Investigation of Federated Learning on multicentric surgical video data in terms of the influence on model performance

Master Thesis

Vincent Markert

Division Translational Surgical Oncology
NCT Dresden

Reviewer:	Prof. Dr. Stefanie Speidel
Second reviewer:	Dr. Sebastian Bodenstedt
Advisor:	M. Sc. Alexander Jenke

Duration: November 15, 2021 – April 18, 2022

NCT
NATIONAL CENTER
FOR TUMOR DISEASES
PARTNER SITE DRESDEN
UNIVERSITY CANCER CENTER UCC

TECHNISCHE
UNIVERSITÄT
DRESDEN

Abstract

The strict regulation of privacy-sensitive personal medical data complicates the ability to gather extensive multicentric datasets, hindering the ability to train highly-performant and well-generalizing medical machine learning models. The novel approach of federated learning decentralizes the training process, allowing several institutions to collaboratively train a model while keeping the used data stored privately at each institution. The diverse range of recently published federated learning approaches requires comparative studies in terms of model performance. Further, comparisons to classical learning procedures, such as trainings with a centralized dataset and trainings where each institution uses only its local data, are needed. These are done in this thesis by thoroughly investigating the performance of federated learning approaches for multiple models on a combination of two datasets containing surgical video data. We were able to show that most of the considered federated learning algorithms achieve comparable results with the centralized training approach on a multi-label classification task and improving clearly over the local only trainings. Regarding the results of the local trainings, two variants of interpretation are established and compared, which represent different client motivations. Furthermore, in this work, performance differences among different institutions of a federation are discussed and associated with the respective client datasets. For the given task, it was found that federated learning yields larger improvements for labels that occur less in the data and that the influence of the considered machine learning model remained small.

Contents

1. Introduction

1.1. Motivation

Machine Learning (ML) has proven to be very capable of solving a large range of different tasks, though the availability of large labeled datasets remains crucial to the performance and generalization capability of ML models in the supervised setting [1]. This dependence can lead to problems in situations where such data is not easily available, such as in the medical domain [2]. Although each medical institution can collect data and build datasets on its own, which already involves a very time- and cost-consuming annotation process, the sharing and publishing of this data underlie heavy regulations and restrictions, making it difficult to gather extensive multicentric datasets [3, 4]. This is especially the case since the increased awareness for data privacy in the society led to the introduction of measures like the General Data Protection Regulation (GDPR) [5], which determines strict regulations for institutions in the European Union that intend to store, process or exchange personal data with other institutions. Existing efforts to bypass these regulation limitations, such as anonymization, can lack the required level of privacy preservation [6, 7, 8]. Therefore, the resulting limited availability of data hinders the classical ML approach. Nevertheless, it is desirable to build ML models on the diverse multicentric medical data, as this increases the generalization capabilities of these models [9].

To be able to build strongly generalizing ML models without the need of sharing data, the novel discipline of Federated Learning (FL) [10] decentralizes the training process from a single to multiple centers. Each institution in the federation computes local updates for a global optimization goal and shares these updates instead of their private data.

Just recently, this approach has enabled a cooperation of hospitals around the world to build a model capable of performing solid predictions on the further disease progress of symptomatic COVID-19 patients [11]. As a result, it yields performance increases for every participating hospital compared to their solely locally trained models. Other applications of FL report similar results, where most of the participants see local performance increases for different tasks in the medical domain, such as whole prostate segmentation [12] or brain cancer prediction [13]. These examples highlight the potential and the importance of this relatively new approach.

While FL provides the clients access to a model with improved capabilities, for the decentralized approach, one still expects to see a decrease in performance when compared to a centralized training [10]. In that regard, it is important to assess the amount of performance reduction that can be anticipated and if the benefit of the increase in performance when learning in a federation outweighs these problems. This examination of the limitations of FL in terms of model performance motivates this thesis.

Although medical application examples such as [11], [12], and [13] report about the performance benefits for their specific tasks, they only use the original federated algorithm named Federated Averaging (FedAvg), and none of the algorithms that were subsequently published.

Additionally, newly published algorithms are often only compared to the baseline of FedAvg and sometimes to the centralized training, where all the data resides at one center. In some cases, they are also matched up against other algorithms. However, those algorithms often build upon the same idea. For instance, algorithms that adjust the training procedure by adding a regularizer to the client loss function are only compared to other algorithms that follow this approach. To our knowledge, how federated algorithms with entirely different methodologies compare to each other on a given task in terms of model performance has not been thoroughly investigated. This motivates the comparative studies in this thesis.

1.2. Objectives

Due to its' novelty, many different methods have been proposed in recent years to tackle different challenges arising in FL. This thesis aims to compare a multitude of promising FL approaches and analyze the potential strengths and weaknesses of each of them, thereby building an overview of the possibilities within the field. A primary focus lies on the applicability of FL for minimally-invasive surgical video data.

The objectives can be summarized as follows:

1. Compare a set of FL methods on medical data with respect to the model performance.
2. Highlight the differences of the federated approach compared to traditional centralized training and local training procedures that are performed at each institution individually.
3. Discuss the relevance and capability of FL in the medical domain.

The thesis is structured as follows. Firstly, in chapter 2, the theoretical definitions of ML and FL are presented, and an overview of the types and challenges of FL is given. The related work in FL is covered in chapter 3. In chapter 4, multiple FL algorithms are presented in more detail, and the required metrics, loss function, and models are introduced. In chapter 5, the experimental setup is presented, and a detailed analysis of the dataset is given. In chapter 6, the results are presented, which are discussed in chapter 7. Here, the differences between the presented FL algorithms are assessed, and they are compared with classical learning approaches. Different aspects of FL are highlighted more thoroughly, and its potential for the medical domain is discussed. Lastly, chapter 8 concludes the findings of this work and provides suggestions for future work.

2. Theoretical Background

In the following subsections, ML is introduced, and the theoretical definition of the FL problem is stated. The different types of Federated Learning Systems (FLSs) are summarized, and the application domain of a Multicentric Medical Federated Learning Systems (MMFLSs) is categorized within these type descriptions. Followed by that, we dive deeper into the challenges of learning models via FL. Additionally, we discuss the relevance of the described problem aspects for federations of medical institutions looking to build performant data-driven models.

2.1. Machine Learning

ML is a discipline of artificial intelligence. The subfield of supervised ML comprises the building of a statistical model using a set of input-output pairs, also called a training dataset. The model learns to make predictions for the inputs based on the labels. This is done by rating the similarity, as quantified by a loss function, of the output predicted by the ML model given the input and the corresponding given target output. An optimizer such as Stochastic Gradient Descent (SGD) [14] is applied to the model parameters, where the gradient is computed from the loss with respect to the inputs. This search for a loss minimum leverages the ability of the model to predict the correct output and is the model training process. The generalization capability, i.e., the ability of the model to infer predictions on unseen inputs, is evaluated by testing the model performance on a test dataset, which is held back during the training process. In classical ML, all data needs to be collected in a single center if datasets from different institutions shall be used.

Due to their ability to characterize highly complex predictive functions, Artificial Neural Networks (ANNs) have seen broad use as an ML model in recent times. An ANN is a network architecture built up out of connected layers of artificial neurons in an effort to mimic the connectivity in the human brain. It consists of an input layer, an output layer, and intermediate hidden layers. The input is propagated from layer to layer as a weighted function of the precedent layer. The predictions are based on the output layer neuron activation. The subfield of Deep Learning (DL) uses models with a high number of hidden layers which further increases the fidelity of the predictive function.

Convolutional Neural Networks (CNNs) [15] are a type of ANN that contain convolutional layers. In these layers, a filter is moved as a sliding window over the layer input. An entry in the layer's output consists of the discrete convolution, i.e., the inner-product, of the filter with the section in the input covered by the filter. Therefore, neighboring activation values influence each other, with the spatial extent of this influence being set by the filter size. CNNs are models known to perform very well on image data [16].

2.2. Federated Learning Setting

Federated Learning was firstly introduced by Google researchers in 2016 in [10]. Instead of having the requirement to combine data from multiple institutions into a centralized dataset and perform training on that dataset, FL enables a decentralized training process that is performed locally at each institution. This procedure alleviates the need to share the data of each institution. In the context of FL, institutions are also called clients.

Let S be a sample space, X be a feature space, and Y be a label space. Let $\mathcal{M} := \{(x_s, y_s)\}_{s\in S}$ be a labeled dataset with $x_s \in X$ and $y_s \in Y$, and $n = |\mathcal{M}|$ be the size of that dataset. Let $f_\omega(x)$ be the model function output of an ML model, parametrized by ω, and $\ell(f_\omega(x), y)$ be a loss of the model output and the ground truth label.

The authors of [10] use the general finite-sum objective

$$\min_\omega h(\omega) \qquad \text{with} \qquad h(\omega) := \frac{1}{n}\sum_{s\in S} h_s(\omega), \tag{2.1}$$

where in the ML setting, $h_s(\omega)$ is usually the loss $\ell(f_\omega(x_s), y_s)$ of the data of sample s with model parameters ω. This way, (2.1) represents the minimization of the global loss over all data points.

If we assume the data is not stored as a global dataset $\mathcal{M}$, but instead in K parts, we can use a partition $\mathcal{P} = \{S_1, ..., S_K\}$ of S to split the data as $\mathcal{M}_{S_1} \dot\cup ... \dot\cup \mathcal{M}_{S_K} = \mathcal{M}$, where $\mathcal{M}_{S_k}$ are the data points in $\mathcal{M}$ from samples S_k. The objective (2.1) then can be rewritten as

$$\min_\omega h(\omega) = \min \sum_{k=1}^{K} \frac{n_k}{n} H_k(\omega) \qquad \text{with} \qquad H_k(\omega) = \frac{1}{n_k}\sum_{s\in S_k} h_s(\omega), \tag{2.2}$$

where $n_k = |\mathcal{M}_{S_k}|$, such that now $H_k(\omega)$ is the loss over a client's dataset $\mathcal{M}_{S_k}$.

This construction reflects the typical situation for FL, where different parts of the dataset are owned by K different clients.

2.3. Federated Learning Types

2.3.1. Processing Order

In order to solve the FL optimization goal (2.2), all clients are required to perform trainings for a global model on their local data. There are different ideas, in which order the clients perform these local trainings. An overview is given in Figure 2.1.

In Institutional Incremental Learning (IIL) [13, 17], a global model is given to the clients in sequential order. The first client gets the initialized model and passes the model on to the next client upon finishing local training. This transfer is done until the last client receives the model.

Cyclic Institutional Incremental Learning (CIIL) [13, 17] adjusts this idea in that it allows for multiple cycles of the IIL process. Therefore, the model is repeatedly trained sequentially for a number of epochs.

Another possibility is performing the local training in parallel and then combining the resulting models to a global model, e.g., by averaging. In [13] and [17], the authors show

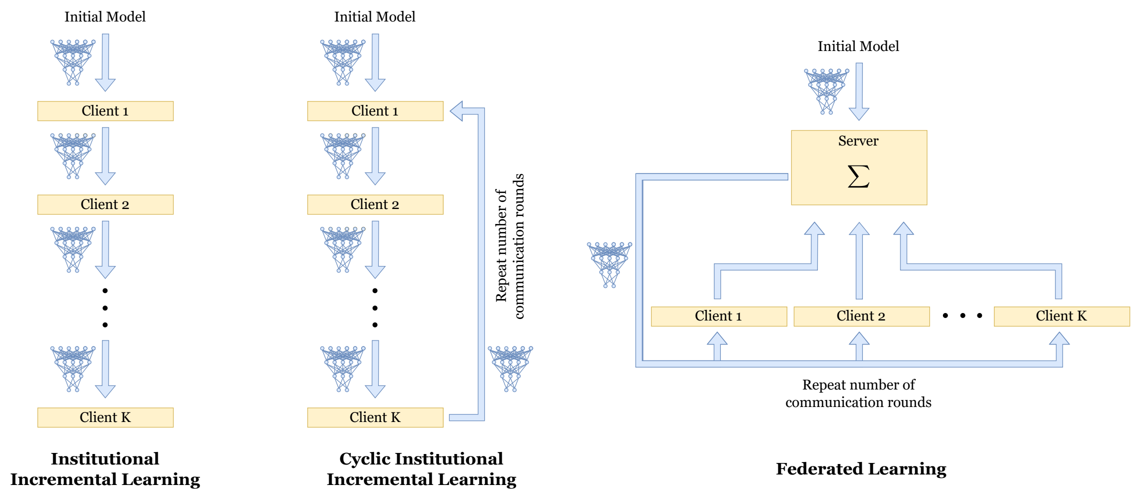

Initial Model
Client 1
Client 2
Client K
Institutional Incremental Learning
Initial Model
Client 1
Client 2
Client K
Repeat number of communication rounds
Cyclic Institutional Incremental Learning
Initial Model
Server
Σ
Client 1
Client 2
Client K
Repeat number of communication rounds
Federated Learning

Figure 2.1.: Different processing orders for federated models. Figure adapted from [13].

that the sequential approaches IIL and CIIL are inferior to the parallel approach due to catastrophic forgetting, which refers to the problem where previously learned patterns are forgotten when training is continued on a new dataset. Hence, the model favors the data of clients that process the model last.

Due to the inferiority of the sequential model training process, most of the algorithms proposed in recent years are based on the structure of a parallel algorithm called FedAvg, which was proposed alongside the FL problem definition in [10] and is presented in more detail in subsection 4.1.2. Several variants that build upon FedAvg in different aspects are presented in subsequent sections.

Generally, in FedAvg, a global model is controlled by a server. The server sends a copy of its current model to a number of clients, which perform local training on their private data and send the resulting model updates back to the server. The server then aggregates the local updates into a single global model update. This process is repeated for a number of epochs. Ultimately, this creates a global model that takes advantage of the various data from all clients and typically provides a performance improvement to most or all participants compared to training on just the respective local data [11, 12, 13].

Some approaches further remove the requirement of having a server coordinating the FL training process, which is called fully decentralized FL, peer-to-peer FL, or swarm learning [18, 19]. Here, the local updates are directly sent to other clients instead of aggregating them at a server. As no positive effects on the model performance are expected for these algorithms, they are not considered in this work.

2.3.2. Data Partitioning

Based on how the data of different clients is distributed over the sample space S and feature space X, FL can be divided into vertical FL and horizontal FL [20, 21, 22].

In vertical FL, different clients possess a similar sample space but distinct feature spaces. An example would be a hospital that sends patients to a specific cardiologist [22]. Both the hospital and the cardiologist perform different tests and collect data of different kinds (features) but will most likely share many of its patients (samples).

In horizontal FL, different clients possess a similar feature space but different sample spaces. An example would be a particular diagnostic protocol performed the same way at different hospitals. The hospitals thereby collect the same features from most likely different samples.

From the examples above, it is seen that both cases occur in a medical environment. Nevertheless, most of the research regarding FL is only concerned with horizontally split data [20, 22]. FedAvg and its closely related variants implicitly require a horizontal data partitioning, as they assume the same model architecture, and hence the same features, for all clients and the server. Our formulation of the FL problem (2.2), which is slightly different from the proposed version in [10], also suggests a horizontal data partitioning since we directly partition the sample set S by $\mathcal{P}$. However, vertical FL is not excluded if the sample set is allowed to contain duplicates, which are distributed to different clients by the partitioning $\mathcal{P}$.

The authors in [20] even claim that, as of yet, there is no well-developed algorithm for the vertical FL. The main issue for vertical FL is finding a way to perform privacy-preserving record linkage, which refers to assigning data from different clients to the same sample [22]. Hence, vertical FL remains an open problem. For this reason, we focus on comparisons of algorithms within horizontal FL in this work.

2.3.3. Federation Scale

The scale of a FLS can be determined by two factors [20]: the number of clients K and the amount of data residing at each client $n_k, k = 1, ..., K$.

We distinguish between two cases [20, 21]: cross-device FLSs and cross-silo FLSs. In cross-device FLSs, there is a large number of participating clients K, which, however, have a comparatively low number of data samples n_k and low computational power. As the name suggests, a typical example is mobile devices.

On the contrary, cross-silo FLSs include a lower number of clients K, which, however, possess larger amounts of data n_k and usually the computational power to process this data. Therefore, the clients are larger bodies such as institutions or data centers. Hence, an MMFLS falls in the segment of cross-silo FLSs.

In reality, a FLS might also lie in between these two scenarios, making these categories not exhaustive [18]. However, these two categories represent the extreme cases that an FL algorithm can be aimed at.

2.4. Statistical Heterogeneity

2.4.1. Data Heterogeneity

One of the most prominent problems arising in the FL setting is data heterogeneity amongst different clients in the federation. Although all clients should use data that is somewhat related to each other, the clients draw their data from various sources, and each client might use a different data collection process. For example, hospitals at distant locations collect information from different demographic groups or might use varying parameter configurations when generating images via Computer Tomography (CT). This concept is also called Medical Institution Data Bias (MIDB) [13] and is known to occur in practice [23, 24]. An MIDB equivalently means that local client data distributions $\mathcal{D}_k$ can deviate strongly from the distributions of the other clients $\mathcal{D}_l, l \neq k$, or the collective global distribution $\mathcal{D}$. Then, $z^k = (x, y)^k \sim \mathcal{D}_k$ and $z^l = (x, y)^l \sim \mathcal{D}_l$ are non-Independent and Identically Distributed (non-IID) random variables for $k \neq l$.

Common ways in which client distributions deviate from each other are [18]: (1) Feature distribution skew, (2) Label distribution skew, (3) Same label, different features, (4) Same features, different labels, and (5) Quantity skew. In reality, the data distributions might result from a mixture of these types.

