Erik Haram Nygård

Federated Learning: Client-side Personalization

Master's thesis in Applied Computer Science
Supervisor: Mariusz Nowostawski
June 2022

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Erik Haram Nygård

Federated Learning: Client-side Personalization

Master's thesis in Applied Computer Science
Supervisor: Mariusz Nowostawski
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Federated Learning: Client-side personalization

Erik Haram Nygård

01/06/2022

Abstract

Federated learning is a machine learning paradigm that focuses on maintaining privacy and security across several clients. It works by employing algorithms on a central server that focuses on aggregating the resulting machine learning models from several clients. The resulting global model will perform worse on local datasets where the data distribution is unbalanced, meaning the model is undesirable for certain use-cases where a high degree of personalization is important. In this thesis the current state of personalization techniques in federated learning will be explored. Furthermore, two approaches that aims to personalize the global model to the local environment will be proposed. The first approach is inspired by federated averaging and aims to combine the local and global model. The second approach is another multi-model approach inspired by ensemble learning. This approach rather than combining two models it only combines the outputs and aims to differentiate the predictions that are measured to be more accurate and combine these. Both of these methods are then tested on the MNIST and CIFAR-10 datasets - the first method shows no increase in performance in comparison to the local model and the second approach shows a slight increase in performance. Ways to improve these approaches are then suggested to potentially show an even bigger increase in performance.

Sammendrag

Federated learning er en maskinlæring paradigme som fokuserer på å opprettholde personvern og sikkerhet utover flere klienter. Den benytter seg av algoritmer på en sentral server som har i oppgåve å samle sammen flere maskinlæringsmodeller fra ulike klienter i en enkel modell. Den resulterende globale modellen vil prestere dårligere på lokale datasett hvor fordelingen av data er ubalansert, noe som vil si at modellen vil være uønsket for enkelte bruksområder hvor en stor grad av personlig tilnærming er viktig. I denne avhandlingen vil de ulike tilnærmingsteknikkene i federated learning bli utforsket. Videre, to fremgangsmåter som skal tilnærme den globale modellen til det lokale miljøet vil bli foreslått. Den første fremgangsmåten er inspirert av federated averaging og har i hensikt å kombinere den lokale- og den globale modellen. Den andre fremgangsmåten er enda en multi-modell fremgangsmåte inspirert av ensemble learning. Denne fremgangsmåten vil kombinere prediksjonene fra begge modellene og vil differensiere hvilken modell er mest nøyaktig for hver enkel prediksjon for å så kombinere de. Begge disse metodene vil så bli testet på datasettene MNIST og CIFAR10 - den første metoden viser ingen forbedring i ytelse, men den andre vil vise en liten økning i ytelse. Diverse metode for å forbedre disse fremgangsmåtene vil deretter bli foreslått for å potensielt vise en enda større økning i ytelse.

Contents

Figures

Tables

Chapter 1

Introduction

In the current age of technology, more and more devices that generate data are being produced and used by consumers. This is happening in all kinds of devices, in devices used by the health sector, by car manufacturers, fitness and more. And with all of this new data being generated, of course laws and regulations will follow to better protect the individual against malicious actors. This increases the burden of responsibility on the producer, as well as limits what can be achieved with this newly generated data as there could be several actors within the same area that could combine their data and generate an overall better result, this could be due to each actor having limited samples, or a bigger dataset overall would lead to a better result. However, due to privacy laws or desires, this is not an option for many. Having more samples generally leads to a better result when training machine learning models, and with machine learning being a relatively old computer science technology, it's starting to show some of its age when applying it to more modern problems and contexts.

The current state of centralized machine learning (ML) is starting to show its negatives in recent years as laws and regulations affecting privacy are starting to get stricter in certain industries. Not only that, but the technology is getting more and more attention and is getting applied to several more use-cases. With factors such as these, some of the weaknesses of centralized machine learning is getting more noticeable. Especially regarding gathering the necessary data to be able to train these machine learning models. Some want to gain the performance increase in gathering as much data as possible, which is simply not a possibility with strict privacy regulations, and some do not want to share their data due to privacy concerns, either regarding industry secrets or customer privacy. These factors are significant concerns when building datasets for practical use in real world applications - they can also be considered obstacles, if the only objective is progress. However, they are important measures to protect the owner of the data, which is the individual most of the time. Datasets used in centralized machine learning are inherently monolithic – meaning all of the data in the dataset is concentrated at a single location. This kind of structure is problematic with the earlier mentioned factors in mind and will cause resulting datasets to be of lower quality due to not being able to make use of all the relevant data. This also causes conflict when two or more parties wishes to collaborate, and they will be unable to share data one way or the other.

Federated learning (FL) is a relatively new way to do machine learning. The main feature of FL is that it is decentralized, as opposed to traditional ML, where it is centralized. Essentially, with FL being decentralized – it means that there are several parties involved that each perform centralized ML, but the resulting model will be combined with the models from the other parties with a specific method of combining them. Without going into details, in most cases, this will result in an average between the models. This leads to a collaborative model between these parties, and each model have no knowledge of any of the data used in the other models, meaning that it is a method to perform ML with a fragmented dataset, without abiding to the monolithic structure of centralized ML datasets.

So far, FL is still not performing as well as centralized ML on the same dataset, however, it is slightly more complicated to get a precise metric for a one-to-one comparison between FL and ML – due to several factors. The one factor easily forgotten, is the fact that FL and centralized ML will simply not be trained on the same datasets, due to FL not being limited to the same datasets that centralized ML is, this factor alone will make a direct comparison in a practical context very difficult. Another important factor is the data distribution and data quality present in both datasets. In centralized ML there is really no worry between the distribution and data, is it is likely to remain constant for all samples, as it likely comes from the same source and therefore any unbalance in the data will be intentional. However, for FL this is not the case – in FL there will be more than one dataset, and there is no guarantee whether the distribution of samples in each client is similar or whether the quality of the data is similar, giving models with wildly different performances, or very similar performances. This is an issue of i.i.d. – which stands for “independent and identically distributed” which will be discussed later in this thesis. FL will open new opportunities where privacy is a concern. These opportunities will come in the shape of better performing models and collaboration between parties that will want to collaborate but can’t either regarding consumer privacy or industry secrets. One such example is health care, where the personal information about each patient is considered confidential and can’t legally be shared. However, with FL there is no need to share the actual data, and only the results. This opens for several health care institutions to create better ML models that can recognize patterns of illnesses or conditions more precisely and allows the health care professionals to provide more efficient and better care to their patients. However, there is no guarantee that these models perform better than the local ones, this is due to the previously mentioned issue of i.i.d. – as it will affect the performance of each model to an extent.

Each local model will be specialized in solving the pattern of problems that exists within the local environment it is trained within, which can be both a good thing and a bad thing, depending on the use case of said model. However, this also means that while the model generated by FL will perform better on average, it is likely to be outperformed by the local model in the local environment – meaning that it is not necessarily desired to replace the local model with it. The reasoning for considering this undesirable is because the data is a reflection of the environment that generates the data, meaning that discrepancies between data generated by two unique environments is because these environments are different, and therefore have a different distribution of problems and might require a different solution than the best-found average between all of the environments.

