, G. A., ... & Bakas, S. (2022). The federated tumor segmentation (FeTS) tool: an open-source solution to further solid tumor research. *Physics in Medicine & Biology*, *67*(20), 204002.

Rieke, N., Hancox, J., Li, W., Milletari, F., Roth, H. R., Albarqouni, S., ... & Cardoso, M. J. (2020). The future of digital health with federated learning. *NPJ digital medicine*, *3*(1), 1-7.

Scoccianti, S., Detti, B., Gadda, D., Greto, D., Furfaro, I., Meacci, F., ... & Livi, L. (2015). Organs at risk in the brain and their dose-constraints in adults and in children: a radiation oncologist’s guide for delineation in everyday practice. *Radiotherapy and Oncology*, *114*(2), 230-238.

Shejwalkar, V., Houmansadr, A., Kairouz, P., & Ramage, D. (2022). Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In *2022 IEEE Symposium on Security and Privacy (SP)* (pp. 1354-1371). IEEE.

Song, J., & Ye, J. C. (2021). Federated CycleGAN for privacy-preserving image-to-image translation. *arXiv:2106.09246*.

Sylolypavan, A., Sleeman, D., Wu, H., & Sim, M. (2023). The impact of inconsistent human annotations on AI driven clinical decision making. *NPJ Digital Medicine*, *6*(1), 26.

Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., & Ristenpart, T. (2016). Stealing Machine Learning Models via Prediction APIs. In *USENIX security symposium* (Vol. 16, pp. 601-618).

Truex, S., Baracaldo, N., Anwar, A., Steinke, T., Ludwig, H., Zhang, R., & Zhou, Y. (2019). A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on artificial intelligence and security* (pp. 1-11).

Usynin, D., Rueckert, D., & Kaissis, G. (2022). Beyond gradients: Exploiting adversarial priors in model inversion attacks. *arXiv:2203.00481*.

Wang, T., Zhang, Y., & Jia, R. (2021). Improving robustness to model inversion attacks via mutual information regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 35, No. 13, pp. 11666-11673).

Wei, K., Li, J., Ding, M., Ma, C., Yang, H. H., Farokhi, F., ... & Poor, H. V. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, *15*, 3454-3469.

Xu, J., Glicksberg, B. S., Su, C., Walker, P., Bian, J., & Wang, F. (2021). Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, *5*(1), 1-19.

Yeganeh, Y., Farshad, A., Navab, N., & Albarqouni, S. (2020). Inverse distance aggregation for federated learning with non-iid data. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning: Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4–8, 2020, Proceedings 2* (pp. 150-159). Springer International Publishing.

Yin, X., Zhu, Y., & Hu, J. (2021). A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys (CSUR)*, *54*(6), 1-36.

Yu, T., & Zhu, H. (2020). Hyper-parameter optimization: A review of algorithms and applications. *arXiv:2003.05689*.

Zhou, Y., Ram, P., Salonidis, T., Baracaldo, N., Samulowitz, H., & Ludwig, H. (2021). Flora: Single-shot hyper-parameter optimization for federated learning. *arXiv:2112.08524*.

Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2223-2232).