As a consequence of the non-IID assumption, in FL, it is not possible to learn the global objective function from just a single client's data [10], i.e.,

$$\mathbb{E}_{S_k}[H_k(\omega)] \neq h(\omega). \tag{2.3}$$

This problem specifically arises in the FL environment, whereas sampling data randomly over the global dataset yields Independent and Identically Distributed (IID) batches in the centralized approach. The absence of an IID assumption for the client data also separates FL from classical distributed machine learning, where the data is sampled uniformly from a given dataset and then processed by multiple devices. Here, the global objective function could be, given enough samples, approximated by any of the devices as

$$\mathbb{E}_{S_k}[H_k(\omega)] = h(\omega). \tag{2.4}$$

The second distinct difference to distributed learning is the required communication between the server and the clients, with a potential for high communication costs [25] (subsection 2.5.3) and a possibility of security risks [26] (subsection 2.5.1) being involved in this communication.

Although FedAvg was proposed with the non-IID setting in mind, the authors in [27] and [28] show that increasingly non-IID client data, in consequence, leads to a significant reduction in performance for FedAvg. They attribute the decreased accuracy to diverging weight updates from the clients. In practice, FedAvg might even diverge [10, 29].

To tackle the data heterogeneity problem, the domain of personalization arose in recent years.

2.4.2. Personalization

Personalization refers to the concept of adapting the training process of a FLS to be able to deal with non-IID client data. This is done by regulating the training process of the global model to be more stable to data heterogeneities or using unique adapted federated models for the clients at inference time instead of a single federated global model. Personalized Federated Learning (PFL) thereby tackles the problem of data heterogeneity by allowing the global model or the unique local models to adapt to the shifted local data distributions $\mathcal{D}_k$ of each client while still benefiting from a shared training process. It represents an intermediate paradigm between an FL-trained global model approach and the purely-local model training paradigm [30].

The ways to accomplish such a personalization are manifold. A comprehensive taxonomy for PFL approaches is given in [31]. An adapted taxonomy is given in Figure 2.2, where the methods used in this work are categorized. The authors of [31] distinguish between data-based and model-based PFL. In data-based PFL, one seeks to reduce the heterogeneity of data residing at the clients participating in the FL process. This reduction is made with data augmentation in order to make the clients' data more homogeneous or with a client selection process that chooses clients that have more homogeneous data in the first place.

Model-based PFL comprises a diverse field of methodologies. Those can be divided into Single-Model, Multi-Model and K-Model approaches, where naming reflects the number of unique inference models trained in the FL process. Hence, for Single-Model approaches, a single global model is trained that is more adapted to the local data, while one separate inference model is trained for each client in a K-Model approach.

PFL is about improving the model performance in a heterogeneous client setting and thereby aims at reducing the difference to the centralized training. Hence, it is the most

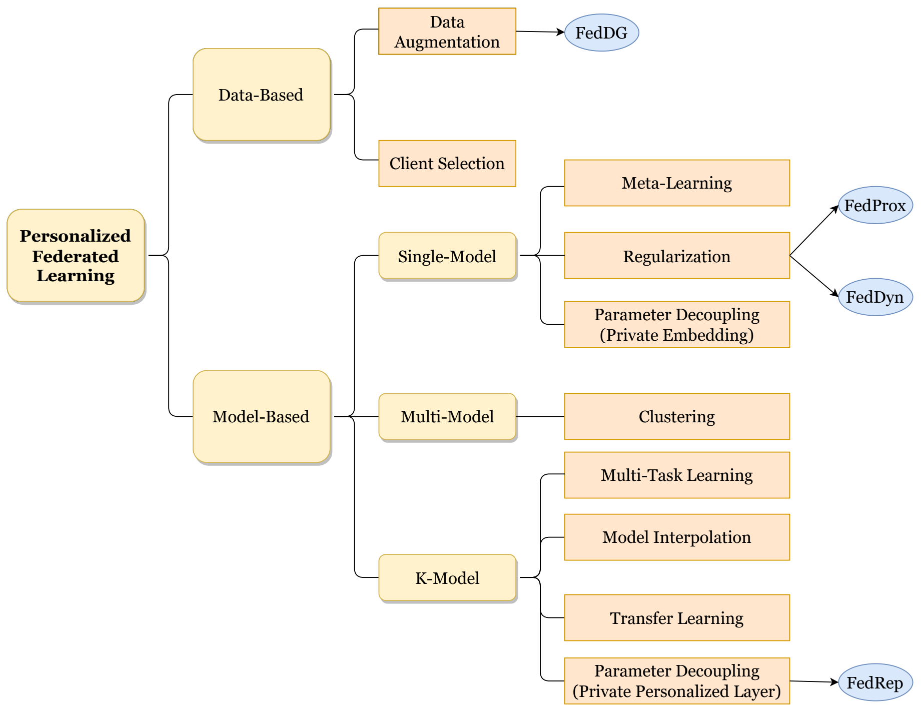


Figure 2.2.: A taxonomy of personalized federated learning adapted from [31].

important research field for the objective of this thesis out of the fields presented in this chapter. Therefore, the methods used for experimentation in this work are mainly personalization techniques. The categorization of each of our used methods is included in Figure 2.2. It can be seen that the methods which were chosen in this work cover a diverse range of ideas.

Due to the relatively low number of clients ($K = 5$) in our cross-silo test case, the domain of Client Selection is not as practicable. The same is the case for Clustering, whereas we want to see the difference between a single global model that might generalize well and K models that can more flexibly fit each client's local optimization objective.

2.4.3. Client Data Analysis

With increasing numbers of inference models, situations with increasingly heterogeneous client distributions can be handled with high performance. More specifically, in some cases, for example with client data heterogeneity in the form of ”same features, different labels”, it might be impossible to model the data correctly with just a single global model. On the other hand, there might be cases where there is no significant heterogeneity amongst clients, and it is unnecessary to train multiple models. Besides running simulations for both settings, the decision on a PFL architecture can merely be made by analyzing the heterogeneity of the data beforehand.

Different metrics to determine the amount of client heterogeneity named in FL research include the 1-Wasserstein distance, also known as the Earth Mover's Distance (EMD), and the Total Variation [32]. The authors in [33] show that out of a handful of available bin-to-bin similarity measures, the EMD is also the best at comparing the similarity of distributions, although at the cost of high computational complexity. Nevertheless, all the metrics mentioned in [33] require direct access to the private client data and therefore are impracticable in a real FLS. Additionally, these metrics are hard to compute for large numbers of RGB images. Other metrics, such as the clients' local dissimilarity [29], can only be evaluated after the training is finished. Nonetheless, this metric is used in this work to estimate data heterogeneities. The problem of an a priori privacy-preserving client heterogeneity analysis remains open [31].

2.4.4. Fairness

The existing heterogeneity in the client datasets also inevitably leads to a situation where the model or models generated in the FL process achieve different accuracies on each client dataset. This fact elicits the question of how fair the performance benefits are distributed amongst the clients. The amount and distribution of the data that a client contributes to the federation are important factors that influence the resulting model. In this work, we take a more detailed look at how different the amounts of benefits for each client are and associate those with the respective contributed data.

2.5. Further Challenges of Federated Learning

Additionally to the investigated topics in this thesis, the following challenges need to be considered in a real-life application of FL. As they are not directly concerned with the performance of the federated ML model, no explicit focus is laid on those topics in the subsequent chapters. Nevertheless, they are essential subjects in FL research, and their relevance in an MMFLS is shortly categorized.

2.5.1. Privacy and Security

In classical (distributed) ML, the training process is typically kept within a single organization and thus protected. Only at inference time can a model be targeted with attacks by adversaries. Contrarily, in FL, the fact that the whole training process is distributed amongst different institutions represents a new attack surface and makes a new range of attacks possible [34].

Essentially, the purpose of FL is to train well-performing models while keeping the data on which the models are trained secret to its owner. This purpose raises the question of how safe and private the client data is when participating in FL and how reasonable it is to assume that a model was trained to the best of its potential and not maliciously influenced by an adversary. Those questions motivate a research field concerned with the many new attack and defense mechanisms [21, 22, 34, 35]. The amount of privacy risk involved in a FLS depends on the position of the adversary, e.g., a client or server. However, in an MMFLS, we assume a reliable and well-intentioned environment on both sides.

2.5.2. System Heterogeneity

In a real-world federation of institutions or devices, each client has a different quantity of data and possesses a specific amount of computational power. When running trainings with a similar-sized model on these clients synchronously, each respective client's training duration might vary highly. This variance, in turn, can lead to much idle time for those clients who can perform updates faster. Approaches that tackle this problem include a variable model size for different clients [36] and an asynchronized training procedure [37, 38].

In MMFLSs, there typically is no specific time requirement for the training process, and in the test setting of this thesis, the data amounts are processable in a reasonable time. Therefore, in this work, no methods with the intent of reducing client idle times are included. Additionally, mixing model architectures, as in [36], introduces an extra decision process for the different models and makes these methods harder to compare.

While the abovementioned ideas speed up the FL training process, communicating the acquired updates in an efficient manner is also important in that respect.

2.5.3. Communication Efficiency

The research field of communication efficiency is concerned with a faster and more efficient transport of the updates between the clients and the server and is mainly important in cross-device FL, where a many clients are present.

The major goal is to compress the communicated gradient updates and shared model without having a significant loss in model performance [18].

3. Related Work

The universality of the idea of decentralizing the training process of an ML model through FL motivates the applicability in any setting where multiple participants are willing to collaborate in building better performing ML models that use data from every participant without explicitly exposing the actual information.

For this reason, the original publication sparked interest across many domains and led to a wide range of exploratory research that suggests adaptions to the original procedure. Alongside broad introductions to the topic of FL, the following works provide overviews of the many approaches proposed in recent years and the challenges of FL.

[20] and [21] provide taxonomies to classify the different types of FL, as discussed in section 2.3. [20] furthermore presents a categorization of many recently published FL algorithms in their taxonomy and names different benchmark frameworks that enable efficient evaluations and comparisons of those algorithms. They also point out that well-developed algorithms are missing for the vertical FL setting.

[18] and [39] identify four of the main challenges for FL and, therefore, the core directions for improvements. These are the performance robustness of FL algorithms influenced by statistical data heterogeneity compared to a centralized training process, privacy and security, system heterogeneity, and communication efficiency. Within each of those research areas, respective publications are presented. Furthermore, the authors in [18] provide discussions of fully decentralized FL and communication efficiency. In [39], the importance of metrics to quantify statistical data heterogeneity applicable in FLSs is highlighted.

Similarly, in [31], it is pointed out that this search for heterogeneity metrics is an open problem. Besides that, [31] focuses on the research field of personalized FL. They provide a general taxonomy for personalization techniques, including all algorithms that are concerned with client data heterogeneity, and which therefore covers all algorithms presented in section 4.1.

More specifically, the following works summarize advances of FL in the medical domain. [22] provides a systematic literature review in FL and covers a large range of topics therein, including the above topics of types and challenges of FL. However, here the authors take the perspective of the health sector and the effects therein. Amongst other applications for medical institutions, they name medical example cases of the different types of FLSs.

The authors in [6] take a more detailed look at the aspects of collecting data in a medical environment and depict the roles and benefits of each stakeholder involved in creating a medical federation. They point out many different arguments for the usage of FL in the medical sector.

Due to the high need for comparative studies in FL there are efforts to make the implementation and assessment of FL algorithms easier and standardized. Frameworks such as FedML [26] and TFF [40] provide research libraries and benchmarks for the federated learning setting. They introduce programming interfaces that allow the users to focus on algorithmic research instead of low-level communication backend details. The FL algorithms can be compared systematically using the provided benchmark datasets and models. Additionally, a handful of federated algorithms are included in the library, covering a diverse field of topologies and training procedures. However, to this point, these include rather representational basic algorithms in each domain.

Regarding the comparison of FL algorithms with traditional learning procedures, [13], [17], and [41] are studies on different subjects in the medical field. These works, however, only consider the original algorithm FedAvg.

4. State of the Art

At the beginning of this chapter, all learning algorithms used in the experimental setup are introduced. Firstly, the two baselines required for comparisons are described. Then, all considered FL approaches are presented. Afterward, different metrics, the loss function, and ML models considered in the experiments are presented.

4.1. Learning Algorithms

4.1.1. Baselines

To compare the FL algorithms with traditional ML procedures, both an upper and a lower baseline learning approach are used.

The upper baseline is represented by a model that is trained on a centralized dataset. This approach corresponds to the classical ML scenario where all the data is accessible and controlled by a single center. Hence, the training can be performed on IID sampled data batches. This type of training is referred to as the Centralized Training (CENT).

The lower baselines are given by models solely trained on each of the clients' respective local datasets. This work refers to those as Local Only Trainings (LOTs). They are representative of clients that neither collaborate in a FLS to benefit from different data distributions from other clients nor have the possibility to access and use explicitly shared datasets. They only use data residing in their own data storage.

With the K models computed in a LOT, there are different variants of interpreting the results. With the Altruistic View (ALTV), the performance of each client model is given as its global score, i.e., the performance of the model over all client test datasets combined. In this case, the generalization capability of each model is included in the evaluation. This variant is similar to how all FL algorithms are evaluated.

On the contrary, with the Egocentric View (EGOV), the performance of each client model is given as the local score on the client's own test dataset. This interpretation is particularly suitable for assessing how beneficial it is for a client to participate in a FLS since a client might be mainly interested in what performance advantage it gains on its own data.

In both cases, to determine the overall performance, a Weighted Average (WAVG) over the client model performances is computed, which, in this work, is defined as an average with weights corresponding to the relative data amounts of clients.

This work refers to the group of all trainings with the following FL learning schemes as Federated Learning Trainings (FLTs).

4.1.2. FedAvg

Federated Averaging (FedAvg) [10] represents the original algorithm framework for the FL setting, as it was published alongside the problem definition.

One way to solve the global FL optimization given in (2.2) would be to move the parameters according to:

$$\omega_{t+1} = \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla H_k(\omega_t) := \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k, \tag{4.1}$$

i.e., applying a step of SGD [14] with learning rate η to the current global parameters ω_t with an accumulated gradient consisting of the local gradients of each of the K clients. This is possible due to the fact that

$$\nabla h(\omega) = \nabla \sum_{k=1}^{K} \frac{n_k}{n} H_k(\omega) = \sum_{k=1}^{K} \frac{n_k}{n} \nabla H_k(\omega) := \sum_{k=1}^{K} \frac{n_k}{n} g_k. \tag{4.2}$$

Equivalently to (4.1), we can first update the parameters for each client separately, starting from the initial global parameters,

$$\forall k : \omega_{t+1}^{k} = \omega_t - \eta_{local} g_k \tag{4.3}$$

and then aggregate the local updates into one global update

$$\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k}. \tag{4.4}$$

This suggests that it is also possible to perform multiple local update steps before aggregating the global update. Also note, that the learning rate is now part of the client update and not the global update anymore. Therefore, it is reworded to η_{local}. Ultimately, this results in the following methodology, which the authors in [10] name FedAvg.

The training process of the global model is controlled by a server. The server initializes its model parameters as ω_0 and sends a copy to a fraction C of random clients. In the following sections, special cases of formulations for $C < 1.0$ are excluded, as C is set to 1.0 in the experiments in this work. The reason for this choice is explained in subsection 5.2.3. The chosen clients synchronously perform a number of E_{local} local training epochs with a local batch size B via SGD:

$$\omega^k = \omega^k - \eta_{local} \nabla \ell(f_{\omega^k}(x_s), y_s) \text{ for } s \in S_k. \tag{4.5}$$

Therefore, each client tries to solve the local subproblem

$$\min_{\omega^k} H_k(\omega^k) = \frac{1}{n_k} \sum_{s \in S_k} \ell(f_{\omega^k}(x_s), y_s). \tag{4.6}$$

Upon finishing its training, each client sends its updated model ω_{t+1}^{k} back to the server, which in turn aggregates the local updates into a single global update as

$$\omega_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k} := \sum_{k=1}^{K} c_k \omega_{t+1}^{k}. \tag{4.7}$$

While, in this aggregation scheme, each local update is weighted by the corresponding client data amount (WAVG), there is also the possibility of an unweighted aggregation with $\forall k : c_k = \frac{1}{K}$. The server repeatedly sends its newly aggregated global model to clients and receives updated local models. This process is done for a given number of global epochs E_{global}, in this context also called communication rounds, or until a convergence criterion is met.

4.1.3. Adaptive Optimizer for Federated Learning

Since $n_1 + ... + n_K = n$, we have $c_1 + ... + c_K = 1$. Therefore, similar to the authors in [42] we can rewrite the FedAvg global aggregation update (4.7) as

$$\omega_{t+1} = \sum_{k=1}^{K} c_k \omega_{t+1}^k = \omega_t - \sum_{k=1}^{K} c_k (\omega_t - \omega_{t+1}^k). \tag{4.8}$$

Let $\Delta_t := \sum_{k=1}^{K} c_k (\omega_t - \omega_{t+1}^k)$. Hence, we see that the FedAvg global update

$$\omega_{t+1} = \omega_t - \Delta_t \tag{4.9}$$

can again be interpreted as a step of SGD [14] with a new learning rate $\eta_{global} = 1$ and pseudo-gradient Δ_t. This argumentation works not only for FedAvg but all FL algorithms with this kind of global aggregation. From this formulation, it appears that other global learning rates η_{global} could be used and that one might apply other optimizers than SGD for the global update.

This fact enables the use of more sophisticated adaptive optimizers such as Adam [43] or AdamW [44]. Alongside, it is possible to use adaptive optimizers for the local training as well [42].

4.1.4. FedProx

As mentioned before, [27] and [28] show that the client updates might diverge in FedAvg for non-IID client data distributions. Regarding that, a minor modification to FedAvg is given in [29]. They tackle the problem of heterogeneous data across different clients by adding a proximal term to the local objective (4.6) of each client:

$$\min_{\omega^k} v_k(\omega^k; \omega_t) = H_k(\omega^k) + \frac{\mu}{2} \left\| \omega^k - \omega_t \right\|^2, \tag{4.10}$$

parametrized by μ. The authors refer to the resulting algorithm as Federated Proximal (FedProx). The introduced regularizer aims to keep the local parameters ω^k from deviating too much from the original global parameters ω_t.