With this in mind, the model generated by the local environment can be considered a specialist of its environment, and replacing it with a generalized model, excluding it from the process of making predictions in the local environment makes little sense. In this thesis, what exactly will be explored is the process of better employing the strengths of the local model, in conjunction with the FL model to generate better performing predictions in the local environments. While these models are not the same, and will have a difference in performance in different cases, the goal is to add the strengths of the specialist model to the global model – as the difference in data distribution between the environments will create discrepancies between how well each model performs on a specific class in the dataset. The technique to achieve that in this thesis will be introduced later in a later chapter.

Further in this thesis the following sections will be outlined: Following this section will be a subsection introducing the research questions relevant for this thesis. Next a section briefly introducing the required background knowledge, followed by literature found relevant, as well as the results generated by a literature review performed to summarize the literature found in relation to the thesis topic and research questions, next is a section containing the methodology used for performing the research, as well as the methods relevant to perform the experiments to test it out, followed by presenting the results from performing the experiments and then the discussion and the conclusion.

1.1 Research questions

To aid in better exploring this topic a set of research questions were developed and proposed. The goal of these questions is to pinpoint what kind of research is needed to be done in order to gain a better understanding of how to achieve client-side aggregation, or personalizing the global model to the extent it consistently outperforms the local model in its environment.

- What are the effects of non-i.d.d. data in federated learning?
- How are the effects of non-i.d.d. data alleviated in federated learning?
- What are the methods for personalization in federated learning?
- Why or why not is personalizing important?

Chapter 2

Background

This chapter will briefly introduce the concepts used in this thesis. It will not explain each and every concept used in this thesis as some of it is assumed to be known beforehand.

2.1 Heterogeneous data

Heterogeneity in data is when irregularity occurs in a collection of data. This irregularity is dependent on the use case of the data and could manifest as anything such as the format, other specific properties of the meta-data to the occurrence of certain contexts within the data. One of the more challenging occurrences of heterogeneity in regards to federated learning is when it is non-i.i.d. (i.e. non-independent and -identically distributed data).

2.1.1 Independent and identically distributed data

The most common way for non-i.i.d. to occur in federated learning is when there is an unbalanced distribution of classes in the dataset. For example if there is a dataset with 4 classes and instead of the classes having a similar amount of examples, 50% of the examples belong to a single class, 40% to another and the remaining 10% is divided between two classes, this would result in an unbalanced distribution and hugely affect the performance of the machine learning model training on that dataset. This effect would then snowball and further affect the federated learning process. The other half of i.i.d. is that each occurrence of a data point is independent of other events. An example of this would be a collection of coin tosses, as the probability of getting heads or tails is unaffected by the last coin toss.

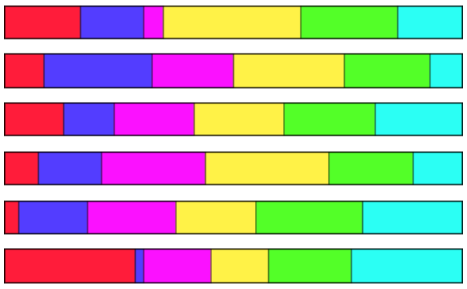


Figure 2.1: An example of how a non-i.i.d. data distribution would look in a FL network

2.2 Machine learning

This will not be a comprehensive overview of machine learning as that would be outside the scope, rather it will be a brief overview of the machine learning techniques which are relevant to this thesis.

2.2.1 Transfer learning

Transfer learning is a deep learning technique used to train machine learning models for various tasks. It focuses on using pre-trained models for one domain and transfer the accumulated knowledge in that model to another domain through fine-tuning the model. The most common approach to transfer learning is by using generalized models trained on huge datasets such as ImageNet and further train them on smaller and more specialized datasets. This works due to the early layers of model architectures such as convolutional neural networks (CNN) being general and only memorizing shapes such as curves and edges, only requiring the later layers of a CNN being re-trained, making them highly reusable with little resources investment. Transfer learning is a method that is less resource intensive than traditional deep learning due to requiring less powerful hardware and smaller datasets. However one of the drawbacks of transfer learning is catastrophic forgetting, meaning that the model will perform worse for every epoch for anything that is not present in the new dataset. A more comprehensive overview of transfer learning can be found from Zhang et. al. [1]

2.2.2 Pruning

Machine learning models are getting bigger and bigger as it allows for models to perform better. However this means that the models use more resources. Pruning can be used as a way to reduce storage space required, as well as reduce inference time of the model. The way pruning is performed is by removing unused weights from the model, effectively reducing the size of the model, this is done by setting the unused nodes to zero. A more complete look on pruning can be found from Blalock et. al. [2]

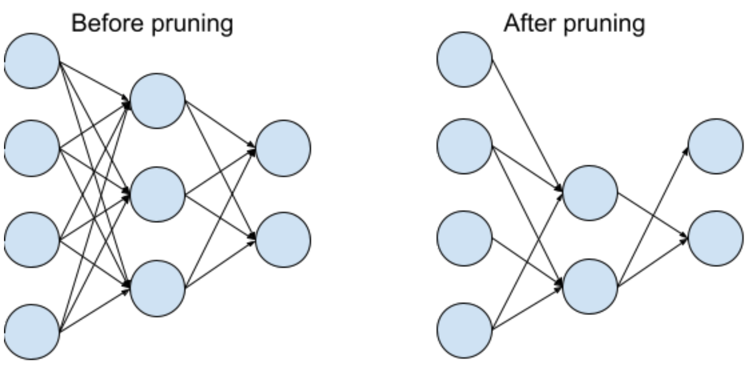


Figure 2.2: Visualisation of before and after pruning a network

2.2.3 Ensemble learning

Ensemble learning is a collection of machine learning techniques that focus on using multiple models. It uses the predictions from multiple models to generate a better prediction. One of the main use-cases of ensemble learning is that it combines the capabilities of multiple diverse models, giving the illusion that it is a single generalized model. Ensemble learning is in some ways similar to FL, in that it uses multiple models to generate an output, however the difference becomes apparent in how these different models become generated and how they are combined. The main ways of combining the outputs in ensemble learning is through methods such as bagging, boosting and stacking. Going into detail for each method would be outside the scope for this thesis, however the interested reader can read more about it from Sagi et. al. [3]

2.2.4 Meta-learning

Meta-learning is a another umbrella term for a set of machine learning techniques. The main idea behind these techniques embodies the idea of learning how to learn in a more general sense. The point of these techniques is to encapsulate the properties humans innately posses by transferring past knowledge to new concepts. The way this is done in ML is to teach the model to perform a set of specific tasks and then take the skills it learned from doing those tasks to perform another task [4].

2.3 Federated learning

Federated learning is a new and emerging branch of machine learning. The main focus of FL is its ability to perform machine learning with decentralized data, eliminating the need for a single huge centralized dataset, thus enhancing privacy [5]. However this perk does not come without its drawbacks, as it introduces other concerns and factors that will affect the performance of the ML model.