4.1.5. FedDyn

The authors in [45] extend the principle of FedProx. Additional to the proximal term used in FedProx, which uses the received server model, they add a dynamic regularizer involving the current local device gradient. They name their method Federated Dynamic Regularizer (FedDyn). The local objectives (4.6) become:

$$\min_{\omega^k} v_k(\omega^k; \omega_t) = H_k(\omega^k) + \frac{\mu}{2} \left\| \omega^k - \omega_t \right\|^2 - \left\langle \nabla H_k(\omega_t^k), \omega \right\rangle. \tag{4.11}$$

The local dynamic regularizer is updated recursively at the end of each local training according to:

$$\begin{aligned} \nabla H_k(\omega_{t+1}^k) - \nabla H_k(\omega_t^k) + \mu(\omega_{t+1}^k - \omega_t) = 0 \\ \iff \nabla H_k(\omega_{t+1}^k) = \nabla H_k(\omega_t^k) - \mu(\omega_{t+1}^k - \omega_t), \end{aligned} \tag{4.12}$$

which uses the first-order condition for local optima. They further adapt the aggregation scheme. The differences between the local updates and the original server state are accumulated as

$$h_{t+1} = h_t - \mu \frac{1}{K} \sum_{k=1}^{K} \left(\omega_{t+1}^k - \omega_t \right), \tag{4.13}$$

and then the aggregation is performed as

$$\omega_{t+1} = \left(\frac{1}{K} \sum_{k=1}^{K} \omega_{t+1}^k \right) - \frac{1}{\mu} h_t. \tag{4.14}$$

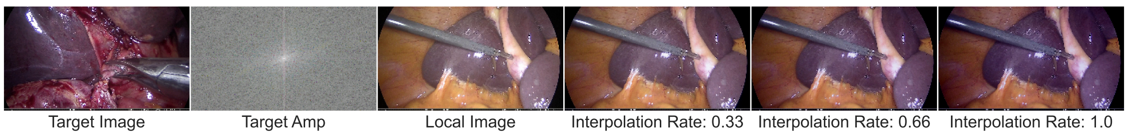


Figure 4.1.: Example of the amplitude interpolation performed in FedDG [46].

4.1.6. FedDG

With Federated Domain Generalization (FedDG), [46] the authors introduce a new way of data augmentation, which uses image data from different clients in a privacy-preserving way. With the Fast Fourier Transform (FFT) [47], the frequency space signal of an image can be extracted, which can be further decomposed into amplitude and phase spectrum, which respectively contain low-level and high-level information of the image. While the phase spectrum contains information about the actual objects in the image, which needs to be kept private, the amplitude spectrum, which encodes the style of the image, can be shared with other clients without the risk of the images being reconstructed from solely that information. In FedDG, the amplitude spectrum signals of all images are therefore centrally stored in an amplitude bank accessible to all. The authors further propose an interpolation scheme that allows to continuously shift the amplitude of an image between two given amplitudes $\mathcal{A}_i$, $\mathcal{A}_j$:

$$\mathcal{A}_{i,\lambda} = (1-\lambda)\mathcal{A}_i + \lambda\mathcal{A}_j, \tag{4.15}$$

where $\lambda \in [0,1]$ is the interpolation rate. The binary mask to control the scale of the amplitude component from the original formulation is excluded here. Therefore, the complete amplitude information is interpolated.

Given an input, the data augmentation can then be performed by applying an FFT to the input, decomposing the frequency into amplitude and phase spectrum, sample an interpolation rate $\lambda \in \mathcal{U}(0,1)$, interpolating the amplitude with a random entry in the amplitude bank, and reconstructing the image with the original phase spectrum and the interpolated amplitude spectrum via an inverse FFT. Hence, the domain of the image is shifted towards the domain of another image. An example is given in Figure 4.1. While the authors in [46] combine this data augmentation with a specific learning objective that they construct for their segmentation task, this objective is not applicable for classification. The standard objective (2.2) is used in this work.

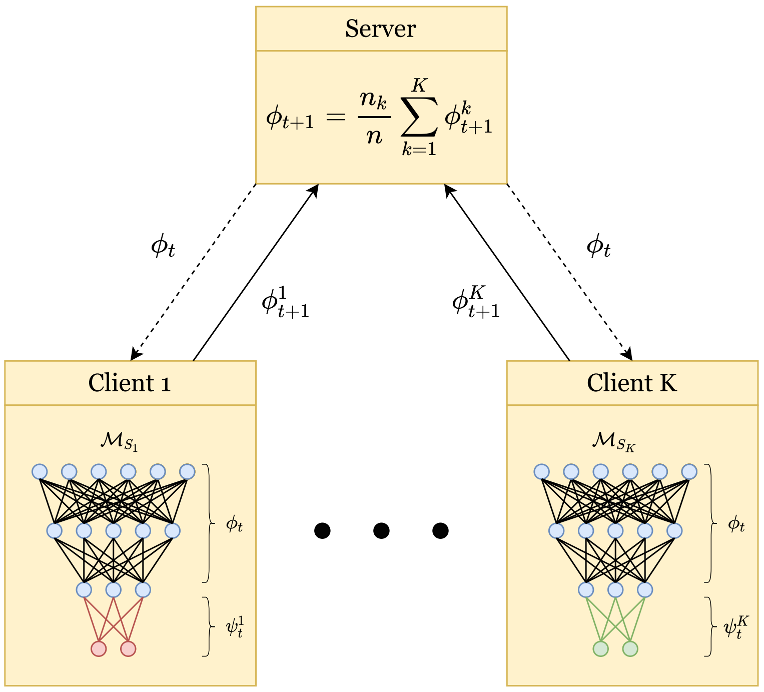


Figure 4.2.: Overview of the workflow of FedRep. Figure adapted from [48].

4.1.7. FedRep

Federated Representation Learning (FedRep) [48] takes a different approach as a K-Model personalization technique. An overview is given in Figure 4.2. The ML model parameters ω are split into a representation ϕ and a head ψ. The model output function becomes $f_\omega(x) = f_\psi \circ f_\phi(x)$. The authors draw upon recent success from centralized learning, where a common representation can be leveraged for different tasks [49].

While the server aggregates and copies a shared feature representation ϕ for all clients, each client also trains its own local head ψ_k to be able to adapt to its own task, which is shaped by the local data. In contrast to the other learning algorithms, the local updates are not computed with respect to all model parameters in one step. Instead, the parameters belonging to the local head and those belonging to the representation are trained in two separate steps. With this, one can choose a different number of local epochs for the head (E_{head}) and the representation (E_{repr}). Only the representation update is sent to the server, while the head is kept private. At inference time, each client uses the global representation paired with its own head to perform predictions, $f_{\psi_k} \circ f_\phi(x)$, resulting in K inference models. FedRep introduces a decision process about which layers are part of the representation and which are part of the head.

Similar to the LOT, computing K models with FedRep leads to an ALTV and an EGOV. Since in FedRep, the models are computed with the intention of using one model for each client data, only the EGOV is considered in the evaluations of this work. The overall performance is, similar to LOT, computed as the WAVG of the per-client scores.

4.2. Metrics

4.2.1. Performance

To evaluate the ability of an ML model to perform a prediction task, a large number of metrics exist. For classification, the Precision, Recall, and F1-Score are prominent choices as performance measures. These metrics are based on the confusion matrix of the ground truth labels and the labels predicted by the ML model. The labels are binary and describe if an object corresponding to the label is present (1) or not present (0). An overview is given in Figure 4.3. The values of a pair of a true label and a predicted label can thereby lead to four different outcomes, called True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN).

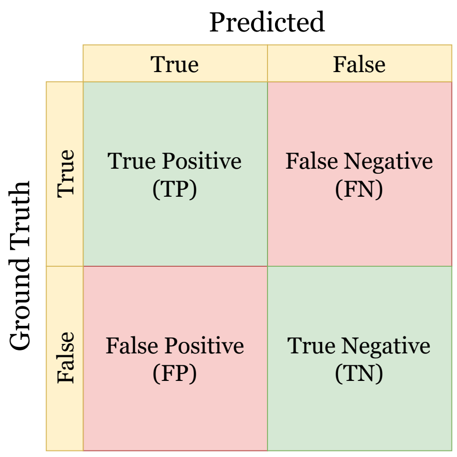


Figure 4.3.: Confusion matrix for a ground truth and a predicted binary label.

The Precision is the proportion of predicted ground truth positives from all predicted positives and computes to:

$$Precision = \frac{TP}{TP + FP}. \tag{4.16}$$

The Recall is the proportion of predicted ground truth positives from all ground truth positives and computes to:

$$Recall = \frac{TP}{TP + FN}. \tag{4.17}$$

The F1-Score is a combination of the Precision and the Recall and is a specification of a more general F-Score metric. The 1 specifies a balanced weighting of the Precision and the Recall. It computes to:

$$F1\text{-}Score = \frac{2 * Precision * Recall}{Precision + Recall}. \tag{4.18}$$

All three metrics are in the range of [0, 1], with 1 being the best.

In multi-label classification, there are different possibilities to factor in each specific label. Each label can be evaluated separately in order to find labels on which the model performs better or worse, and the corresponding results are referred to as *individual scores*. The unweighted average of the individual scores is further referred to as the *macro score*.

In an FLS, since the data is split amongst clients, there are datasets for each client. When evaluating the metrics on each client test dataset separately, the results are referred to as *local scores*. If required, these local scores can be determined at each client separately and then made accessible to the server. When evaluating over all test datasets combined, the result is referred to as the *global score*. This score can not be determined in a real FLS due to data privacy. Here, only a potentially weighted average over the local scores can be computed.

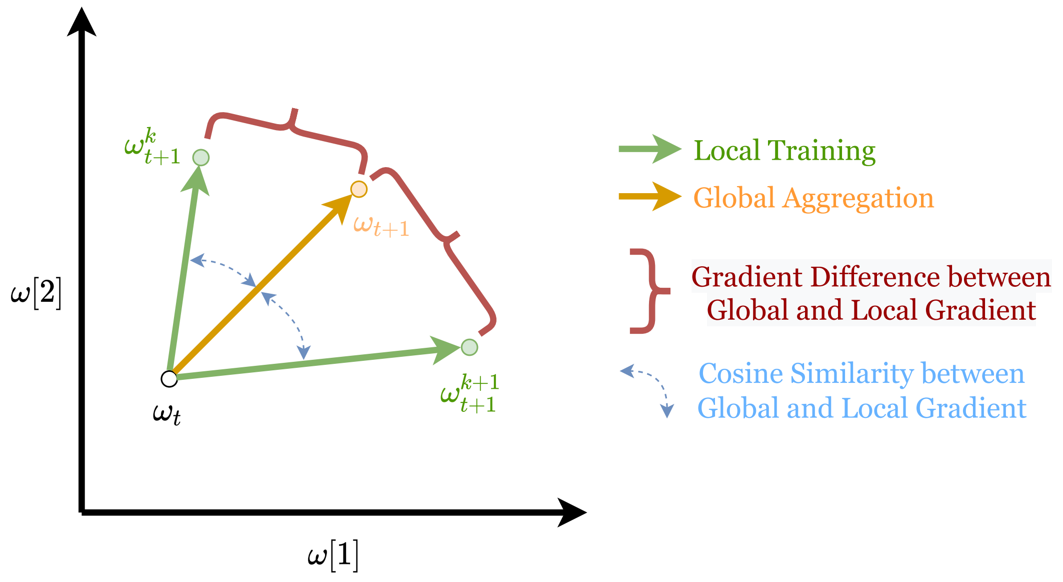


Figure 4.4.: Visualization of the considered gradient statistics by projecting the gradients onto the plane of two arbitrary model parameters $\omega[1]$ and $\omega[2]$.

4.2.2. Gradient Information

The cosine similarity is a measure of the similarity of two vectors. For n-sized vectors a and b, it is defined as:

$$CosSim(a,b) = \frac{a \cdot b}{\|a\| \, \|b\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}. \tag{4.19}$$

It ranges from -1 meaning exactly opposite orientation, to 1 meaning identical orientation. In this work, it is used to measure the similarity of local gradient vectors with the aggregated global gradient vector, i.e., $CosSim\left(\omega_{t+1}^{k}, \omega_{t+1}\right)$.

Furthermore, in this work, the squared length of the difference between two vectors is used as a measure for the distance between gradient updates. More specifically, similar to the works in [29], this distance is computed between each of the local updates and the combined global update to evaluate the amount of deviation of the local gradients from the global update, i.e.,

$$GradDiff\left(\omega_{t+1}^{k}, \omega_{t+1}\right) = \left\|\omega_{t+1}^{k} - \omega_{t+1}\right\|^2. \tag{4.20}$$

The WAVG of these gradient differences over all clients $k \in [1, K]$ corresponds to the variance of local gradients used in [29] to estimate local dissimilarities. An overview of the considered gradient information is given in Figure 4.4.

4.3. Loss Function

The Binary Cross Entropy (BCE) Loss is a loss that is commonly used for classification tasks. It takes the predicted output probabilities of an ML model and measures how distant the probability is from the ground truth binary label. Therefore, the loss is small when the output probability is very low and the ground truth label is 0, or if the probability is high and the ground truth label is 1.

For multi-label classification, the BCE Loss over a batch of B examples for L label classes, between the matrix of output probabilities $f_\omega(x)$ and the binary target matrix y, is defined as:

$$BCE(f_\omega(x), y) = -\frac{1}{ML}\sum_{i}^{M}\sum_{j}^{L} p_j y_{i,j}\log(f_\omega(x)_{i,j}) + (1 - y_{i,j})\log(1 - f_\omega(x)_{i,j}), \quad (4.21)$$

where p_j is the positive weight of class label j. The positive weight of a label j is the rate of negative and positive examples of that class in the train dataset, i.e.

$$p_j = \frac{\sum_i^n 1 - y_{i,j}}{\sum_i^n y_{i,j}}, \quad (4.22)$$

where n is the size of the considered dataset. Using positive weights leads to the classes with a low amount of occurrences being valued more during the training.

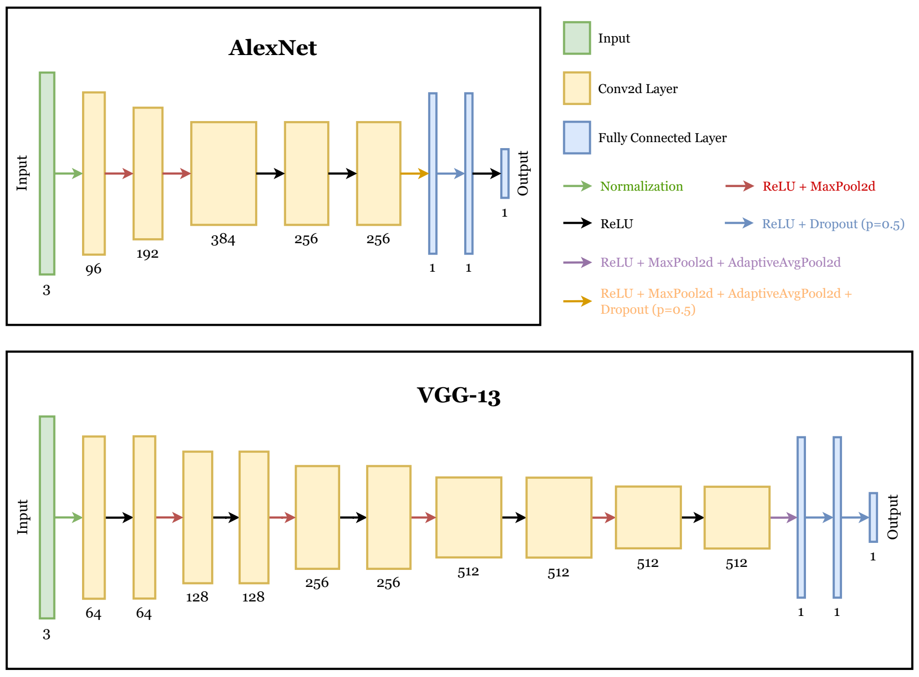


Figure 4.5.: The architectures of the ML models. The number of channels is given below each layer.

4.4. Models

In this section, the ML models used in the experimental setup are presented. An overview of the architectures is given in Figure 4.5.

4.4.1. AlexNet

An AlexNet [50] is a CNN published in 2012. It won the ImageNet [51] competition [52] of the same year, clearly outperforming all other contending models up to that point in time. It consists of five convolutional layers, where some are followed by a max-pooling layer, and three fully connected layers. The main features of the AlexNet that led to its success are the adoption of Rectified Linear Units (ReLUs) as the activation function, having overlapping pooling areas, and the usage of multiple GPUs for training. The latter allowed for both a larger model size and faster training times. The publication of the AlexNet sparked the development of many other models that build upon the intuitions gained with the AlexNet. To this day, the AlexNet, with its simple model architecture, is still used as a baseline for image classification tasks.

4.4.2. VGG

One of the spiritual successors of the AlexNet is the VGG-Network Architecture [53] proposed in 2014. The VGG-Network has different variants: VGG-11, VGG-13, VGG-16, and VGG-19, where the number indicates the number of hidden layers. Similar to the AlexNet, the VGG-19 won the ImageNet competition in the same year of publication [54].

Compared to the AlexNet, it reduces the filter size, leading to fewer model parameters, allowing for a larger number of layers and a more discriminative predictive function. Similar to the AlexNet, it uses ReLU activation functions.

5. Methods

5.1. Datasets

We empirically test the presented learning algorithms by training models on the combination of two medical image datasets, namely the HeiChole [55] and the Cholec80 [56] datasets. Both datasets contain videos of minimally invasive cholecystectomy surgeries captured by an endoscope. During a minimally invasive cholecystectomy, the gallbladder is removed with only small incisions to the body.