2.3.1 Client

Clients are edge devices that participates in training a shared global model and are typically numerous. For each client there will be a machine learning model that is local to that client, this model is trained on a unique dataset that is stored on that client. The data on the clients will be generated naturally by the user and most data that is generated naturally will be mostly non-i.i.d. due to how a user interacts with the device. If the device is a smartphone it can be due to the users interests and habits. If the client is a hospital it could be due to the environment and nature of injuries or diseases in the area, resulting in an unbalanced dataset. Point being that the chances for a perfectly independent and identically distributed dataset to naturally occur is low and therefore the clients can't be relied upon to provide it. Non-i.i.d. datasets perform worse in machine learning, and this is an issue in particular for federated learning due to how it gathers data, Zhao et. al. [6] shows that non-i.i.d. data will reduce the accuracy by up to 55%.

2.3.2 Server

After the client has trained its own model, that model will then be used to update the shared global model. The global model is generated on a central server that acts as a organizational body for the clients and will initiate training rounds for a selected set of clients and, the clients will then continue to train their models on their local dataset and send the result to the server, which will use the result to generate a new global model using a defined aggregation algorithm and then update the clients with it. The client selection is the step where the central server will select which clients to participate in the aggregation process and this can be done at random, by defining a scoring metric for the clients or other means such as reinforcement learning. Furthermore it can be used as a mechanism to counteract the non-i.i.d. nature of the clients [7, 8].

2.3.3 Aggregation

An aggregation algorithm defines how the various models will be combined to create the best possible general model for the entire network, which does not guarantee that the model will perform better than a local model, due to variances in each environment. It should however perform better on average than local models across all of the clients, provided it's a properly implemented aggregation algorithm such as federated averaging [9].

2.3.4 Scheme

Federated learning has an overarching scheme which sets it apart from other forms of ML. Usually data is centralized, however with FL the data is decentralized, this means that the data used for training the machine learning models is spread out over several clients.

In federated learning none of the data used for training is exchanged between clients and the server, allowing for an increase in privacy and security. Having the clients being responsible for their own data eliminates any possible conflict of rights regarding the data, it also allows for industries which have typically very strict data sharing policies, such as the health industry to collaborate with other entities within the industry in creating better ML models.

Federated learning introduces new challenges to tackle, such as the communication cost of transmitting data between central server and clients. Making developing aggregation algorithms that minimize communication important. Communication itself introduces a new vulnerability as it is possible to extract data from the updates provided by clients if not handled correctly [5, 10]. As mentioned earlier in the section, non-i.i.d. data is an issue for FL which steps have to be taken to counteract. Furthermore the general approach to non-i.i.d. data in FL is that it is undesirable when that is not always the case. As mentioned earlier, the data is a product of the environment it was generated in, making it highly relevant for personalizing the model for individual clients.

Chapter 3

Related work

The focus of this thesis will be on the client-side part of federated learning, rather than server-side. The goal will be to uncover if there is a way to implement a client-side aggregation method that will aid in further personalizing the global model after it arrives on the client. Personalization referring to the process of optimizing a global model to a client's local environment. In this section found research related to achieving that goal will be presented.

As mentioned in 2.3 the non-i.i.d. nature of some of the clients data serve to make some local models more accurate due to the data being a reflection of its environment, effectively removing the incentive for that client to participate in the FL process. Kulkarni et. al. surveys exactly this issue and concludes suggests that some techniques could be incorporated into the FL scheme to generate more specialized local models, such as: transfer learning and employing a mixture of models on the client [11]. The answer to this issue is not to simply retain unbalanced data and expect the global model to be easily malleable to each client as Zhao et. al. has shown it would reduce the accuracy of the model by up to 55%, furthermore shows that it can be mitigated by having a shared dataset across all of the devices, however this would conflict with some of the ideals of FL, as it would generate the need for a central dataset [6]. It would not address the issue of personalizing either, as it would have to be general enough to fit most of the possible environments of the clients.

There has also been research done to server-side aggregation to mitigate the issue of non-specialized models. Kopparapu et. al. developed an aggregation method that aims to group clients with similar data [12], this is achieved by a principle they introduce as cloning-and-deletion. Xiao et. al. noticed that traditional aggregation methods such as federated averaging behave in a volatile manner when computing on non-i.i.d. data due to the contribution difference, in terms of quality and amount of data, from clients and proposes an accuracy based averaging approach [13]. Another approach proposed by Briggs. et. al. aims to create clusters of similar clients to produce a global joint model for the defined clusters. These clients will then work in tandem to update their own joint model [14].

Most of the research done regarding this topic is focused on solving it from the server-side, as from the aforementioned research it aims to manipulate the client selection in some way or develop an aggregation method that can achieve this. However one could also approach this topic from the client-side, rather than doing the majority of the computation regarding personalization of models on the server-side, it could be moved to the client-side. Transfer learning, as suggested by Kulkarni et.al. [11] could achieve this, but it highlights the issue of catastrophic forgetting. The majority of the available research approaches this through grouping similar clients, instead of generating one general model for the entire network, it rather creates several sub-networks with different specializations and personalizes the global model that way [15, 16].

3.1 Literature review

A literature review was performed for this thesis. The reason for this review was to explore the theory and methods used for handling non-i.d.d. data and personalizing the global model to the local environments of the clients. In this section the findings will be presented. Table 3.1 summarizes the findings. It was discovered that there were a couple of broader categories that encapsulates the essence of what each paper tried to accomplish with their methods. The method that is most comparable to this thesis is the category identified as ensemble learning. As stated in subsection 2.2.3, this approach relies on utilizing more than one model to perform a single prediction, which is in-line with the approach of this thesis, which will be elaborated upon in a later chapter. However, what is not mentioned in the table, is that some of the papers aim to analyse the effect of data heterogeneity on federated learning and not necessarily propose a method that aims to solve it. Further in this section, each identified method in the table will be elaborated upon further in terms of what was uncovered in the literature review - as well as some topics that went outside the boundaries of the identified methods.

3.1.1 Data heterogeneity

Yang et.al. [17] performs a study on heterogeneity in FL and discovers that it degrades overall performance by up to 9.2% and can lengthen training time by a factor of 2.32 and can impact the fairness of the chosen aggregation method, in the sense that it impacts the client selection and the associated weights for each selected client. Some did propose ways to solve data heterogeneity between clients, and the most common approach to this problem was to establish some kind of common ground between the clients. Wu et.al. [18] proposes a framework that would aid in performing FL with clients that contain heterogeneous data. The approach is to partition each client in a manner that lets it process each partitioned component independently, making this a client-specific process. As opposed to traditional FL which is indiscriminate with how it performs aggregation and updates. Xue et.al. [19] achieved this through first establishing a difference between personalized and shared modules. The shared modules would have their

Method	Explanation
Pruning	This is described briefly in subsection 2.2.2, however it is applied slightly differently in federated learning. In federated learning pruning is used for reducing communication overhead, as well as personalize the models. This is done by averaging the intersection of the pruned models.
Ensemble learning	Briefly described in subsection 2.2.3. The concepts of EL can be applied to FL, however the execution is different. The models used are trained on entirely different datasets on different devices, causing more heterogeneity than just the data itself. The impact of this is difficult to measure, regardless it is still applicable and manages to achieve results to more traditional FL approaches.
Local optimization	Tuning of layers within the network appears to be the most researched approach to personalization from the results of this review. Transfer learning which is described in subsection 2.2.1 is similar to this approach, however the execution is slightly different, but the concept remains the same. Instead of using pre-trained models, the global model acts as the pre-trained model in this case. There is also the possibility of using a pre-trained model as the initial model for the FL network.
Client clustering	Also a very popular approach to this problem. Similar in concept to the ML technique of clustering, however instead of clustering similar data points together, it focuses on clustering similar clients together. These client clusters will then generate their own sub-network within the FL network and the FL network will then consist of several global models with their own specializations.
Meta-learning	The main idea behind these techniques is described in subsection 2.2.4. This is a common approach when applying federated learning to recommender systems. The meta-learning algorithm REPTILE is also commonly used with FL.

Table 3.1: Table of the uncovered methods

own parameters which where shared with the clients, however the clients would also retain parameters which were specific for each client.

3.1.2 Ensemble learning

There are several proposed methods which take root in principles of ensemble learning to address the proposed problem. Guo et. al. [20] proposes a framework based on the mixture of experts technique from ensemble learning, which aims to mix the outputs from both the local model and the global model, combining their efforts. Thonglek et.al. [21] proposes a method which addresses the limitation that several devices might be constrained by hardware. They proposed a method which combines weighted average with ensemble techniques in order to combine the output from each model. Li et.al. [22] has an interesting approach, which is the use of an ensemble technique known as temporal ensembling, essentially it is the method of combining the predictions of all past models.

3.1.3 Transfer learning

The most employed technique to personalize model output in this literature review was definitely transfer learning. The most common approach was to take the generated global model and then tune it using the local data on each client. This can be seen in the paper by Chen et.al. [23], where the goal is to personalize the global model to fit a patients unique profile. Furthermore, Khoa et.al. [24] in combination with the fine-tuning approach, it includes an oversampling method for the data with the goal of balancing out the data in a heterogeneous distribution. Guo et.al. [25] shows various strategies in fine-tuning the global model to a local environment in the context of hand gesture recognition, they made the discovery that fine-tuning the earlier layers of the model achieved a higher degree of improvement than the later layers.

3.1.4 Meta-learning

Some proposed approaches relied on techniques from meta-learning. Jalalirad et.al. [26] proposes a simplified approach that employs meta-learning techniques in order to train a recommender system, their implementation is an extension of the meta-learning algorithm REPTILE [4]. Wang et.al. [27] proposes another FL implementation of REPTILE for a fast-adapting FL-based recommender system. Balakrishnan et.al. [28] proposes another meta-learning based system based on task-similarity that aims to improve model personalization in FL as well as improving generalization across non-i.d.d. data. Xiong et.al. [29] proposes a method that aims to personalize the global model through the use of model agnostic meta-learning (MAML) [30].

3.1.5 Pruning

Another popular approach to personalization is pruning, which is elaborated upon in subsection 2.2.2. Liu et. al. [31] proposes a framework called "FedPrune", which aims to extends federated averaging to include a pruning variable that tells if a client should be pruned or not. Li et. al. [32] proposes a similar method using pruning, however where they differ is that they employ the lottery ticket hypothesis [33] in order to divide the models into subnetworks. Vahidian et.al. [15] proposes a method that aims to extend federated averaging, called, SubFedAvg. It aims to take an average of the intersection of the remaining parameters.

3.1.6 Clustering

Clustering is also a popular method to achieve this. However this often results in several subnetworks within the FL network, which can be argued is a good thing and a bad thing. One such method is proposed by Qin et.al. [34] where they divide the FL network into several subnetworks using the local data and the biases of each model to generate a subnetwork, making sure to cluster the similar models together. Luo et.al. [35] is a similar clustering method in that it divides the FL network into subnetworks, however where it differs is that it focuses on efficiency in terms of communication cost.

3.1.7 Personalization

Cho et.al. [36] proposes a new architecture to FL when several devices can be considered the same user (i.e. an individual has several wearable devices). The idea is to consider a collection of devices as a single client, rather than each device as a single client. However what is interesting is that they propose a way to properly select and aggregate the clients after complicating the network by introducing a layer of abstraction. Another approach that was relatively unique was one by Wu et.al. [37] where they propose a method to use bayesian fusion rules to merge trained models to perform optimally in a local environment - meaning it personalizes the global model to the local environment. Mestoukirdi et.al. [38] proposed something they call User Centric Federated Learning, which is a concept where they implemented several server-side methods to further personalize the global model towards individual clients, however this approach affects amount of communication between server and client.

There were also other approaches proposed to handle personalization that were interesting, however the approach to achieve this was less prevalent than others. Such as Kelli et.al. [39] proposing a method to combine FL with active learning in to better personalize the resulting models for their respective environments. In this case active learning being a method to adapt the global model to the client's own traffic. There were also more exploratory papers that popped up during the literature review. Such as surveys of proposed techniques to adapt to federated learning to improve the personalization of the global model. Kulkarni et.al. [11] released one of such surveys where the proposed techniques were as follows: transfer learning, mutli-task learning, meta-learning and knowledge distillation.

Furthermore, it appears that personalization in FL is a huge issue in using it for health-related solutions as the parameters that are important to each individual differ to a large enough degree to cause problems in using generalized models [23, 24], for example Kirsten et.al. [40] explored the effects of using FL as a tool in treatment for OCD and found that personalization is important. Moreover, Wu et.al. [41] proposes a FL framework which specializes in at-home health monitoring, specifically for the aging population, for such a use-case, personalization is vital, due to differences in variables of an individuals health abd their current condition and diagnoses.

Chapter 4

Methodology

In this section the methodology for the thesis will be explained. What will be presented is how the topic was chosen, the methods that were employed to gather information about the topic, how the gathered information was selected for use, how the work was organized and structured over time throughout the semester, the relevant tools that were used for developing the tools that would generate the results and the methodology employed to perform said development.

4.1 Exploration

The first two months of this thesis was used as an exploration phase. This phase consisted of looking into federated learning as a field and explore various topics of interest. Early in the exploration phase it was already apparent that the state of data was vital to the performance of the entire field. This brought the focus towards topics tackling these issues and caused the scope of the exploration phase to gradually shrink over the course of the exploration phase. Furthermore, the were a lack of resources that focused on using federated learning for the client rather than the network as a whole. These factors led to the scope being narrowed down progressively over the span of the exploration phase until it arrived at comparing methods to personalize the global model to the client's environment. Prior to starting the exploration phase it was already decided that it would be a thesis exploring federated learning, as it is a relatively new topic and therefore more avenues for exploration.