HeiChole consists of 33 videos captured at three different surgical centers: Heidelberg University Hospital, Salem Hospital (Heidelberg), and GRN-Hospital Sinsheim. The publicly available subset of 24 videos intended as the training set for models in the Surgical Workflow and Skill Analysis Challenge [57], which is part of the Endoscopic Vision Challenge 2019 [58], is used. This subset contains 12 videos from Heidelberg University Hospital and 12 videos from Salem Hospital. Cholec80 consists of 80 laparoscopic cholecystectomy videos captured at the Strasbourg University Hospital. All videos were anonymized to comply with the GDPR.

The original videos are captured at 25 fps or 50 fps, however, the datasets are downsampled to 1 fps. The images or frames contain surgical tools that can be categorized into 7 tool classes. Each class can contain multiple different unique tools. However, those share their functionality. A frame can contain an arbitrary number of tool classes. Hence, it can contain no tools at all or multiple tools simultaneously. As tools are sometimes tricky to visually detect, a tool is labeled as present if at least half the tool tip is visible in Cholec80 and if the tool tip was visible in the frame sequence beforehand in HeiChole, with exceptions being explained in [55]. Images that were replaced with white frames in HeiChole, as described in [55], are not considered in the experiments.

Even though the labelings for both datasets are very similar, they are not the same. To assure a consistent labeling, the labels of Cholec80 are mapped to the labels of HeiChole

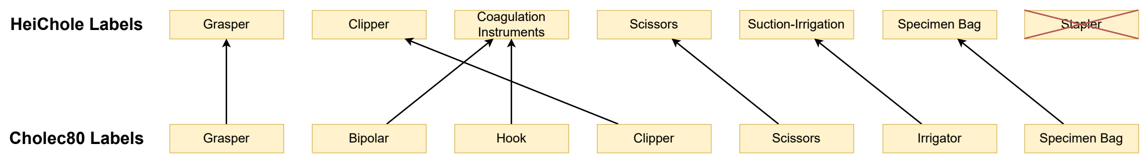


Figure 5.1.: Label mapping from Cholec80 to HeiChole.

according to Figure 5.1. Both the Cholec80 labels ”Bipolar” and ”Hook” are categorized as ”Coagulation Instrument” in HeiChole. All other tool categories are handled similarly in both datasets. The tool ”Stapler”, only contained in HeiChole, is not considered due to its deficient number and problems arising in the context with positive weights, as discussed in section 7.3.

Since the sample space S is the set of sequential frames captured by a single camera, the data partitioning is horizontal. The task of the ML model is to predict the presence of each of the surgical tools in the frames. That poses a multi-label classification problem.

5.1.1. Dataset Split

Starting from the centrally stored combined dataset, the data can be artificially split to simulate clients with different data cohorts and data distributions. The data can then be further divided into local train and test datasets for each client.

Since the HeiChole training set originates from two surgical centers, we use precisely this criteria to split the data resulting in client *HeiChole1* (HEI1) with 12 videos from Heidelberg University Hospital and client *HeiChole2* (HEI2) with 12 videos from Salem Hospital. Note that this naming convention is inconsistent with the naming in [55] where HeiChole-1 refers to the first surgical video. For both clients, three surgical videos are sampled randomly and used as the test dataset of the respective client. The remaining videos are used for training.

On the other hand, because Cholec80 is constructed from videos solely from Strasbourg University Hospital and anonymization makes it impossible to split the data according to the operating surgeon, we require a different splitting criterion here. To introduce additional effects of statistical data heterogeneity on the FL process, we split the data by means of the size of the black edges at the left and right image border. The frames are categorized into three classes: none or small-sized black edge, intermediately-sized black edge, and large-sized black edge. The respective clients are referred to as *Cholec1* (CHO1; large edge), *Cholec2* (CHO2; intermediate edge), *Cholec3* (CHO3; small edge). In some cases, the size of the black edge changes throughout the surgery. In this case, we consider the size of the black edge that occurs most. For division into train and test set, the same split as in [59] is used.

For differentiation, we call the LOTs performed on the different datasets LHEI1, LHEI2, LCHO1, LCHO2, and LCHO3, respectively.

5.1.2. Data Analysis

First, the data residing at the clients are statistically analyzed more thoroughly. As pointed out in subsection 2.4.3, this is not possible in a real-world FLS as it requires direct access to the private client data. However, in order to compare the learning algorithms, the distributions of the data and the existing data heterogeneities need to be considered.

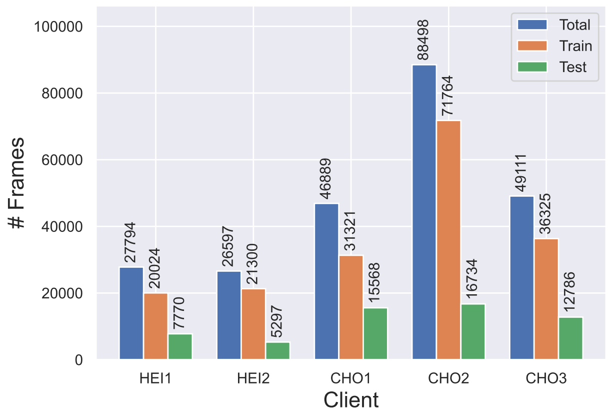


Figure 5.2.: Number of frames in the train and test datasets for every client.

The data amounts resulting from the dataset split can be seen in Figure 5.2. The clients originating from the HeiChole dataset possess less data than those originating from Cholec80. For Cholec80, CHO2 holds almost double the amount of data of CHO1 and CHO3. Therefore, a noticeable amount of data quantity skew is present for the clients. The percentage of test data compared to the total data lies between 19% (CHO2) and 33% (CHO1).

In Figure 5.3, the means over all images in the respective client datasets are shown. The mean images of CHO1, CHO2, and CHO3 underline the described dataset split. For HeiChole, HEI1 has intermediately sized black edges, while HEI2 includes no black edges. Figure 5.4 shows the violin plot of the mean client images. Besides the black edges for the respective clients, the HeiChole images have higher color intensities. Furthermore, there are peaks for white and black pixels in HEI2, located in the information bar at the bottom of the frames.

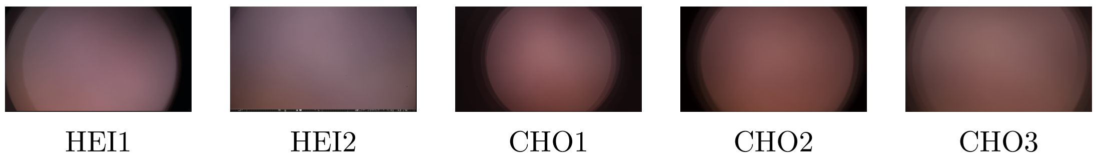


Figure 5.3.: Mean images over each respective dataset.

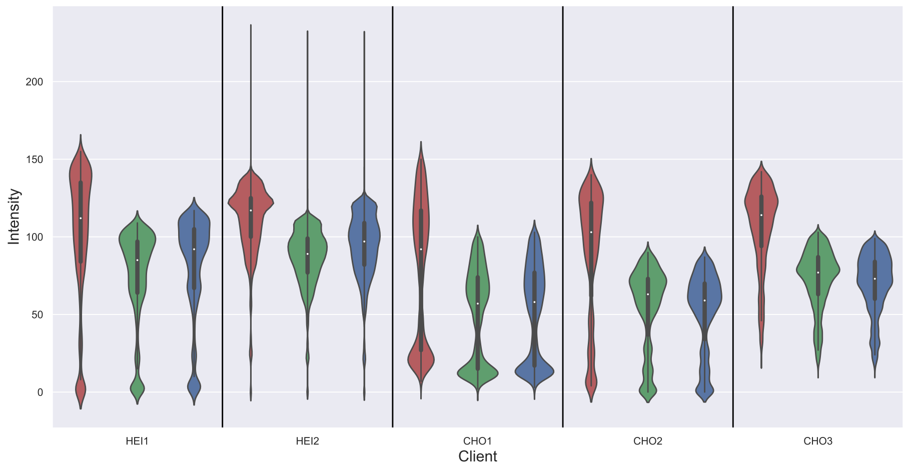


Figure 5.4.: Violin Plot of the mean RGB images of all clients.

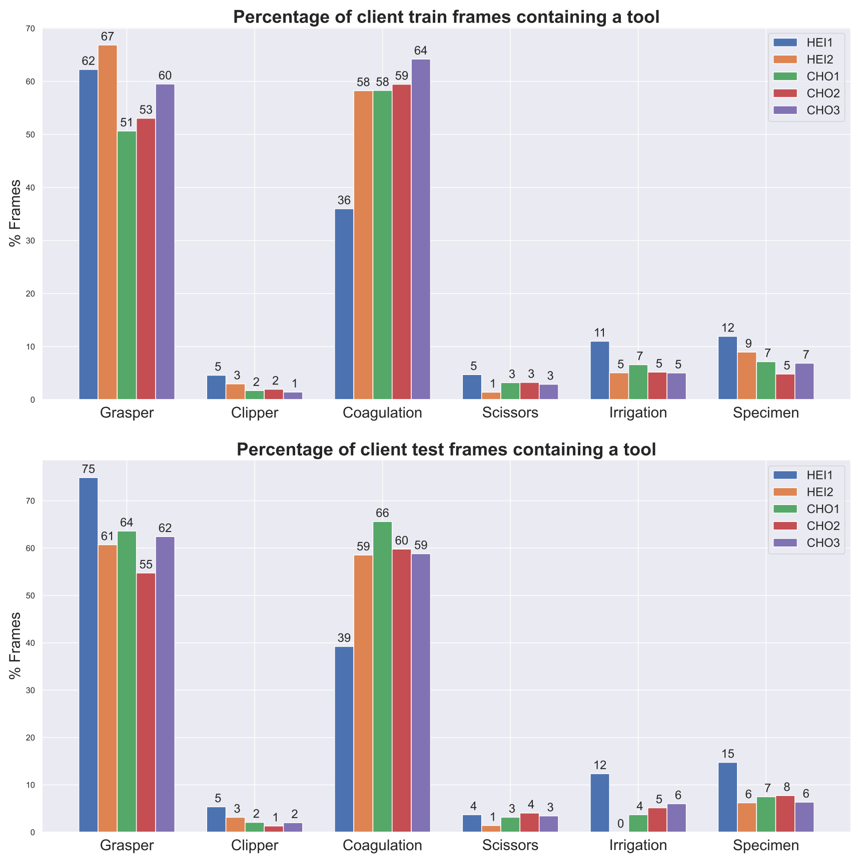


Figure 5.5.: Label percentages of all clients.

Figure 5.5 shows the percentage of frames that contain the individual tools for each client. The tools "Grasper" and "Coagulation Instrument" are represented most for all clients, occurring in more than 50% of the images in all training and test sets. Exceptions are the train and test set of HEI1, which contain the "Coagulation Instrument" in less than 40% of the frames. Instead, HEI1 contains higher percentages of all other tool categories than the other clients. However, the other tools generally appear only in lower numbers than those mentioned above. The maximum is the "Specimen Bag" with a percentage of 15% of the frames in the test set of HEI1. Hence, a significant skew between different tools is present. The label distribution skew across different clients is smaller, where HEI1 shows the highest deviation.

5.2. Federated Training

5.2.1. Models

Given the aforementioned datasets, models are trained for the multi-label classification problem. As ML models, an AlexNet [50] and a VGG-13 [53], which are common models in image classification, are used. From all VGG variants, the VGG-13 performed best in preliminary studies, with the VGG-16 reaching similar performances but with higher computation times.

Different ResNet [60] architectures, such as the ResNet18 and the ResNet50, which were commonly used in the Surgical Workflow and Skill Analysis Challenge [55, 57], were also considered. However, they are not used for further evaluation, as they require an additional deliberation of an averaging scheme for the inherent statistics of Batch Normalization layers, as discussed in [61]. An alternative would be to substitute the Batch Normalization with a different type of layer. For example, the authors in [45] use a Group Normalization layer instead. Aggregating all ResNet parameters naively led to poor performances in the preliminary tests.

5.2.2. Implementation

The training schemes are implemented with the PyTorch library [62]. The experiments are run as single-computer simulations. Hence, the client updates are computed sequentially but aggregated only once all client trainings are finished. The pretrained PyTorch implementations of the models mentioned above are used, where the pretraining is performed on the ImageNet dataset [51]. All images are normalized, according to the recommendation for the pretraining, with $mean = [0.485, 0.456, 0.406]$ and $std = [0.229, 0.224, 0.225]$. Furthermore, all images are resized to $(width, height) = (240, 135)$.

For FedDG we precompute and store the amplitude spectrum signals for all images in the datasets.

5.2.3. Hyperparameters

The AlexNet and VGG-13 are trained with all learning algorithms presented in section 4.1. For both models, $E_{global} = 50$ global epochs of training are run. The AlexNet is trained with a batch size of 128. Due to the higher amount of model parameters of the VGG-13, the batch size is reduced to 16 here. The same batch size is used for the CENT, the LOT and the local updates of FL algorithms, i.e., clients are using the same local batch size B as globally trained models. The loss function $\ell(f_\omega(x), y)$, used for all updates, is a BCELoss [63] with positive weights, as presented in section 4.3. Therefore, a Sigmoid activation layer is applied to the output of the models.

For all FLTs, in each communication round, all clients are tasked to perform local updates, i.e. $C = 1.0$. Experiments regarding the participation percentage in a FLS, like those in [45], indicate that higher levels of participation lead to faster convergence. Also, a higher participation percentage allows for higher server learning rates, as the variance reducing effect of the global aggregation increases with more active clients [38]. Furthermore, there are typically few clients in the cross-silo MMFLS setting, which excludes technical difficulties that would arise with larger amounts of clients, such as communication loads. The negative effects of stragglers (clients with higher processing time) are rather small, as there is typically no time requirement for the model training.

To be comparable with the CENT, each client performs only one local training epoch $E_{local} = 1$ in a communication round. The server aggregation of local FL updates is weighted by the client data amounts (WAVG). Furthermore, the global aggregation updates of all FL algorithms are rewritten as explained in subsection 4.1.3, which allows the usage of arbitrary optimizers. For all trainings, AdamW is used as the global optimizer. As the local optimizer, which is only present in FL algorithms, SGD without momentum or weight decay is used in order to introduce no unexpected behavior in the local objective. That also removes the requirement to decide how momentum is carried over into each new communication round. The local learning rate is set to 1e-3 for all clients. The global learning rate is set to 1e-3 in all FLTs. A learning rate of 1e-4 is used for the updates in LOTs. For convenience, those are also referred to as global learning rates. The global learning rate for the CENT is set to 1e-4 for the AlexNet and 1e-5 for the VGG-13. A one-cycle learning rate schedule [64], with the mentioned learning rates as maximal learning rates, is used on the global learning rates in CENT and LOTs, and used on the local learning rates in FLTs. For the latter, by applying the schedule to the local learning rates instead of the global ones, this work follows the argumentation in [27], in that the local learning rate needs to decay for a convergence guarantee of FedAvg. This happens in the second half of the learning rate schedule. The previously mentioned optimizer and learning rate settings performed best in the preliminary tests.

Regarding the hyperparameters introduced by the FL algorithms, μ, which is used by FedProx and FedDyn, is set to 1e-2. For FedRep, one head epoch, i.e. $E_{head} = 1$, and one local representation epoch, i.e. $E_{repr} = 1$, are run in that order. For that, for both the AlexNet and the VGG-13 the three final fully-connected layers, as in Figure 4.5, are used as the local head ψ. All prior convolutional layers for both networks are considered as the representation ϕ.

5.3. Evaluations

The performance statistics presented in subsection 4.2.1, i.e., the Precision, Recall, and F1-Score, on each separate client test dataset as well as the combined global dataset, are observed across all algorithms, epochs, and tools. Furthermore, information about the local gradient updates, as explained in subsection 4.2.2, is observed for all clients. For FedRep, the gradient information is only computed on the representation updates ϕ_{t+1}^{k}, which are shared with the server. While the observation of the lengths of the gradient differences as a training statistic was introduced in [29], to our knowledge, the cosine similarity has not been used as a metric to evaluate the training behavior in a FLS in any other work before.

Additional to the trainings, statistical tests are performed on the resulting F1-Score performances in order to determine the statistical significance of the differences between different learning approaches. For that, a Mann-Whitney U test [65] is used for three different alternative hypotheses. With regards to the F1-Score performances, the following alternative hypotheses are investigated:

1. ALTV < FLT:
   The performances of LOTs with an ALTV are stochstically less than the performances of a training with a given FL algorithm.
2. EGOV < FLT:
   The performances of LOTs with an EGOV are stochstically less than the performances of a training with a given FL algorithm.
3. FLT < CENT:
   The performances of a training with a given FL algorithm are stochstically less than the performances of a CENT.

These tests are evaluated for each FL algorithm separately. For the test statistic, the final F1-Scores across all clients and all tools are used. In this work, a difference in distributions is defined to be statistically significant if a p-value less than 0.05, i.e., a confidence level of 95%, is reached.

Finally, to estimate the influence of the client data amounts on the FL process, observations on trainings with an unweighted global aggregation are made with FedAvg as an example.

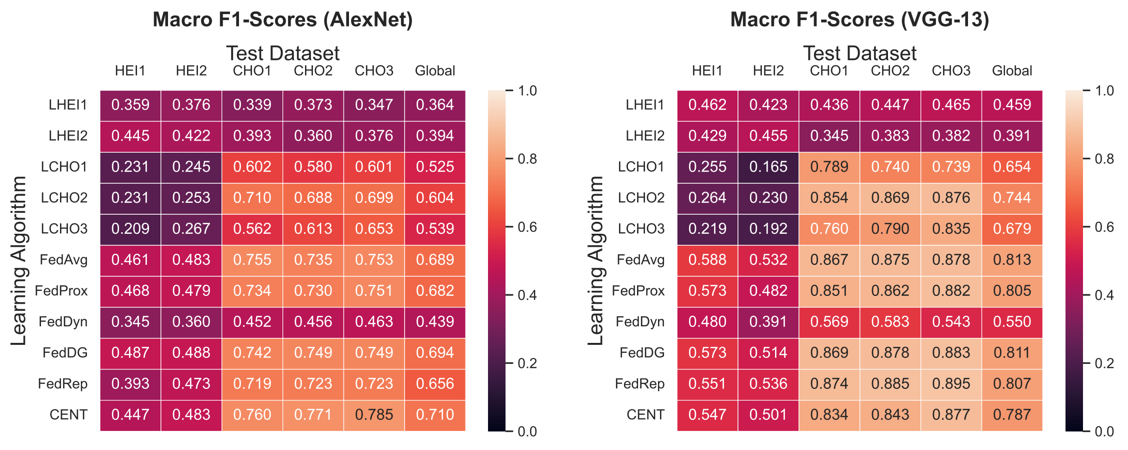
6. Results

6.1. Performance Metrics

In Figure 6.1, the final F1-Scores of all learning algorithms are presented, including both the local test scores in the first five columns and the global test score, in the final column. The scores of all LOTs learning algorithms are reported individually. In this heatmap, the ALTV of the LOTs corresponds to the final column, i.e., the global scores, of the first five rows. The EGOV is given as the diagonal of the first five rows/columns. When looking at the global F1-Scores of FL algorithms, FedDG has the highest performance for the AlexNet and FedAvg has the highest performance for the VGG-13.

To be able to compare the FL algorithms with the baselines across different clients, a visual comparison is shown in Figure 6.2. Using the same data, the performances and the resulting performance gains/losses of each learning algorithm, compared to a LOT with an ALTV, are given for each client test set in the appendix in Table A.1. Additionally, the means and variances across all learning algorithms or across all FL algorithms and all test sets, respectively, are given here.

Generally, a clear distinction between the subsets of HeiChole and Cholec80 can be made. The performances on the HeiChole subsets are substantially lower than those of Cholec80.



Macro F1-Scores (AlexNet)

Learning Algorithm	HEI1	HEI2	CHO1	CHO2	CHO3	Global
LHEI1	0.359	0.376	0.339	0.373	0.347	0.364
LHEI2	0.445	0.422	0.393	0.360	0.376	0.394
LCHO1	0.231	0.245	0.602	0.580	0.601	0.525
LCHO2	0.231	0.253	0.710	0.688	0.699	0.604
LCHO3	0.209	0.267	0.562	0.613	0.653	0.539
FedAvg	0.461	0.483	0.755	0.735	0.753	0.689
FedProx	0.468	0.479	0.734	0.730	0.751	0.682
FedDyn	0.345	0.360	0.452	0.456	0.463	0.439
FedDG	0.487	0.488	0.742	0.749	0.749	0.694
FedRep	0.393	0.473	0.719	0.723	0.723	0.656
CENT	0.447	0.483	0.760	0.771	0.785	0.710

Macro F1-Scores (VGG-13)

Learning Algorithm	HEI1	HEI2	CHO1	CHO2	CHO3	Global
LHEI1	0.462	0.423	0.436	0.447	0.465	0.459
LHEI2	0.429	0.455	0.345	0.383	0.382	0.391
LCHO1	0.255	0.165	0.789	0.740	0.739	0.654
LCHO2	0.264	0.230	0.854	0.869	0.876	0.744
LCHO3	0.219	0.192	0.760	0.790	0.835	0.679
FedAvg	0.588	0.532	0.867	0.875	0.878	0.813
FedProx	0.573	0.482	0.851	0.862	0.882	0.805
FedDyn	0.480	0.391	0.569	0.583	0.543	0.550
FedDG	0.573	0.514	0.869	0.878	0.883	0.811
FedRep	0.551	0.536	0.874	0.885	0.895	0.807
CENT	0.547	0.501	0.834	0.843	0.877	0.787

Figure 6.1.: Final F1-Scores for single client test datasets (HEI1-CHO3) and the combined test dataset (Global).

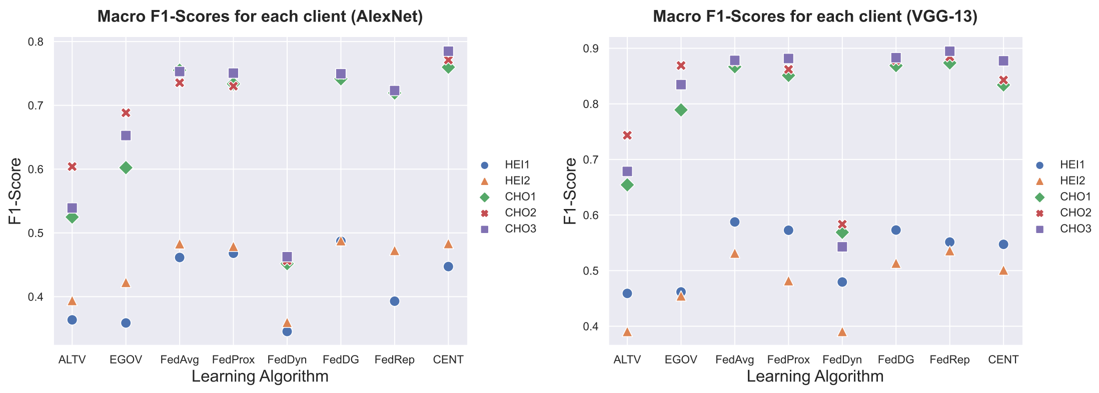


Figure 6.2.: Per-client comparison of all algorithms with the baselines.

The results show, however, that for every client, all FL algorithms, except FedDyn, yield improvements over the LOT with an ALTV. Although not as distinctly as for the ALTV, this statement also holds in all but one case for the EGOV. With an EGOV and for the VGG-13, the F1-Score on the test dataset of CHO2 drops by 0.007 for FedProx, compared to the corresponding local training LCHO2. The highest improvements, in terms of mean F1-Score, over the LOTs are seen from FedDG (AlexNet), and FedAvg and FedRep (VGG-13). The highest variances of performances between different clients are seen by the CENT and FedRep (AlexNet), and EGOV (VGG-13). On the other hand, when considering statistics across all learning algorithms or only across all FL algorithms, generally, the clients originating from Cholec80 exhibit higher performance benefits, and the variances of these clients are higher as well. For both models, the client with the highest mean performance gain from all FL algorithms is CHO1.

Regarding the CENT, it shows the highest performances for the AlexNet, but only with a slight improvement over the best performing FL algorithms. For the VGG-13, the best performing FL algorithms exceed the F1-Scores of the CENT, which was not expected and is further discussed in section 7.3.

Learning Algorithm	ALTV < FLT	EGOV < FLT	FLT < CENT
FedAvg	5.60e-7	0.06171	0.52824
FedProx	2.11e-6	0.08014	0.49267
FedDyn	0.67091	0.99909	3.64e-6
FedDG	5.64e-7	0.06768	0.50104
FedRep	2.54e-6	0.09340	0.44884

Table 6.1.: p-values of the three performed statistical Mann-Whitney U tests. The alternative hypotheses are that the former distribution are stochastically less than the latter.

To determine the statistical significance of the observed results, the outcomes of the performed statistical tests are shown in Table 6.1. In the first two columns, the significance of the improvement of all FL algorithms over the LOTs is evaluated. The final column is considered with the significance of the improvement of the CENT over the FLTs. With a confidence level of 95%, all improvements of FL algorithms over the LOTs with an ALTV are significant, except for FedDyn. The improvements of FL algorithms over LOTs with an EGOV are not significant, although FedAvg, FedProx, FedDG and FedRep all get a p-value close to the significance level. Therefore, it was observed that there is a considerable difference between the two interpretation variants of LOTs. The improvement of the CENT over the FLTs, or equivalently the decreases in performance of the FL algorithms

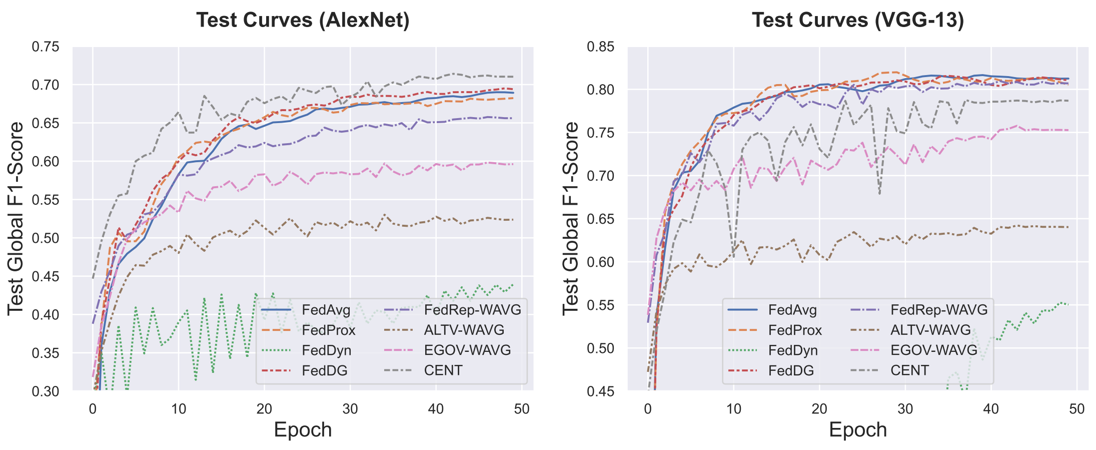


Figure 6.3.: Macro F1-Scores on global test dataset over training process for all learning algorithms. For the ALTV, EGOV and FedRep, the global score is computed as the WAVG of the local scores.

compared to the CENT, are clearly not significant, with p-values around 0.5. For FedDyn, the performances are not significantly better than both the LOT interpretations, and the performances are significantly worse than the ones of the CENT.

Overall, the margins between the best performing FLTs are minimal. This can also be seen in Figure 6.3. The figure shows the macro F1-Scores on the global test set throughout the training process for all learning algorithms. The global scores of the ALTV, EGOV and FedRep are computed as the WAVG of the local scores. For the AlexNet the final performances of FedAvg, FedProx and FedDG lie within a small margin of 0.006. For the VGG-13, the margin is 0.007 and also includes the performance of FedRep. The clear difference between the ALTV and the EGOV can also be observed. The LOTs with an EGOV achieve a WAVG F1-Score of 0.59 (AlexNet) and a WAVG F1-Score of 0.75 (VGG-13). On the other hand, with the ALTV, the F1-Score drops about 0.07 (AlexNet) and 0.11 (VGG-13) when compared to the EGOV.

Also seen in Figure 6.3, all of the FLTs, except FedDyn, have a steep slope in the beginning which flattens out after 10 epochs. The F1-Score of the CENT rises faster at the beginning for the AlexNet. For the VGG-13, it falls behind the best FL algorithms right from the start. Lastly, the performance of FedDyn oscillates at a low level for both models.

To summarize the test curves and to be able to compare the models, the mean and standard deviation of the curves of all algorithms for both models are shown in the appendix in Figure A.1. Both curves converge at the end. The VGG-13 achieves a higher overall performance, however, with a higher standard deviation during the training.

Figure A.2 in the appendix shows the effect of a weighted and an unweighted aggregation scheme for FedAvg on all clients. There is no major difference in performance for any client for both models.

Additionally, investigations are made for each tool separately. Similar as for the clients, the performances of all learning algorithms for different tools and the resulting performance changes are shown in the appendix in Table A.2. A visual presentation of all learning algorithms on the different tools is given in Figure 6.4. The most prominent tools in the dataset, i.e., ”Grasper” and ”Coagulation Instrument”, are predicted correctly most consistently in almost all cases. However, the performance gains compared to a LOT with an ALTV are the smallest for these tools. Instead, higher performance gains are observed for all lesser occurring tools, where the largest benefit is seen for the ”Clipper”. The latter

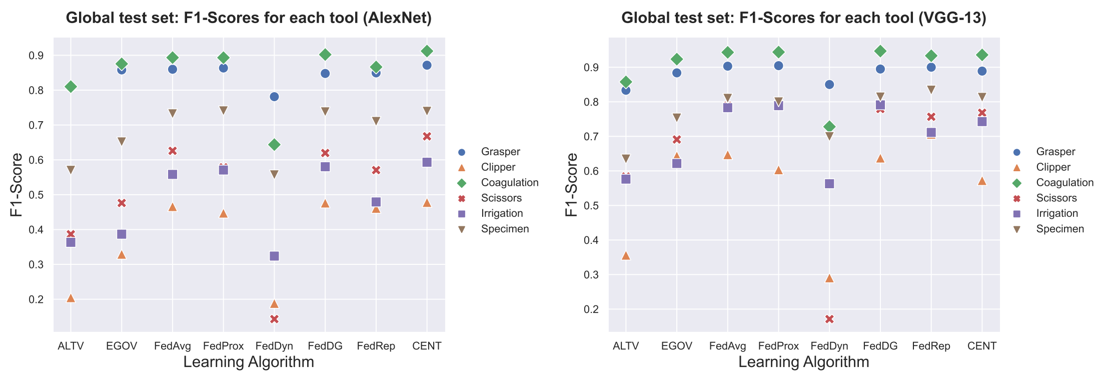


Figure 6.4.: Per-tool comparison of all algorithms with the baselines.

results from a meager performance value of the ALTV baseline for the ”Clipper” for both models, where the EGOV baseline yields a clear improvement in that regard with the VGG-13. Furthermore, the variances for lesser occurring tools are higher. For HEI2, the F1-Score for the ”Suction-Irrigation” tool is 0, as there are no occurrences of the tool in the client test dataset. That negatively affects the macro score on the test dataset of HEI2, as further discussed in subsection 7.1.5.

An alternative way to handle the missing tool in the test dataset of HEI2 is not to consider the tool score of the missing tool in its macro score. For this case, the adapted local and global scores on each client are given in the appendix in Figure A.3.

Lastly, for a complete overview of the results, in the appendix in Figure A.4, the per-tool F1-Scores are presented for each client test dataset separately, where for each dataset, the corresponding LOT of that client, all FL algorithms and the CENT are considered. That summarizes all F1-Scores across all learning algorithms, clients, and tools, except that for the LOTs only the EGOV, i.e., each client on its own test dataset, is considered.

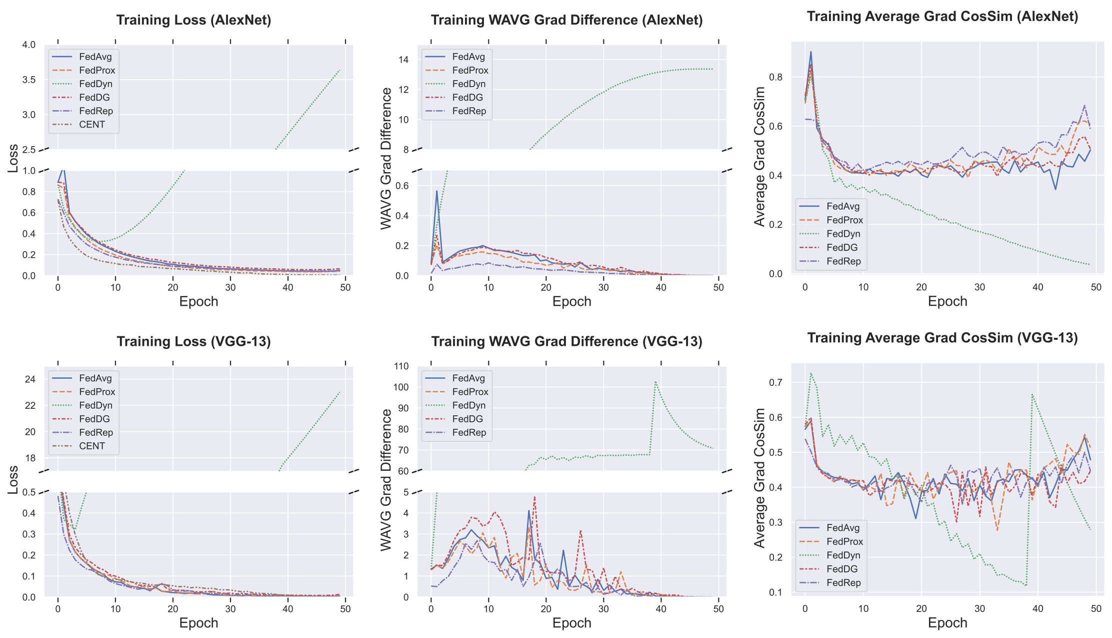


Figure 6.5.: FL training statistics.

6.2. Training Statistics

In order to observe the convergence behavior of the FL methods in more detail, information about the losses and gradients during training is given in Figure 6.5. The graphs on the left show the average training losses of the CENT and all FLTs across every client. All FL algorithms, except FedDyn, show a similar convergence behavior to the one of the CENT. On the contrary, the loss of FedDyn starts to diverge after a few epochs.

The center column shows the WAVG of the squared lengths of the differences between the global and the local updates, as explained in subsection 4.2.2. For all FL algorithms, except FedDyn, the gradient lengths peak in the beginning and start to decrease after 1 epoch (AlexNet) and around 10 epochs (VGG-13). Afterward, they go down to zero, whereas for the VGG-13, this happens discontinuously. The gradient difference of FedDyn rises to and plateaus at a high number for both models.

In the right plot, one can see that all FL algorithms, except FedDyn, also show similar behavior with respect to the average cosine similarity between the local gradient updates and the resulting global gradient update. Starting at a similarity of around 0.8 (AlexNet) or 0.6 (VGG-13), it drops to 0.4 before oscillating towards a cosine similarity of around 0.5. The average cosine similarities of FedDyn start high before decreasing down to 0.