After the exploration phase, and the topic of the thesis was properly scoped down, it was time to perform a literature review to gather more information about the topic, as well as know which direction to go with the thesis. For the literature review, a framework was used. The systematic literature review borrowed ideas from the framework PRISMA [42]. Preferred Reporting Items for Systematic Reviews and Meta-Analyses is a framework that assists in performing a complete systematic literature review. It provides a checklist of the necessary items to properly complete it, as well as steps that explains how to process and select reliable sources of information. While PRISMA might not be specifically constructed to perform literature reviews for computer science topics, most of the steps defined in PRISMA still apply. This framework was chosen due to ease of use and convenience. Having an available checklist [43] that explains each step in the list and what exactly to look for while performing the review is immensely helpful. The methodology of PRISMA will be further elaborated upon in section 4.4.

4.2 Plan

After the the exploration, and the planning of the literature review - a proper work plan was developed to keep a semblance of structure to the work that has to be done. The plan can be seen in Table 4.1.

At glance value this plan appears to be reasonable. However, it is not - looking at the plan now, in retrospect, a lot of the work is only completed in the last month and a half. This means that if anything prior to these last few weeks is left unfinished it will leave a lot more work those last few weeks compared to the first few, which is how it turned out. During the first half of the plan there were some work that turned out incomplete by the end of its planned working time, which meant that some of the plan had to be pushed back and this only snowballed throughout the entirety of the plan and left the last few weeks to be very busy compared to the first weeks. If I were to redo this thesis there would be some changes to the plan. Firstly, I would limit the exploration period to the first month only, giving me another month to work with. With another month at disposal I would add another week to spend on the literature review, the implementation and then spend the remaining time as a buffer time at the end to finish up the thesis properly.

Date	Objective	Milestone
10.03 - 17.03	Write a plan, continue development, start researching the properly scoped topic	Finish plan
18.03 - 24.03	Continue development, research and start writing about the background and research	
25.03 - 31.03	Finalize the groundwork for the code, finish research and writing of background and research	Finish lit. review and ch 2 and 3
01.04 - 07.04	Finish coding and start experimentation, properly develop the structure for the thesis	Finish implementation
08.04 - 14.04	Have proper results from experiments and write about implementation, experiments and research	
15.04 - 21.04	Cont. and finish writing from last week	Finish ch 5
22.04 - 28.04	Write the sections about results, discussion and conclusion	
29.04 - 05.05	Cont. and finish from last week	Finish ch 6, 7 and 8
06.05 - 12.05	Write the introduction and abstract and finish	Finish abstract and ch 1
13.05 - 19.05	Write and finishup whatever is remaining and start properly organizing the thesis	Finish ch 3 and 4
20.05 - 26.05	Finalize	
27.05 - 01.06	Finalize	Hand in the finished thesis

Table 4.1: How the thesis should have gone according to plan.

4.3 Implementation strategy

During the development of the implementation, various software principles were used to achieve a more structured, efficient, and overall better result. The process of the entire thesis followed something resembling an agile development framework, such as Scrum. While principles from scrum were borrowed, it wasn't implemented in its entirety due to several reasons, and the big one being there was only a single member on the team, making some of the implementations less useful. However, what was used was retrospectives and focused sprints to an extent. The scrum framework implements these two principles. A sprint is a specified time interval with a defined set of goals that are to be achieved during that time interval. A retrospective is a meeting that is performed at the end of a sprint, where the goal is to identify problems that may have occurred or been identified, as well as identify what went well. After the exploration phase, each week was designated its own focus area for the thesis. During this week there were loosely defined goals, such as working on and finishing the background section of the thesis or implementing the proof-of-concept and then the next week would be to generate results and write about those results. The retrospectives were performed weekly with the supervisor, where I would summarize the progress from the previous week and then discuss the next steps. As can be seen, there wasn't a strict implementation of Scrum in this thesis due to the aforementioned reasoning. Principles from the programming paradigm object-oriented programming (OOP) was used for designing the architecture of the implementation. Designing classes that that will be instantiated as objects that contain all its own essential functionality is the core of OOP. However, while OOP has some useful patterns to implement, it wasn't blindly followed, as some of the functionality between the server class and client class especially share some functionality, it would be more efficient to simply develop a general method to be used by both rather than spending time perfecting each class in accordance with OOP.

4.4 Literature review

4.4.1 Criteria

Two sets of criteria were decided upon for review, inclusion- and exclusion criteria.

Inclusion criteria

For papers to be included in this review they had to pass a set of inclusion criteria:

1. The paper is available in English.
2. The overarching topic of the paper is federated learning.
3. Focuses on ways to personalize the global model.
4. Paper is peer-reviewed and fully published.

Exclusion criteria

1. Paper strays from inclusion criterion 3.
2. Unavailable in English.
3. Is grey literature, or any other kind of literature not peer-reviewed.

4.4.2 Information sources

For this literature review the papers were gathered from the scientific literature database Scopus.

4.4.3 Search strategy

The main focus of the search strategy was to find articles relating to federated learning, then narrowing it down to articles that would address the issue of personalizing the global model to the local environment. After refining and testing out different search queries the final key words ended up being: *federated learning* as the article had to be about federated learning, then it would further have to include *client update* or *client aggregation* or *personalization*. The reason it had to include any of the latter three terms was to narrow the papers down to more client related papers. Furthermore the paper had to be classified as a conference paper, article or review. There was little reason to include publish year as federated learning would already reduce the year interval significantly.

4.4.4 Paper selection

After performing the initial search there were 212 papers found, for this initial search the paper had to be in English. Then it was checked for duplicates, where none were found. It was then enforced to include the key word "federated learning" or "personalization" in the key word list, and the title had to be deemed relevant - which further reduced it to 56. The next step was to review the abstract and the conclusion if the abstract was unclear, which reduced it down to 40. Then the full texts were read, which did end up excluding some papers, finally ending up with 26 papers in total.

4.4.5 Data collection

The data collection process was guided by the research questions defined in section 1.1. The information gathered from the papers were methods relating to locally optimizing the global model and dealing with non-i.i.d. data.

4.4.6 Synthesis

The information extracted from the papers will be synthesized into a table summarizing the methods used for answering the research questions. The results can be seen in Table 3.1.

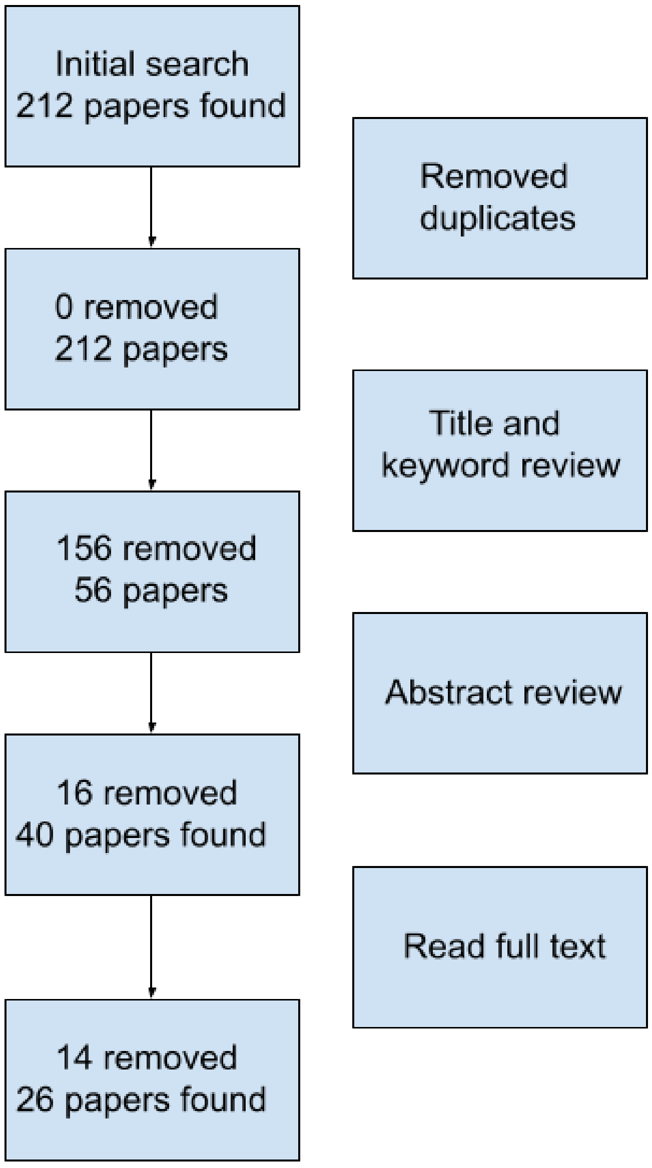


Figure 4.1: Illustration of the selection process

4.5 Prioritization

One of the main features of federated learning is that it allows machine learning to be performed on several machines. However, this is a feature which is irrelevant to the chosen topic and will add needless complexity to the development of the software as it will introduce several additions that will only add hardship. Implementing this feature would first require the development of a proper network communication feature, which would include numerous complications. Next would be access to several machines with hardware able to run machine learning algorithms. It could be argued that one could simulate the network connectivity by running the clients on the same machine, but then why add the complexity of implementing it at all. Furthermore, the desired results generated from the developed tool will be numbers that are mostly unaffected by the communication itself, unless said communication goes wrong or functions improperly. With these things in mind, it was deemed an unnecessary amount of work as well as an unnecessary risk. How exactly the communication was simulated in this thesis will be further explained in section 5.4.

Chapter 5

Implementation

In this chapter the implementation methods and strategies will be presented. It will contain the tools used for implementing the software, programming language, framework and libraries. Furthermore, the approach used in developing the software itself, as well as workarounds for certain features that are usually included in federated learning network, but will be skipped for a reason, the reason for not including certain features will be elaborated upon in their corresponding sections.

5.1 Requirements

Before introducing the tools that were used for the implementation, the requirements that these tools satisfy will first be clarified, as each tool has its own purpose for being chosen. The first tool that will be needed to implement this proof-of-concept is the programming language, which needs to have a set of features to even be considered. These features are the ability to handle data structures such as matrices and lists, the ability to perform simple to complex mathematical operations on said data structures, furthermore the language needs to have accessible machine learning frameworks. Furthermore, the machine learning framework needs to be able to train convolutional neural networks, as the experiments will be performed on datasets that consists of images. This framework also needs to have available documentation and/or other resources that will assist in constructing a fitting ML model to perform required training. The reasoning for many of the decisions made will be discussed in section 7.2

5.2 Tools

In this section the tools used for development will be presented. The tools that will be presented are Python, Tensorflow, NumPy.

5.2.1 Python

Python is a programming language that has over the years become the standard language for data science. It's a language that includes numerous libraries and frameworks that allows for an easier and more efficient development of data science projects such as machine learning. The main reason for choosing this language is just for that reason, as it tremendously reduces the amount of required work to be done and reduces the number of potential errors. Another important reason for choosing Python is the available documentation for both Python itself and the available addons one could access using Python. The chosen version for this thesis is Python 3.9.2.

5.2.2 Tensorflow

Tensorflow (TF) is an open-source machine learning framework for Python that contains a comprehensive number of tools and functionality for building and training ML models. It is one of the most widespread and used machine learning frameworks along with PyTorch. TF implements most of the common machine learning algorithms as well as the most important classes and data structures necessary for this thesis, allowing the focus to solely be on answering the research questions rather than developing an entire framework before one could start. TF also has the added benefit of numerous and easily accessible documentation. The chosen TF version is 2.6.0.

5.2.3 NumPy

Python has the drawback of having a lacking default library to handle complex mathematical data structures and operations efficiently if it could even handle it at all. This raises the need for a library that extends the mathematical functionality of Python. NumPy which is a library that implements more efficient arrays and matrices, as well as methods to perform operations on these data structures. All the matrix computations done in this thesis is performed using NumPy. The version of NumPy used was 1.19.5

5.3 Data distribution

There have been several mentions about how data is distributed in a dataset accordance to i.i.d. in this thesis. The goal of this thesis is to identify efficient ways to personalize the model to a specialized environment. And the term specialized environment in this context is synonymous with a non-i.i.d. dataset on the client as this will create a "natural" bias in the local model towards certain classes, thus making it perform better than a generalized model on these classes, hence the local model being specialized. With that reasoning in mind, the next step would be to identify how much should the model skew in some directions. Should it be an extremely skewed distribution, i.e. there is only a single class on a client, or a distribution that is only slightly skewed. Finding a perfect "natural" distribution in a simulated environment is a study of its own and outside the scope of this thesis, however it is still possible to do a best estimate. Before making an estimate - there are some steps that can be assumed. First off, there are probably none, or very few distributions that look similar, making it safe to assume that each distribution is unique. Next assumption would be that extreme distributions in either ends, skewed or general, is an unnatural distribution, making the second assumption that extreme distributions should be avoided. And the last assumption is that there is a similar amount of data samples in each client. This assumption is not a realistic assumption, however it decreases the complexity of the task, as the number of samples would be another variable in the experiments that would have affect the results to a big enough extent to be problematic. These are the set assumptions the data distributions in the experiments are operating under, showing a clear structure to how each client will be handling their data.

5.4 Communication

In this project there was no perceived need to implement a full federated learning framework complete with server to client communication, as it was mentioned in section 4.5, it was deemed an unnecessary amount of work and risk. Rather the relationship between server and client was enforced and simulated through the design of the software architecture. The design choices will be further elaborated upon in section 5.5. By employing principles of OOP, it made it possible to efficiently emulate what this relationship would look like in a real environment, however without the added complexity of implementing it, and of course without the actual functionality being present. For this thesis, there was no need to implement any kind of persistent storage, as everything existed in the same memory and had access to all data at all times if wanted.

5.5 Architecture

As mentioned in chapter 4, design patterns from OOP were adapted to be used in this project. In the finalized design, the software architecture consisted of three classes. One class for keeping track of the server and its functions, one for the clients and one for the dataset in each client. During execution of the software, there would be one instantiated server that would hold a list of instantiated clients, and then each client would have their own unique dataset. In this section each implemented class will be explained.