Figure 6.6 shows how, in the case of FedAvg, the different client updates contribute to the average cosine similarity. The updates of HEI1 demonstrate the highest similarity with the global updates, followed by CHO2. The other client updates show less similarity with the global gradient.

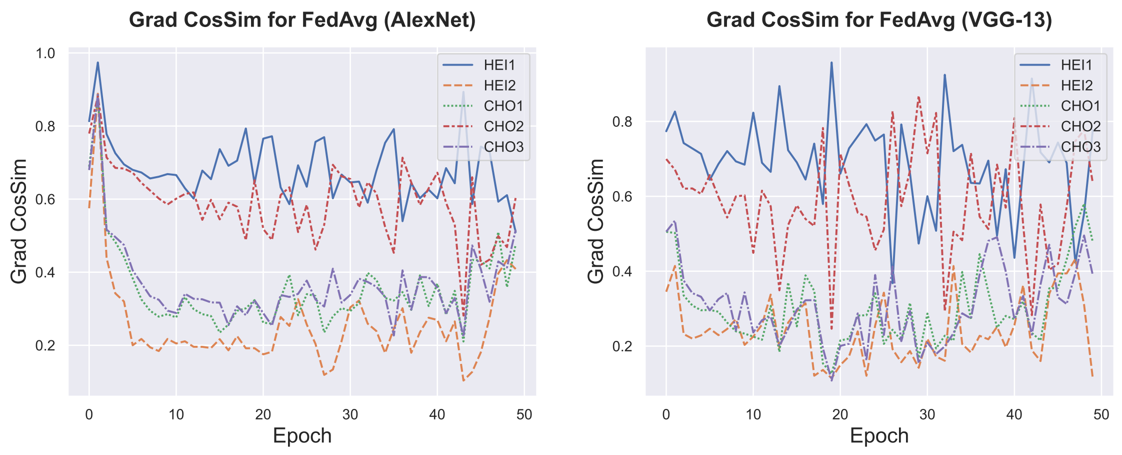


Figure 6.6.: Per-client cosine similarities between global update and local updates for FedAvg.

7. Discussion

As FedAvg, FedProx and FedDG show very similar results among all tested statistics, they are treated jointly, and the group of them is referred to as Fed**A**vg Fed**P**rox Fed**D**G (APD).

7.1. Federated Learning vs. Baselines

In the first subsections, the different algorithms' results are discussed and compared with each other and the given baselines. In subsection 7.1.2, properties that are not specific to any FL algorithm are discussed. Then, the outcomes of the experiments for different clients and individual tools are examined and interpreted in more detail. Finally, in subsection 7.1.6 this thesis comments on the differences between the ALTV and EGOV.

7.1.1. Comparison of Federated Learning Algorithms

7.1.1.1. APD

Generally, the three learning algorithms perform very well, almost matching the upper baseline of CENT with the AlexNet and exceeding it for the VGG-13. It was shown that the CENT holds no significant performance gain over the FLTs. Accordingly, they improve clearly over the LOTs, for both the ALTV and EGOV.

Altogether, this shows that these FL approaches are an excellent alternative to the traditional CENT for the given task. They enable for a ML model to be trained on data, residing privately at each client, without a major loss in model performance. If the possibility exists, centrally collecting all data can have an advantage in terms of performance in some cases. However, in the performed tests, this advantage was not significant, and it was only existent for the AlexNet.

For clients looking to improve with respect to the generalization capabilities of their private model, FL yields a significant potential for improvement. The improvement on each respective clients' own data set is not as impactful as for the generalization, but even here, in all but one case, the clients observed a performance benefit in the performed experiments, highly encouraging the use of these algorithms. The mentioned decrease in performance occurred for FedProx. While the overall performance was increased by FedProx, the performance on the test dataset of CHO2 decreased compared to the corresponding LOT with an EGOV. Generally, LCHO2 for the VGG-13 shows very high performances for the test datasets of all three clients originating from Cholec80, almost matching the results of all well-performing FL algorithms, which suggests that the risk for the decreased performance is more likely when a client is already able to perform very well on its own dataset, which is supported by the high amount of training data that is present at CHO2. Therefore, although rarely and to a small extent, an adverse change in performance on the own dataset is possible for a client participating in FL.

On the other hand, when taking into consideration only the results of the other two FL algorithms, there is no reason against a participation in a FLS from a performance standpoint.

7.1.1.2. FedDyn

In contrast to the other FL algorithms, FedDyn achieves no decent result in the performed experiments. Its performance is worse than the ones of the lower baselines, which can be interpreted as bad, considering the good results of all other FLTs. The model parameters are not able to converge to a minimum of the global objective, which is shown by the diverging loss. That can be attributed to the dynamic regularizer that was introduced in FedDyn, as described in subsection 4.1.5. While the proximal term introduced by FedProx, which is also present in FedDyn, leads to no significant change in the model performance, the inclusion of the dynamic regularizer, which is computed recursively, results in a diverging local objective (4.11). The average cosine similarities of the clients dropping to zero indicate that the local gradient updates are also diverging and are not finding any consensus on the parameter updates towards the end of the training. Therefore, FedDyn seems to be ineligible in the performed experiments. For this reason, it is not considered in the further discussions, and it is explicitly excluded when mentioning FL algorithms.

7.1.1.3. FedRep

FedRep is the only FL algorithm that shows slightly different results for the two types of ML models. While, concerning the global score, FedRep lies behind APD for the AlexNet, it performs bests for the VGG-13, together with FedAvg. That indicates that the approach is more susceptible to the used model than the other algorithms. Moreover, the choice of the representation and head, which also depends on the model, influences the performance. Since we use the same head for the AlexNet and VGG-13, and the VGG-13 has a larger model size, the VGG-13 has access to a larger and more complex representation. The lower performance for the AlexNet could result from a shared representation that does not have the required complexity to capture the heterogeneous data.

Still, similar to APD, FedRep shows good results overall. In the discussion about model performance, it exhibits no disadvantages for a participation in a federation, with a similar argumentation as in subsubsection 7.1.1.1.

Lastly, for FedRep, the local update consists of two separate update steps for the local head and the representation, respectively. Therefore, the client dataset needs to be propagated through the model twice in a local epoch. That leads to a considerably higher computation time. On the other side, the size of the representation update is smaller than for the other FL approaches.

7.1.2. General Findings

The overall findings of the differences between FL algorithms and the CENT baseline being low are compliant with the results of other works of FL in the medical field [13, 17, 41]. Regarding the decrease in performance on CHO2 for FedProx and the VGG-13, the results in [13] show similar cases of decreases for three out of ten of the clients' LOTs with an

EGOV. Similarly, in all other cases for the EGOV, and all cases for the ALTV, there are performance benefits.

Furthermore, the minor differences in performances between the algorithms of APD as well as FedRep show that the subsequent works, which build on the intuition of FedAvg, do not provide significant improvements for the model performance in this test case. Though, FedAvg does not leave possibilities for major improvements, as it already almost matches or exceeds the upper baseline. The test settings that are used in those works to show that the proposed algorithms can handle non-IID data better are often containing highly heterogeneous client data distributions. An example test case is to distribute the data of CIFAR-10 [66] so that each client only possesses examples with 1 or 2 of the 10 labels, resulting in highly skewed label distributions across different clients. The amount or type of client data heterogeneity in this thesis's experimental setup seems not to be enough to show similar effects. Given that all considered extensions of FedAvg require additional computational steps and show no actual performance increase over FedAvg, they seem to have no specific advantage over it for this task.

7.1.3. Differences between Clients

As shown in Figure 6.2, there exists a clear disparity between different clients in terms of F1-Score performance for all learning algorithms. More specifically, there is a clear difference between the test data subsets originating from HeiChole or Cholec80.

Firstly, this underlines the presence and impact of statistical data heterogeneity in the different datasets. In that regard, the resulting differences in the performances are not an attribute that is unique to FL, but instead also occur for all baselines. In fact, the variance between client performances is higher for the CENT than it is for any of the FLTs, in the case of the AlexNet. Hence, in that case, the performance gains for each client deviate the most when using the classical ML learning technique.

Secondly, although HeiChole was further split into the videos of Heidelberg University Hospital and Salem Hospital, and Cholec80 was further split by the extent of the black edge in the images, it is still easy to distinguish the original full datasets, when looking at the performances. Therefore, the underlying data distributions resulting from the protocols when taking the videos and collecting the datasets have a much stronger influence on the model performance than the one introduced by the data split. This indicates that the two hospitals that shared data for the HeiChole dataset have decisively different parameter settings and protocols for the data gathering and preprocessing process than the Strasbourg University Hospital, from where Cholec80 originates. This result affirms the existence of a MIDB in these datasets.

Thirdly, it can be observed that for these gaps in performance between clients originating from HeiChole and Cholec80, which are already present for the classical learning domains of LOTs and the CENT, the FL algorithms show no particular progress in closing these gaps. As mentioned in the results, the clients originating from Cholec80 have a higher mean performance gain than the ones from HeiChole, therefore widening the gap compared to a LOT with an ALTV. Regarding fairness, HEI1 and HEI2 are provided with fewer performance benefits. However, they also contribute less data to the federation.

This suggests that the client data amounts could be essential for the resulting client performances. An argument against the data amounts as a deciding factor for the different client performances is, however, that CHO1 and CHO3 show about double the performance increase of CHO2 for FL algorithms compared to LOTs with an ALTV, while these two clients have only half the amount of data of CHO2. Here, it can also be seen that CHO2 achieves higher performances in its LOT. This indicates that the higher amount of data of CHO2 already enables the client to create a local model that can get a better performance on its own test set. Meanwhile, this is less the case for CHO1 and CHO3. They can benefit more from participating in the federation, especially considering that they seem to have a related data distribution with CHO2. Hence, the results suggest that CHO1 and CHO3 are provided with more new relevant data than CHO2, while HEI1 and HEI2 can not benefit as much from this data as CHO1 and CHO3, which might be caused by the MIDB.

Another reason for the differences in performance could be the weighted aggregation performed with the local updates at the server, which is also influenced by the data amounts. Here, the local gradients of clients with the most data have the strongest influence on the global gradient update. To see the effect of this weighted aggregation on the client performances, and since FedAvg shows comparable results with the other FL algorithms, exemplary tests are performed with FedAvg with an unweighted aggregation in this work. As mentioned in the results, Figure A.2 shows no major difference in performance for both aggregation versions. Hence, the aggregation scheme is not the reason for dissimilar performances across client datasets for this task. Another potential reason for the differences is the distribution of labels amongst different clients. This is discussed in the following subsections.

7.1.4. Differences between Tools

As shown in the Figure 5.5, the dataset label distribution is highly skewed. The performance on the two most occurring tools is high for all learning algorithms, including the baselines, as the models have a high number of positive examples in the local training datasets from which to learn the specifications of the two tools. That leads to smaller improvement possibilities compared to the LOT with an ALTV, as the performance is already high for the baseline.

The results show that more improvements over ALTV can be made for the lesser occurring tools. On the other hand, with the EGOV, the VGG-13 achieves high F1-Scores for all tools. Hence, the clients can detect the presence of the lesser occurring tools in their own test dataset with high accuracy but cannot detect these tools from other test datasets. This suggests that, similar to the case of a CENT, the FL algorithms allow the clients to reach higher performances on the tools for which they do not have enough training data. However, this increase is mainly distributed to examples of the tools from other client datasets.

7.1.5. Combination of Clients and Tools

Regarding the differences in performance between different clients, discussed in subsection 7.1.3, HEI1, while having a slightly higher percentage of frames with a ”Grasper”, has much fewer examples of the ”Coagulation Instrument”. Instead, the dataset contains comparably more examples of the lesser occurring tools, which are, as mentioned, observed to have overall lower F1-Scores. While this provides an argument for the overall lower performance for HEI1, the high presence of the lesser occurring tools should, on average, lead to a larger improvement over the ALTV baseline, as discussed in the preceding subsection. However, on the contrary, the performance improvements are smaller for HEI1 than for other clients. This suggests that the specific tool categories are more challenging to detect in HEI1 than in CHO1-CHO3.

For HEI2, the test dataset contains no example of the ”Irrigation-Suction” tool. That negatively influences the local macro F1-Score on HEI2, compared to the other local scores. The macro score factors in the tool score of 0, whereas in reality, the model might be able to detect the tool with higher accuracy.

An alternative way to handle the missing tool in the data set is not to consider the tool score of the missing tool in the respective local macro score. As seen in Figure A.3, this positively influences the macro scores for HEI2. However, they remain lower than the scores of CHO1-CHO3, despite HEI2 having a similar label distribution for all other labels as CHO1-CHO3. For this reason, and as HEI1 and HEI2 seem to share a roughly similar image distribution, it is indicated that the tools in HEI2 are also harder to detect.

The abovementioned problem of missing tools is caused by the experimental setup and results from the random division into train and test datasets. However, this is a situation that is possible in a real-world federation, where clients do not have enough data for specific labels in order to have them appropriately represented in the train and test datasets. For both described solutions for handling the missing tools, a bias is introduced into the results.

7.1.6. ALTV vs. EGOV

As seen, for example, in Figure 6.3, there is a clear difference in the reported performance, depending on whether one approaches the results of the LOTs with the interpretation of the ALTV or the EGOV. This argument also holds in terms of statistical significance. To our knowledge, this difference in interpretation has not yet been discussed in the research of FL. Nevertheless, it is a vital topic, as it might heavily influence the motivation of clients to participate in a FLS.

For a client, it is crucial to decide which of the two interpretations is the more important one and shall be used when assessing the benefits of FL. Suppose a client is mainly interested in improving ML services at the own institution, for hospitals, given the own patient population and screening settings. In that case, the determining factor should be the difference between the results of the LOT with the EGOV and the results of FL. On the other hand, if the client is interested in building a model that performs robustly over different data distributions and does not only consider the own patient population and screening settings but instead emphasizes on equally treating all populations, including local minorities, then the ALTV should be the important interpretation. In this case, the reported performance improvement of FL algorithms over the LOTs is much higher, as the LOTs produce models with low generalization capabilities, as backed by the presented data. For the EGOV, the smaller improvement margin between the approaches leads to the requirement for more careful considerations.

In both cases, the corresponding results can then be used to weigh the benefits of the participation in a federation with the costs of building the infrastructure for such a system.

7.2. Training Statistics

From the results in section 6.2, it can be observed that the mean loss across clients behaves similarly to the CENT loss. Moreover, the length of the gradient difference between the global and local updates for FL algorithms shows a similar convergence behavior to the losses after an initial peak. That is similar to the findings in [29] and indicates that the length of the gradient difference might be a reasonable estimate for the convergence behavior of the local models, and hence the overall objective. This might be especially helpful in cases where, for some reason, the server has no access to the local losses, while he always has access to the gradient updates.

The statistics of the cosine similarities between the global and local update vectors yield a very intuitive interpretation. The similarity starts very high in the first epochs. That can be interpreted as all of the local models first adapting to the overall task, given a pretrained model. As the PyTorch models are pretrained on ImageNet [51], which contains images of many arbitrary objects, the FLTs start by finetuning the local models to adapt to the endoscope images with primarily red pixels. After the initial consensus and potentially reaching the coarse optimal region, the similarity between local updates drops. Subsequently, the similarity remains on an intermediate level, however, the overall performance keeps slightly rising. This indicates that for the further training progression, the clients are not approaching the same objective, but instead, each client tries to achieve their own local objective. Still, these objectives are similar enough to push the global model slightly towards the goal of the global objective.

When looking at Figure 6.6, it can be seen that the updates of HEI1 and CHO2 have the highest similarity with the global update, on the example of FedAvg. This is similar for FedProx, FedDG and also FedRep. For CHO2, this seems reasonable, as the client does provide the highest amount of data in the federation and hence has the highest impact on the global update when a weighted aggregation is used. Given this impact, the difference between its local update and the global update should be smaller. On the other hand, for HEI1, there is no obvious explanation for the highest similarity with the global update, even higher than the one of CHO2. Although HEI1 highly agrees with the global update, the test performance on HEI1 is the lowest due to the underlying data distribution. Thus, interestingly, the similarities of the local updates with the global update appear to be in no correlation with the performance of the corresponding client. Therefore, the per-client cosine similarities do not seem to provide any valid information when estimating the resulting distribution of performance benefits amongst the clients. This finding is also consistent with and explains the non-existence of a substantial difference between the weighted and unweighted aggregation for FedAvg, as reported in subsection 7.1.3. In the context of a weighted aggregation, reducing the influence of the local gradients of the clients originating from Cholec80 on the global update and giving more influence to the clients originating from HeiChole, i.e., performing an unweighted aggregation, should not impact the global optimization substantially, as the local update of HEI1, similar to CHO2, shares a high similarity with the global update.

7.3. Stability of Federated Learning

During the preliminary experimentation, multiple hyperparameter settings resulted in an unstable CENT, where the performance across all metrics dropped immediately close to zero after normally rising for some epochs. From the observations during our tests, we attribute this instability to the usage of positive weights for the loss function, as the removal of positive weights stabilized the trainings. The positive weights are highly different for each label, caused by the label imbalance in the training set. Since the percentage of positive labels is very low for some of the tools, the corresponding positive weights are very high. That might lead to exploding gradients when labels with high positive weight are flipped to a different output. In fact, for this reason, the "Stapler" tool was not considered in the training setting, as it was only present at one client and for that client only in one surgical video, which resulted in an extremely high positive weight for that tool and an unstable training when using this video and the tool in the training dataset. With the removal of this tool from the consideration, an adjustment to the learning rate could control the stability of the CENT. However, despite remaining stable overall, the performance of the CENT could not reach the performance of the FL algorithms for the VGG-13. We suspect that the large gradients caused by tools with high positive weights still disrupted the CENT process, which is supported by the bumps in the test curve of the CENT for the VGG-13 in Figure 6.3. Here, a further reduction of the learning rate led to an overall worse performance.