5.5.1 Server

The server can be considered the controller class, as it is initiating the entire federated learning process. It's responsible for initiating training rounds with the clients, generating a global model through federated averaging and then update each client with the current global model. The server class will only store the weights of the global model and a list of current clients.

5.5.2 Client

The client's main responsibility is producing data. It will receive a message from the server to start training and then take the input in the shape of data received from the dataset class to perform machine learning. The result of the training will be transmitted to the server to be aggregated, and then that result will be transmitted to the client. The client can then use that global model to perform one of the client-side aggregation methods if called. The client stores the weights of the local model, global model, and a dataset.

5.5.3 Dataset

The dataset class behaves purely as a data class. It only has methods that allows interaction with the data it stores. It stores training data, training labels, testing data and testing label. The methods used for separating and making distributions for the data are separate utility methods independent of this class, as the goal for this class is to be a pure data class. This is by design as there is no purpose for these methods outside of initializing the dataset class.

5.5.4 Multi-model approach

Ensemble learning, which as been introduced in previous sections, is the paradigm of using several machine learning models for the same task. For this implementation the idea is to use both the local- and the global models and use both of their strengths to achieve the best possible results. The main way this is done, is by using the predictions from the local model for the specialized cases and the global model for anything else. However, to achieve this a method to calculate the class wise performance is needed. This was implemented by calculating the class wise f-measure for both the local model and the global model using the local validation set to generate true positives, false positives, true negatives, and false negatives for both models. The reason f-measure was used was due to it being a product of precision and recall which will aid in measuring the accuracy of the models on each class present on the client.

The metrics are caluclated as follows:

$$\frac{truepositive}{truepositive + falsepositive} = Precision \tag{5.1}$$

$$\frac{truepositive}{truepositive + falsenegative} = Recall \tag{5.2}$$

$$2 \cdot \frac{precision \cdot recall}{precision + recall} = F - measure \tag{5.3}$$

these three formulas are calculated each round for each class. The predictions for each model will then be compared to each other and the model with the highest f-measure for its respective prediction is the one that will be chosen. If both happen to have an equal score, the global model will take precedence.

5.5.5 Weighted averaging

A weighted average would help generate a more accurate model due to its ability to suppress the influence of factors deemed less important, but still a factor. On one hand the local model is biased towards certain classes due to imbalance in the data and this should be suppressed in one way to even out the results, but on the other hand this bias is a result of its environment, meaning there is a justifiable reason for this model to be biased and therefore the global model's generalizing effect should be suppressed to an extent. The implementation of weighted averaging is not a difficult implementation; however, the complicating part of this implementation comes in the weighing - how to properly determine the weighing of the models. If the models have an equal weight, its just an average, which would only give the resulting model a slightly better performance if both the models were very similar to begin with. Always favouring the local model would give it a better performance for specialised cases, and the opposite would be true for general ones, assuming the federated learning network is populated enough to generate generalized models.

There are many ways to go about determining the weights, however for this implementation it was inspired by the implementation of federated averaging - as it can be considered a weighted average based on the amount of data samples on each client. However, for this experiment this relationship was turned inverse - as in the scaling factor is related to how much of the data used for the model is based on the client's data. In federated averaging the weight for each client would be calculated as follows:

$$\frac{S_c}{S_s} = W_c \tag{5.4}$$

where S_c denotes the amount of data samples on the client, S_s the amount of data samples globally and W_c the weights for that client. This weight can also be used when combining the models on the client-side. The logic behind this is that the extent the local model should affect the averaged model scales with the amount of data used to train it - as the global model is a product of a lot more data than the local one, letting the global model be the primary factor in the equation will result in a better model in most cases. With the local weight being calculated using the same methodology as federated averaging, a method to calculate the weight for the global model has to be developed. The weight for the model is simply calculated as follows:

$$1 - W_c = W_g \tag{5.5}$$

Chapter 6

Experiments and Results

Before doing any experiments there has to be an established baseline first. This baseline will serve as a common point of reference to better evaluate the measured performance of the methods. The first point of reference will be a centralized machine learning model trained and evaluated on the same dataset, in its entirety, rather than split into pieces for FL. The second established baseline would be the performance of the global model using federated averaging. The reason for choosing this baseline is because it is the chosen aggregation method for all the FL experiments and it produces the global model that will be used for the experiments, and it is therefore necessary to compare the performance before and after the methods in the experiments have been applied. However, what is different is that the data distribution of the datasets will be non-i.d.d., meaning the performance of the global model will be somewhat lower than if it was i.i.d.

6.1 The baseline

In addition to using the federated averaging result as a baseline, there will also be the centralized machine learning baseline. This is to draw a comparison from the results of the experiments to centralized machine learning, to compare how it performs. It is expected that the results will perform worse than centralized ML - however, the goal is not to surpass centralized ML, but rather to try and close the gap to some extent between federated learning and centralized machine learning.

For the CIFAR-10 dataset, the centralized model returned the accuracy of 0.656 in the last epoch - which can be seen in Figure 6.1. The parameters of the centralized model were as identical as possible to federated learning one. This means there were 100 epochs, with a learning rate of 0.01, stochastic gradient descent (SGD) was chosen as the optimizer, the loss function chosen is categorical cross entropy and the accuracy chosen is categorical accuracy. The accuracy was calculated on the validation set.

For the MNIST dataset, the centralized model returned the accuracy of 0.967 in the last epoch - which can be seen in Figure 6.2. The parameters used were

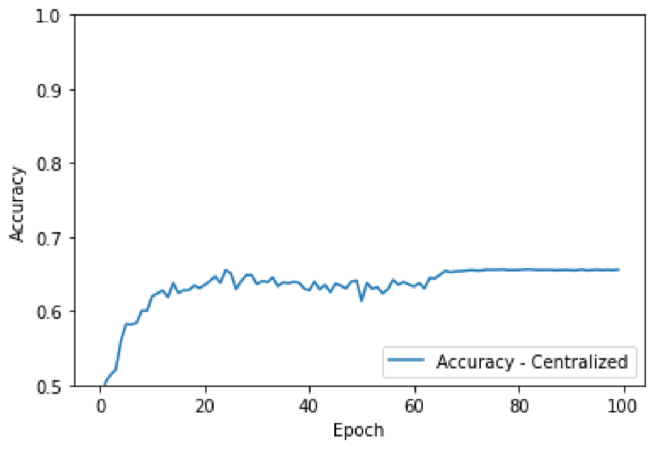


Figure 6.1: The accuracy of the centralized model on the CIFAR-10 validation set.