There were no comparable cases of instability for the FLTs and the LOTs. The trainings here remained stable, independent of the usage of positive weights. We assume that for the

FLTs, the global aggregation is able to smooth out potential outliers in the local gradients, keeping the training stable. Hence, this encourages further research on the stability of FL algorithms.

7.4. Influence of the Machine Learning Model

For most of the observed statistics, the relative results of the FL algorithms were quite similar for both considered ML models. However, the VGG-13 performed better overall, which most likely originates from the more complex model architecture, and should be no attribute of FL, as this performance increase is also seen for all baselines. When looking at Figure 6.3, one can see that only FedRep showed slightly better results for the VGG-13 than for the AlexNet, which might be influenced by the size of the representation and local head, as previously discussed.

Besides the FL algorithms, a LOT with an EGOV also showed, relatively speaking, better results for the VGG-13 than for the AlexNet. The contrary is true for the ALTV. This indicates that the VGG-13, with its higher complexity, overfits on the comparably small local datasets and therefore performs better on its own data but generalizes worse to the data of other clients. That needs to be taken into consideration when comparing such models with the two lower baselines. For models that exhibit more amounts of overfitting, the difference of FLTs to the LOTs with an EGOV will be smaller and the difference to the LOTs with an ALTV will be larger.

Though, in terms of the performances of the FL algorithms, the observed results give the intuition that FL does not depend substantially on the ML model and that it facilitates the use of an arbitrary model for this particular task. However, only two models were compared that also share their basic structure. More thorough investigations are required on many models with different architectures, e.g., ResNets, to determine a definite rule for the influence of the ML model on the FL training process.

7.5. Federated Learning in the Medical Context

The efficiency of FL in training ML models on medical data, in this case, surgical video data, is consolidated by the results of this work. FL in medicine provides the possibility to train highly-performant ML models in a domain, where it otherwise might be hard or impracticable to centrally collect enough data at a reasonable cost to achieve comparable results. It allows for the data residing at medical institutions to be used more publicly and to the benefit of more institutions and people, without violating privacy concerns and data regulations, given that the studies of FL privacy and security can assure these traits. This also comes at a time when medical ML models are becoming more and more important in the healthcare sector, and increasing amounts of data are collected and available at medical institutions.

As discussed in subsection 7.1.6, FL not only can provide performance benefits to the institutions but especially has the potential to significantly enhance the generalization capabilities of the ML models. In a medical context, this can alleviate the risks of biases present in the model, which can lead to serious health hazards when decisions are made based upon wrong model predictions. Otherwise, and although this should not be as significant of a problem for surgical videos, patients from a different demographic population than the one that is mainly represented at a hospital could experience real disadvantages and risks when such a bias was introduced into the model, due to a lack of comprehensive data. Moreover, such biases can not even be detected by observing the performance on an institution's own test dataset, as this is typically drawn from roughly the same distribution. While it also is not possible in FL to observe the performance on the test data of other clients, it is reasonable to assume that the global model trained in a federation can perform better across different data distributions, as seen in the results.

As also the authors in [6] state, FL furthermore removes the requirement of anonymizing the patient data and removes the requirement of duplicating a dataset at multiple medical institutions. FL in medicine gives the patients the ability to revoke the consent for the usage of their personal data that is only residing at that institution, which lowers the burden of becoming a data donor. Additionally, patients in remote areas can get access to the same high-quality ML-aided treatment as in hospitals in more populated regions. Finally, medical institutions can regulate more precisely what the data is used for.

Given this diverse range of potentials and benefits of FL, broad usage of FL in the medical domain is likely to be seen in the future.

8. Conclusion

The evaluations in this thesis have shown that FL provides a powerful tool for learning ML models on medical data that is distributed and stored privately at several institutions. All except one of the considered FL algorithms achieve performances that are comparable with classical centralized ML, where the data is required to be collected at a single center. Further, they yield a significant increase in performance compared to local trainings, where an institution trains the model only with its own data. This performance gain is specifically associated with the generalization capabilities of the FL global model on heterogeneous data. The benefits of FL in terms of performance on each clients' own test dataset are smaller, but the experiments also showed positive influences in all but one case. In order to investigate the two abovementioned types of possible client motivations in generalization and personal local benefit, thorough investigations were made for both ideas, and the differences between the two interpretation variants were discussed.

It was found that there were only slight differences in performance amongst the best performing FL algorithms, and FedAvg [10] which was proposed alongside the problem definition of FL yielded very competitive results with the least amount of computations required. With FedDyn, a FL algorithm was considered that did not perform well across all performance metrics, failing to match the performances of the lower baselines of LOTs.

Regarding differences in performance between different client datasets, FL showed no tendency in reducing the gap which is present for LOTs with an ALTV, instead even slightly favoring the already better performing client datasets. The differences in performances across all learning algorithms were coherent with the original datasets, HeiChole and Cholec80, showing the existence of a MIDB. Regarding surgical tools, FL yields larger improvements for the lesser occurring, or in this case equivalently, initially worse performing tools.

Different statistics such as the length of the gradient difference and the cosine similarity between the global and local updates were investigated for the FL training process. The length of the difference between the global and local updates appears to be a good measure of model convergence. The cosine similarity gave insight that there is no correlation between the consistency of a local gradient with the global gradient and the resulting local performance. Lastly, the FL algorithms showed only small differences between the two considered models, indicating that FL is not highly dependent on the ML model. All of the abovementioned findings accomplish the objectives of this thesis.

For future work, more thorough studies of the stability of FL are needed. Since, in this work, unstable trainings occurred with the CENT, where FL algorithms remained stable, the influence of different factors such as positive weights should be investigated.

Regarding FL algorithms, there exist many more different approaches that could be assessed with respect to their model performances, such as asynchronous or fully-decentralized FL algorithms. Additionally, the adoption and performance evaluation of FL on other medical prediction tasks is an interesting direction for further research.

Furthermore, more rigorous evaluations need to be made for different ML model architectures in order to prove the influence of these models on the FL training process.

List of Acronyms

ML Machine Learning

DL Deep Learning

ANN Artificial Neural Network

CNN Convolutional Neural Network

ReLU Rectified Linear Unit

IIL Institutional Incremental Learning

CIIL Cyclic Institutional Incremental Learning

FL Federated Learning

FLS Federated Learning System

PFL Personalized Federated Learning

MMFLS Multicentric Medical Federated Learning System

MIDB Medical Institution Data Bias

GDPR General Data Protection Regulation

IID Independent and Identically Distributed

non-IID non-Independent and Identically Distributed

EMD Earth Mover's Distance

FFT Fast Fourier Transform

TP True Positive

FN False Negative

FP False Positive

TN True Negative

BCE Binary Cross Entropy

SGD Stochastic Gradient Descent

WAVG Weighted Average

LOT Local Only Training

ALTV Altruistic View

EGOV Egocentric View

FLT Federated Learning Training

CENT Centralized Training

FedAvg Federated Averaging

FedProx Federated Proximal

FedDyn Federated Dynamic Regularizer

FedDG Federated Domain Generalization

FedRep Federated Representation Learning

APD Fed**A**vg Fed**P**rox Fed**D**G

List of Figures

9. Appendix

A. Results

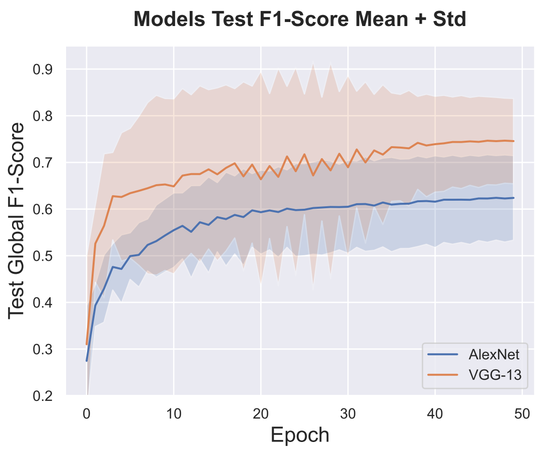


Figure A.1.: Comparison of the ML models. Mean and variance across all learning algorithms for both models.

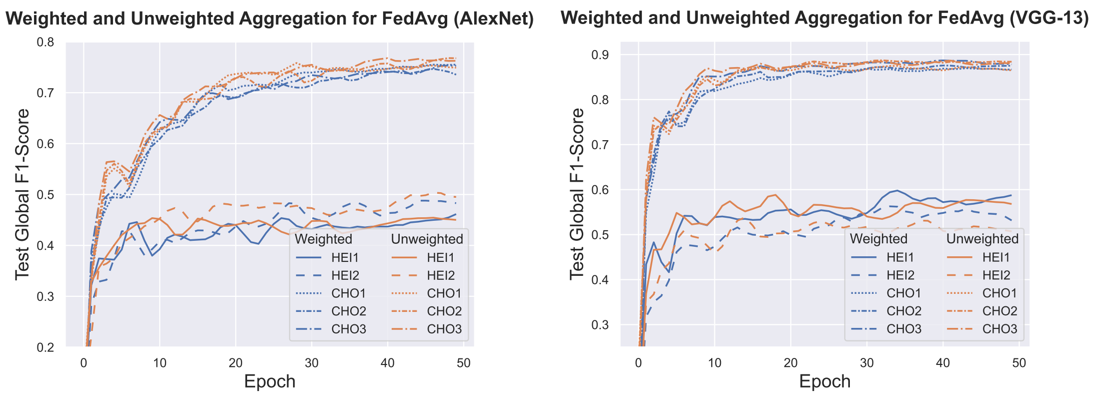


Figure A.2.: Comparison of macro F1-Scores on global test dataset for a weighted and an unweighted server aggregation.

AlexNet

		Test Dataset						
		HEI1	HEI2	CHO1	CHO2	CHO3	Mean	Var
Learning Algorithms	ALTV	0.364	0.394	0.525	0.604	0.539	0.485	0.008
	EGOV	0.359 (-0.005)	0.422 (+0.028)	0.602 (+0.077)	0.688 (+0.084)	0.653 (+0.114)	0.545 (+0.060)	0.01702
	FedAvg	0.461 (+0.098)	0.483 (+0.089)	0.755 (+0.230)	0.735 (+0.131)	0.753 (+0.214)	0.638 (+0.152)	0.01832
	FedProx	0.468 (+0.104)	0.479 (+0.085)	0.734 (+0.209)	0.730 (+0.126)	0.751 (+0.211)	0.632 (+0.147)	0.01686
	FedDyn	0.345 (-0.018)	0.360 (-0.034)	0.452 (-0.073)	0.456 (-0.148)	0.463 (-0.076)	0.415 (-0.070)	0.00264
	FedDG	0.487 (+0.123)	0.488 (+0.094)	0.742 (+0.217)	0.749 (+0.145)	0.749 (+0.21)	**0.643 (+0.158)**	0.01614
	FedRep	0.393 (+0.029)	0.473 (+0.078)	0.719 (+0.195)	0.723 (+0.119)	0.723 (+0.184)	0.606 (+0.121)	**0.02071**
	CENT	0.447 (+0.084)	0.483 (+0.089)	0.760 (+0.235)	0.771 (+0.167)	0.785 (+0.246)	0.649 (+0.164)	0.02277
	Mean	0.416 (+0.052)	0.448 (+0.054)	0.661 (+0.136)	**0.682** (+0.078)	0.677 **(+0.138)**		
	Var	0.00279	0.00212	**0.01244**	0.00954	0.01197		
	MeanFL	0.431 (+0.067)	0.456 (+0.062)	0.680 **(+0.155)**	0.679 (+0.075)	**0.688** (+0.149)		
	VarFL	0.00284	0.00236	**0.01318**	0.01244	0.01279		

VGG-13

		Test Dataset						
		HEI1	HEI2	CHO1	CHO2	CHO3	Mean	Var
Learning Algorithms	ALTV	0.459	0.391	0.654	0.744	0.679	0.585	0.01847
	EGOV	0.462 (+0.003)	0.455 (+0.064)	0.789 (+0.135)	0.869 (+0.125)	0.835 (+0.156)	0.682 (+0.097)	**0.03401**
	FedAvg	0.588 (+0.129)	0.532 (+0.141)	0.867 (+0.213)	0.875 (+0.132)	0.878 (+0.200)	**0.748 (+0.163)**	0.02398
	FedProx	0.573 (+0.114)	0.482 (+0.091)	0.851 (+0.197)	0.862 (+0.118)	0.882 (+0.203)	0.730 (+0.145)	0.02828
	FedDyn	0.480 (+0.020)	0.391 (+0.000)	0.569 (-0.085)	0.583 (-0.160)	0.543 (-0.136)	0.513 (-0.072)	0.00502
	FedDG	0.573 (+0.114)	0.514 (+0.123)	0.869 (+0.215)	0.878 (+0.135)	0.883 (+0.205)	0.743 (+0.158)	0.02706
	FedRep	0.551 (+0.092)	0.536 (+0.145)	0.874 (+0.219)	0.885 (+0.141)	0.895 (+0.216)	**0.748 (+0.163)**	0.02792
	CENT	0.547 (+0.088)	0.501 (+0.111)	0.834 (+0.180)	0.843 (+0.099)	0.877 (+0.199)	0.721 (+0.135)	0.02613
	Mean	0.529 (+0.079)	0.475 (+0.084)	0.788 **(+0.134)**	**0.817** (+0.074)	0.809 (+0.130)		
	Var	0.00250	0.00297	0.01151	0.00963	**0.01451**		
	MeanFL	0.553 (+0.094)	0.491 (+0.100)	0.806 **(+0.152)**	**0.817** (+0.073)	0.816 (+0.138)		
	VarFL	0.00148	0.00287	0.01410	0.01365	**0.01870**		

Table A.1.: F1-Scores for each learning algorithm and client. The difference in performance compared to ALTV on the same client is written in brackets below each value. Additionally, the mean and variance of the F1-Scores is computed and shown across clients (right) and across learning algorithms (bottom). The means and variances over the learning algorithms are computed with respect to all algorithms (Mean, Var) and only with respect to the FL algorithms (MeanFL, VarFL). The highest values for the means and variances are highlighted in bold font, where the CENT is excluded for the learning algorithms.

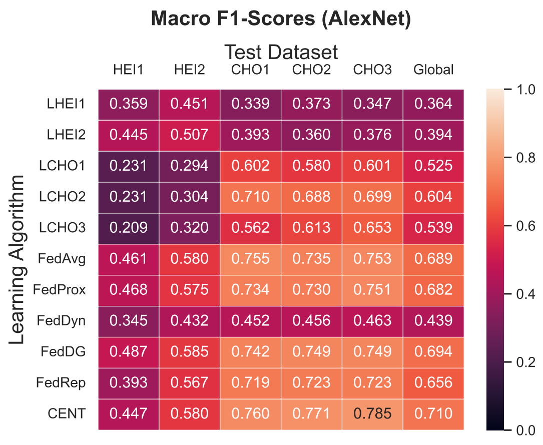
AlexNet

		Tool							
		Grasper	Clipper	Coagulation	Scissors	Irrigation	Specimen	Mean	Var
Learning Algorithms	ALTV	0.808	0.205	0.810	0.387	0.363	0.570	0.524	0.05192
	EGOV	0.858 (+0.050)	0.329 (+0.125)	0.875 (+0.065)	0.476 (+0.090)	0.387 (+0.023)	0.651 (+0.082)	0.596 (+0.072)	0.04654
	FedAvg	0.860 (+0.052)	0.466 (+0.262)	0.893 (+0.083)	0.626 (+0.239)	0.558 (+0.195)	0.732 (+0.162)	0.689 (+0.166)	0.02394
	FedProx	0.864 (+0.055)	0.447 (+0.243)	0.893 (+0.083)	0.579 (+0.192)	0.571 (+0.208)	0.741 (+0.171)	0.682 (+0.159)	0.02654
	FedDyn	0.781 (-0.027)	0.188 (-0.016)	0.644 (-0.166)	0.143 (-0.243)	0.324 (-0.039)	0.557 (-0.013)	0.439 (-0.084)	**0.05612**
	FedDG	0.848 (+0.040)	0.476 (+0.272)	0.902 (+0.092)	0.620 (+0.233)	0.580 (+0.217)	0.738 (+0.168)	**0.694 (+0.170)**	0.02247
	FedRep	0.850 (+0.041)	0.461 (+0.256)	0.866 (+0.056)	0.571 (+0.184)	0.479 (+0.116)	0.710 (+0.140)	0.656 (+0.132)	0.02688
	CENT	0.872 (+0.063)	0.478 (+0.273)	0.912 (+0.102)	0.667 (+0.281)	0.593 (+0.230)	0.739 (+0.169)	0.710 (+0.186)	0.02286
	Mean	0.842 (+0.034)	0.381 **(+0.177)**	**0.850** (+0.039)	0.509 (+0.122)	0.482 (+0.119)	0.680 (+0.110)		
	Var	0.00085	**0.01343**	0.00692	0.02620	0.01049	0.00528		
	MeanFL	**0.840** (+0.032)	0.408 **(+0.203)**	**0.840** (+0.030)	0.508 (+0.121)	0.502 (+0.139)	0.696 (+0.126)		
	VarFL	0.00091	0.01214	0.00975	**0.03369**	0.00926	0.00491		

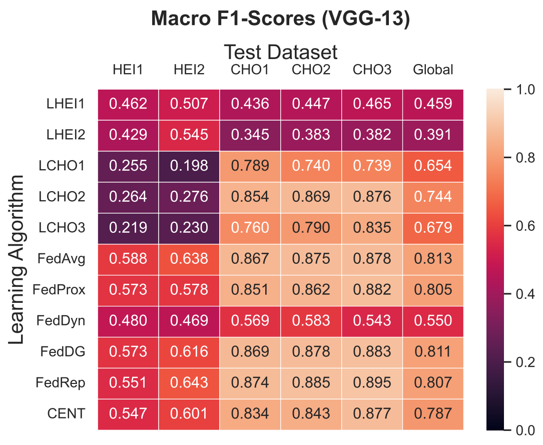
VGG-13

		Tool							
		Grasper	Clipper	Coagulation	Scissors	Irrigation	Specimen	Mean	Var
Learning Algorithms	ALTV	0.833	0.356	0.858	0.584	0.576	0.634	0.640	0.02877
	EGOV	0.884 (+0.050)	0.644 (+0.287)	0.924 (+0.066)	0.691 (+0.107)	0.622 (+0.045)	0.753 (+0.119)	0.753 (+0.112)	0.01322
	FedAvg	0.903 (+0.070)	0.647 (+0.291)	0.943 (+0.085)	0.789 (+0.205)	0.783 (+0.207)	0.810 (+0.176)	**0.813 (+0.172)**	0.00902
	FedProx	0.905 (+0.071)	0.603 (+0.247)	0.944 (+0.086)	0.791 (+0.207)	0.789 (+0.213)	0.800 (+0.165)	0.805 (+0.165)	0.01172
	FedDyn	0.850 (+0.017)	0.291 (-0.066)	0.728 (-0.130)	0.171 (-0.413)	0.563 (-0.013)	0.699 (+0.065)	0.550 (-0.090)	**0.05916**
	FedDG	0.895 (+0.061)	0.637 (+0.281)	0.947 (+0.089)	0.779 (+0.195)	0.791 (+0.215)	0.814 (+0.180)	0.811 (+0.170)	0.00951
	FedRep	0.900 (+0.067)	0.706 (+0.350)	0.933 (+0.076)	0.757 (+0.173)	0.711 (+0.135)	0.834 (+0.199)	0.807 (+0.167)	0.00786
	CENT	0.889 (+0.055)	0.572 (+0.216)	0.936 (+0.078)	0.768 (+0.185)	0.743 (+0.167)	0.813 (+0.178)	0.787 (+0.147)	0.01360
	Mean	0.882 (+0.049)	0.557 **(+0.201)**	**0.902** (+0.044)	0.666 (+0.082)	0.697 (+0.121)	0.770 (+0.135)		
	Var	0.00061	**0.01976**	0.00504	0.03930	0.00816	0.00425		
	MeanFL	0.891 (+0.057)	0.577 **(+0.221)**	**0.899** (+0.041)	0.657 (+0.074)	0.727 (+0.151)	0.791 (+0.157)		
	VarFL	0.00042	0.02160	0.00734	**0.05923**	0.00765	0.00224		

Table A.2.: F1-Scores for each learning algorithm and tool. The difference in performance compared to ALTV on the same tool is written in brackets below each value. Additionally, the mean and variance of the F1-Scores is computed and shown across tools (right) and across learning algorithms (bottom). The means and variances over the learning algorithms are computed with respect to all algorithms (Mean, Var) and only with respect to the FL algorithms (MeanFL, VarFL). The highest values for the means and variances are highlighted in bold font, where the CENT is excluded for the learning algorithms.



	HEI1	HEI2	CHO1	CHO2	CHO3	Global
LHEI1	0.359	0.451	0.339	0.373	0.347	0.364
LHEI2	0.445	0.507	0.393	0.360	0.376	0.394
LCHO1	0.231	0.294	0.602	0.580	0.601	0.525
LCHO2	0.231	0.304	0.710	0.688	0.699	0.604
LCHO3	0.209	0.320	0.562	0.613	0.653	0.539
FedAvg	0.461	0.580	0.755	0.735	0.753	0.689
FedProx	0.468	0.575	0.734	0.730	0.751	0.682
FedDyn	0.345	0.432	0.452	0.456	0.463	0.439
FedDG	0.487	0.585	0.742	0.749	0.749	0.694
FedRep	0.393	0.567	0.719	0.723	0.723	0.656
CENT	0.447	0.580	0.760	0.771	0.785	0.710



	HEI1	HEI2	CHO1	CHO2	CHO3	Global
LHEI1	0.462	0.507	0.436	0.447	0.465	0.459
LHEI2	0.429	0.545	0.345	0.383	0.382	0.391
LCHO1	0.255	0.198	0.789	0.740	0.739	0.654
LCHO2	0.264	0.276	0.854	0.869	0.876	0.744
LCHO3	0.219	0.230	0.760	0.790	0.835	0.679
FedAvg	0.588	0.638	0.867	0.875	0.878	0.813
FedProx	0.573	0.578	0.851	0.862	0.882	0.805
FedDyn	0.480	0.469	0.569	0.583	0.543	0.550
FedDG	0.573	0.616	0.869	0.878	0.883	0.811
FedRep	0.551	0.643	0.874	0.885	0.895	0.807
CENT	0.547	0.601	0.834	0.843	0.877	0.787

Figure A.3.: Final F1-Scores for single client test datasets, where the missing ”Irrigation-Suction” tool is not considered for HEI2.

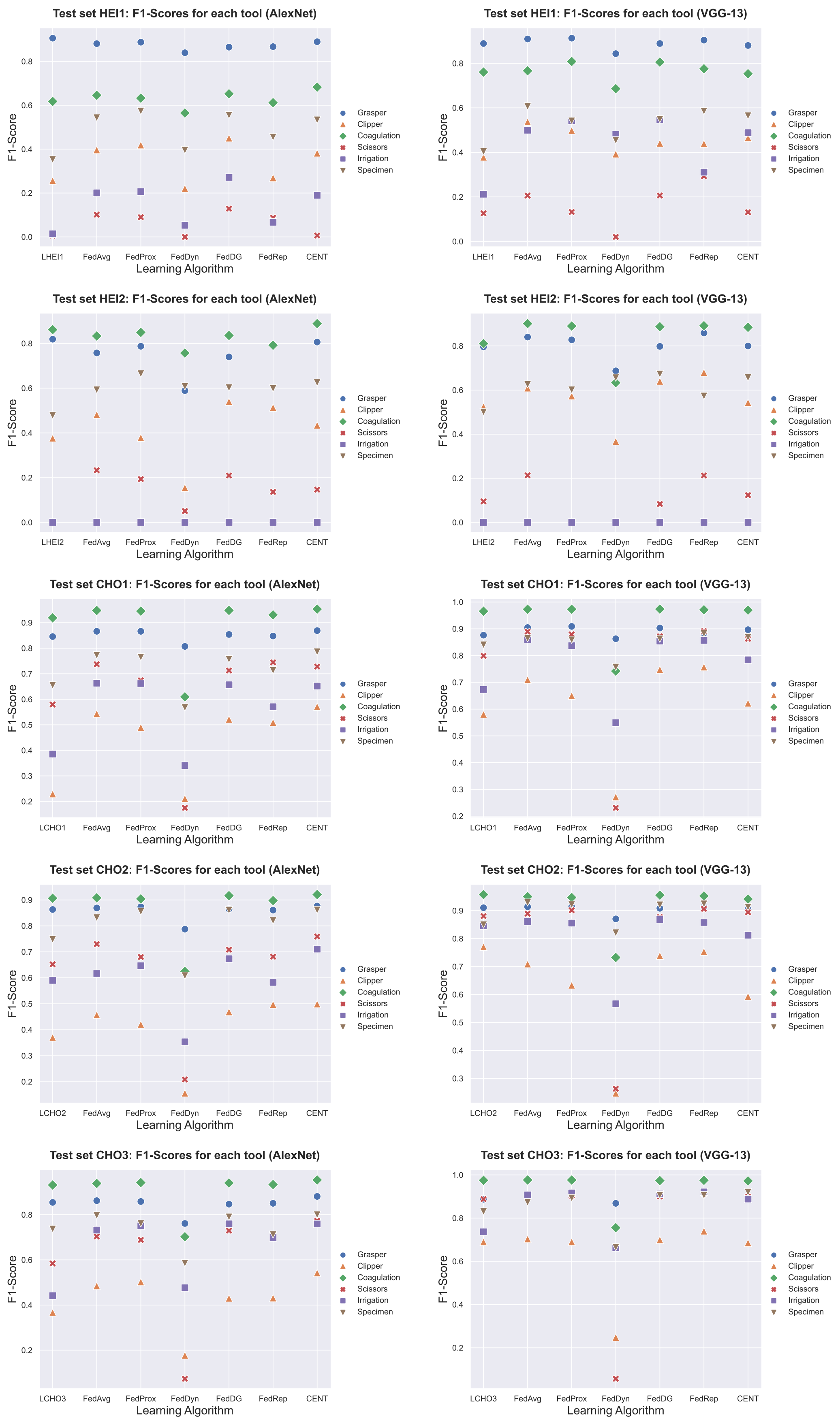


Figure A.4.: Tool-wise F1-Scores on each unique test dataset. Comparison of the respective client LOT, all FLTs and the CENT.

Bibliography

[1] A. Althnian, D. AlSaeed, H. Al-Baity, A. Samha, A. B. Dris, N. Alzakari, A. Abou Elwafa, and H. Kurdi, "Impact of dataset size on classification performance: An empirical evaluation in the medical domain," *Applied Sciences*, vol. 11, no. 2, p. 796, 2021.

[2] S. Bodenstedt, M. Wagner, B. P. Müller-Stich, J. Weitz, and S. Speidel, "Artificial intelligence-assisted surgery: Potential and challenges," *Visceral Medicine*, vol. 36, no. 6, pp. 450–455, 2020.

[3] W. G. Van Panhuis, P. Paul, C. Emerson, J. Grefenstette, R. Wilder, A. J. Herbst, D. Heymann, and D. S. Burke, "A systematic review of barriers to data sharing in public health," *BMC public health*, vol. 14, no. 1, pp. 1–9, 2014.

[4] L. O. Gostin, "National health information privacy: regulations under the health insurance portability and accountability act," *Jama*, vol. 285, no. 23, pp. 3015–3021, 2001.

[5] E. Commission. What data can we process and under which conditions? Accessed: 2021/11/23. [Online]. Available: https://ec.europa.eu/info/law/law-topic/data-protection/reform/rules-business-and-organisations/principles-gdpr/what-data-can-we-process-and-under-which-conditions_en

[6] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ digital medicine*, vol. 3, no. 1, pp. 1–7, 2020.

[7] L. Rocher, J. M. Hendrickx, and Y.-A. De Montjoye, "Estimating the success of re-identifications in incomplete datasets using generative models," *Nature communications*, vol. 10, no. 1, pp. 1–9, 2019.

[8] C. Wachinger, P. Golland, W. Kremen, B. Fischl, M. Reuter, A. D. N. Initiative *et al.*, "Brainprint: A discriminative characterization of brain morphology," *NeuroImage*, vol. 109, pp. 232–248, 2015.

[9] G. Mårtensson, D. Ferreira, T. Granberg, L. Cavallin, K. Oppedal, A. Padovani, I. Rektorova, L. Bonanni, M. Pardini, M. G. Kramberger *et al.*, "The reliability of a deep learning model in clinical out-of-distribution mri data: a multicohort study," *Medical Image Analysis*, vol. 66, p. 101714, 2020.

[10] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[11] I. Dayan, H. R. Roth, A. Zhong, A. Harouni, A. Gentili, A. Z. Abidin, A. Liu, A. B. Costa, B. J. Wood, C.-S. Tsai *et al.*, "Federated learning for predicting clinical outcomes in patients with covid-19," *Nature medicine*, vol. 27, no. 10, pp. 1735–1743, 2021.

[12] K. V. Sarma, S. Harmon, T. Sanford, H. R. Roth, Z. Xu, J. Tetreault, D. Xu, M. G. Flores, A. G. Raman, R. Kulkarni *et al.*, "Federated learning improves site performance in multicenter deep learning without data sharing," *Journal of the American Medical Informatics Association*, vol. 28, no. 6, pp. 1259–1264, 2021.

[13] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen *et al.*, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific reports*, vol. 10, no. 1, pp. 1–12, 2020.

[14] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*. PMLR, 2013, pp. 1139–1147.

[15] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[16] Image classification on imagenet. Accessed: 2022/03/20. [Online]. Available: https://paperswithcode.com/sota/image-classification-on-imagenet

[17] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas, "Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation," in *International MICCAI Brainlesion Workshop*. Springer, 2018, pp. 92–104.

[18] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[19] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer federated learning on graphs," *arXiv preprint arXiv:1901.11173*, 2019.

[20] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[21] P. M. Mammen, "Federated learning: Opportunities and challenges," *arXiv preprint arXiv:2101.05428*, 2021.

[22] B. Pfitzner, N. Steckhan, and B. Arnrich, "Federated learning in a medical context: A systematic literature review," *ACM Transactions on Internet Technology (TOIT)*, vol. 21, no. 2, pp. 1–31, 2021.

[23] A. M. McCarthy, B. M. Keller, L. M. Pantalone, M.-K. Hsieh, M. Synnestvedt, E. F. Conant, K. Armstrong, and D. Kontos, "Racial differences in quantitative measures of area and volumetric breast density," *JNCI: Journal of the National Cancer Institute*, vol. 108, no. 10, 2016.

[24] J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, J. J. Titano, and E. K. Oermann, "Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study," *PLoS medicine*, vol. 15, no. 11, p. e1002683, 2018.

[25] H. Yuan, M. Zaheer, and S. Reddi, "Federated composite optimization," in *International Conference on Machine Learning.* PMLR, 2021, pp. 12 253–12 266.

[26] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu *et al.*, "Fedml: A research library and benchmark for federated machine learning," *arXiv preprint arXiv:2007.13518*, 2020.

[27] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[28] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[29] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *arXiv preprint arXiv:1812.06127*, 2018.

[30] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *arXiv preprint arXiv:2002.10619*, 2020.

[31] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *arXiv preprint arXiv:2103.00710*, 2021.

[32] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," *arXiv preprint arXiv:2002.07948*, 2020.

[33] E. Bazan, P. Dokládal, and E. Dokladalova, "Quantitative analysis of similarity measures of distributions," in *British Machine Vision Conference 2019, BMVC 2019*, 2019.

[34] M. S. Jere, T. Farnan, and F. Koushanfar, "A taxonomy of attacks on federated learning," *IEEE Security & Privacy*, vol. 19, no. 2, pp. 20–28, 2020.

[35] O. Zari, C. Xu, and G. Neglia, "Efficient passive membership inference attack in federated learning," *arXiv preprint arXiv:2111.00430*, 2021.

[36] E. Diao, J. Ding, and V. Tarokh, "Heterofl: Computation and communication efficient federated learning for heterogeneous clients," *arXiv preprint arXiv:2010.01264*, 2020.

[37] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[38] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. M. Esmaeili, and D. Huba, "Federated learning with buffered asynchronous aggregation," *arXiv preprint arXiv:2106.06639*, 2021.

[39] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[40] Tensorflow federated: Machine learning on decentralized data. Accessed: 2021/11/25. [Online]. Available: https://www.tensorflow.org/federated

[41] P. Sharma, F. E. Shamout, and D. A. Clifton, "Preserving patient privacy while training a predictive model of in-hospital mortality," *arXiv preprint arXiv:1912.00354*, 2019.

[42] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," *arXiv preprint arXiv:2003.00295*, 2020.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[45] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization," *arXiv preprint arXiv:2111.04263*, 2021.

[46] Q. Liu, C. Chen, J. Qin, Q. Dou, and P.-A. Heng, "Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1013–1023.

[47] H. J. Nussbaumer, "The fast fourier transform," in *Fast Fourier Transform and Convolution Algorithms*. Springer, 1981, pp. 80–111.

[48] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 2089–2099.

[49] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[50] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[51] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[52] Large scale visual recognition challenge 2012 (ilsvrc2012). Accessed: 2022/03/20. [Online]. Available: https://image-net.org/challenges/LSVRC/2012/results.html

[53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[54] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[55] M. Wagner, B.-P. Müller-Stich, A. Kisilenko, D. Tran, P. Heger, L. Mündermann, D. M. Lubotsky, B. Müller, T. Davitashvili, M. Capek *et al.*, "Comparative validation of machine learning algorithms for surgical workflow and skill analysis with the heichole benchmark," *arXiv preprint arXiv:2109.14956*, 2021.

[56] A. P. Twinanda, S. Shehata, D. Mutter, J. Marescaux, M. De Mathelin, and N. Padoy, "Endonet: a deep architecture for recognition tasks on laparoscopic videos," *IEEE transactions on medical imaging*, vol. 36, no. 1, pp. 86–97, 2016.

[57] Sub-challenge: Surgical workflow and skill analysis. Accessed: 2022/01/05. [Online]. Available: https://endovissub-workflowandskill.grand-challenge.org/

[58] Endoscopic vision challenge. Accessed: 2022/01/05. [Online]. Available: https://endovis.grand-challenge.org/endoscopic_vision_challenge/

[59] I. Funke, A. Jenke, S. T. Mees, J. Weitz, S. Speidel, and S. Bodenstedt, "Temporal coherence-based self-supervised learning for laparoscopic workflow analysis," in *OR 2.0 context-aware operating theaters, computer assisted robotic endoscopy, clinical image-based procedures, and skin image analysis*. Springer, 2018, pp. 85–93.

[60] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[61] M. Andreux, J. O. d. Terrail, C. Beguier, and E. W. Tramel, "Siloed federated learning for multi-centric histopathology datasets," in *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning.* Springer, 2020, pp. 129–139.

[62] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[63] Bcewithlogitsloss. Accessed: 2022/03/20. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html

[64] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in *Artificial intelligence and machine learning for multi-domain operations applications*, vol. 11006. International Society for Optics and Photonics, 2019, p. 1100612.

[65] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, pp. 50–60, 1947.

[66] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.