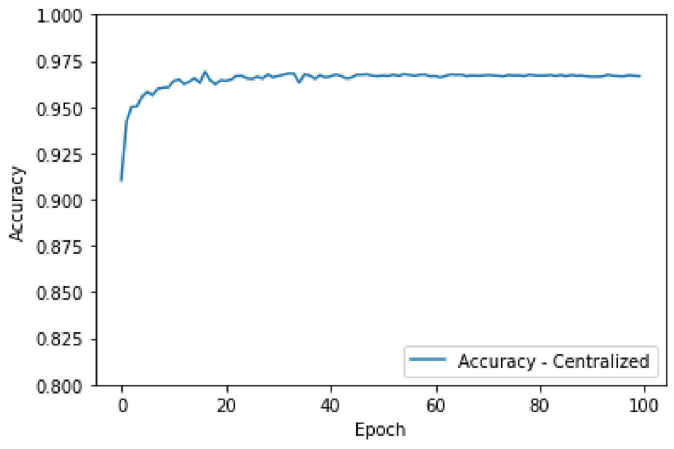


Figure 6.2: The accuracy of the centralized model on the MNIST validation set.

identical to the ones used for CIFAR-10. The reason for this is to have as few variables as possible between the experiments. The same learning rate, loss function, metrics and optimizer were chosen for the MNIST model as the CIFAR-10 model. The accuracy was calculated on the MNIST validation set.

6.2 Experiments setup

It was decided that there would be ten clients in total, this was purposeful design for several reasons. The first and most impactful reasons were hardware limitations, as the video memory required to be able to run ten clients compared to a hundred vary, as well as the time to complete a hundred communication rounds on ten clients versus a hundred is also significant. Secondly, ten clients would al-low each class to be represented by its own specialist – which will presumably show different results for each client. The validation set was split in an equal manner, which will affect the metrics as models with a higher accuracy on the over-represented classes will achieve better resulting metrics.

6.2.1 Datasets and data distribution

The datasets used in the experiments are MNIST as well as the CIFAR10 dataset. The reasons for picking these two datasets are several. First of all, they are public and easily accessible. Furthermore, from section 3.1 they were two of the most commonly used datasets for performing FL experiments, providing previous work and documentation of approximately how well federated averaging will perform on the data, as one of the main challenges of FL currently is finding suitable datasets. Lastly, the datasets are somewhat similar, but still different. Both CIFAR10 and MNIST has ten different classes, but the images themselves are different, the MNIST images are 28x28x1 and the CIFAR10 images are 32x32x3. What is different with the data used for these experiments is that the distribution will be non-i.i.d., however as stated in [9], this will not affect the end result of the model, but rather the rate of convergence – meaning that it will take longer for the global model to converge due to the local models being heavily biased. Of course, the rate of convergence will be affected by the degree of non-i.i.d. as extreme cases will generate more noise during the aggregation. However, this is not an issue for these experiments, as the degree of non-i.i.d. will be limited, as extreme cases will be considered rare and therefore less relevant. The data distribution was designed as follows:

- Order and separate each class in the dataset.
- Divide each class into 10-15 fragments of equal size (size meaning amount of data points).
- Assign each client one fragment of each class.
- Select one client to be a specialist of one class.
- Assign the 1-5 extra fragments to its corresponding specialist.

6.2.2 Multi-model approach

The multi-model experiment was performed with ten clients, where each client had a learning rate of 0.01, where it would decay by a factor of the learning rate divided by the amount of communication rounds for each communication round. Each client also held a unique dataset with a similar amount of samples where each client model would train on the dataset for five epochs each communication round. The metrics would then be calculated locally for each client to measure the performance each round over a set amount of 100 rounds. The measurement metrics would be calculated for both the global model and the local model, and then the results of those would be used to calculate the performance of the combined predictions of those two models. According to expectations the ensemble method shows a slight increase in accuracy which can be seen in the graphs and the table. This makes sense as both models are slightly different and has a slight difference in performance on each class and combining both models' ability to predict should boost the accuracy of the predictions, rather than relying on the predictions from a single model.

Multi-model	Local model	Global model
0.76	0.76	0.72
0.69	0.70	0.67
0.43	0.41	0.37
0.35	0.38	0.35
0.39	0.44	0.40
0.50	0.45	0.50
0.67	0.69	0.69
0.61	0.58	0.65
0.61	0.62	0.58
0.66	0.65	0.67

Table 6.1: The class-wise f-measure scores from each model.

Table 6.1 shows the class-wise f-measure scores for each of the ten classes in the CIFAR10 dataset in a single client. It shows that on average the multi-model scores higher than both models, which makes sense as it is a combination of both models' predictions. However the reason the f-measure is not higher than both other models in all areas is due to the f-measure being recalculated using the combined predictions, rather than it being a one-to-one mapping of the metrics after the predictions - which will cause the scores to be somewhat different than first expected.

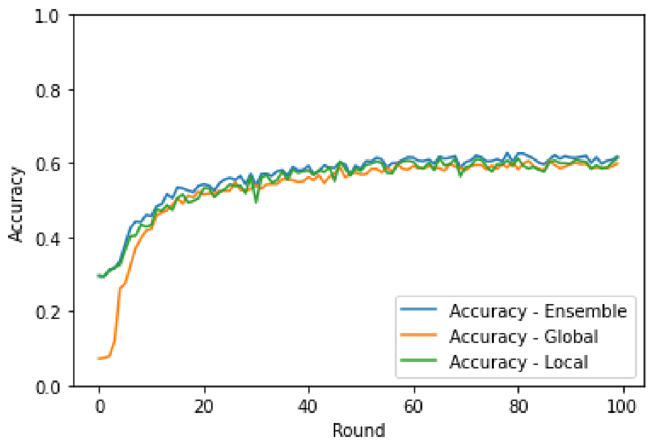


Figure 6.3: One of the clients from the multi-model approach - CIFAR10 data

Figure 6.3 and Figure 6.4 show the plots for one client each for each dataset. First to address the differences in the figures. There's an overall lower accuracy in the CIFAR10 compared to MNIST, this is due to the increased complexity in the CIFAR10 data as explained in subsection 6.2.1 - there is also a quicker convergence rate due to the same reason, however this is also affected by the class distribution of each client. As for what the plots show in common, is that there is a slight increase in overall accuracy for the multi-model approach, with the line for the multi-model approach scoring at the level of the local model or higher.

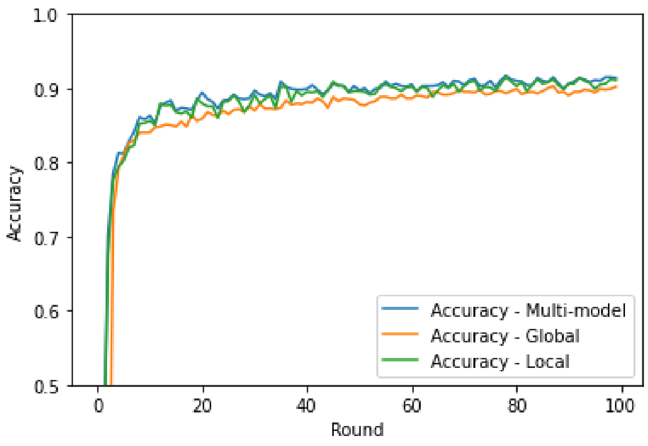


Figure 6.4: One of the clients from the multi-model approach - MNIST data

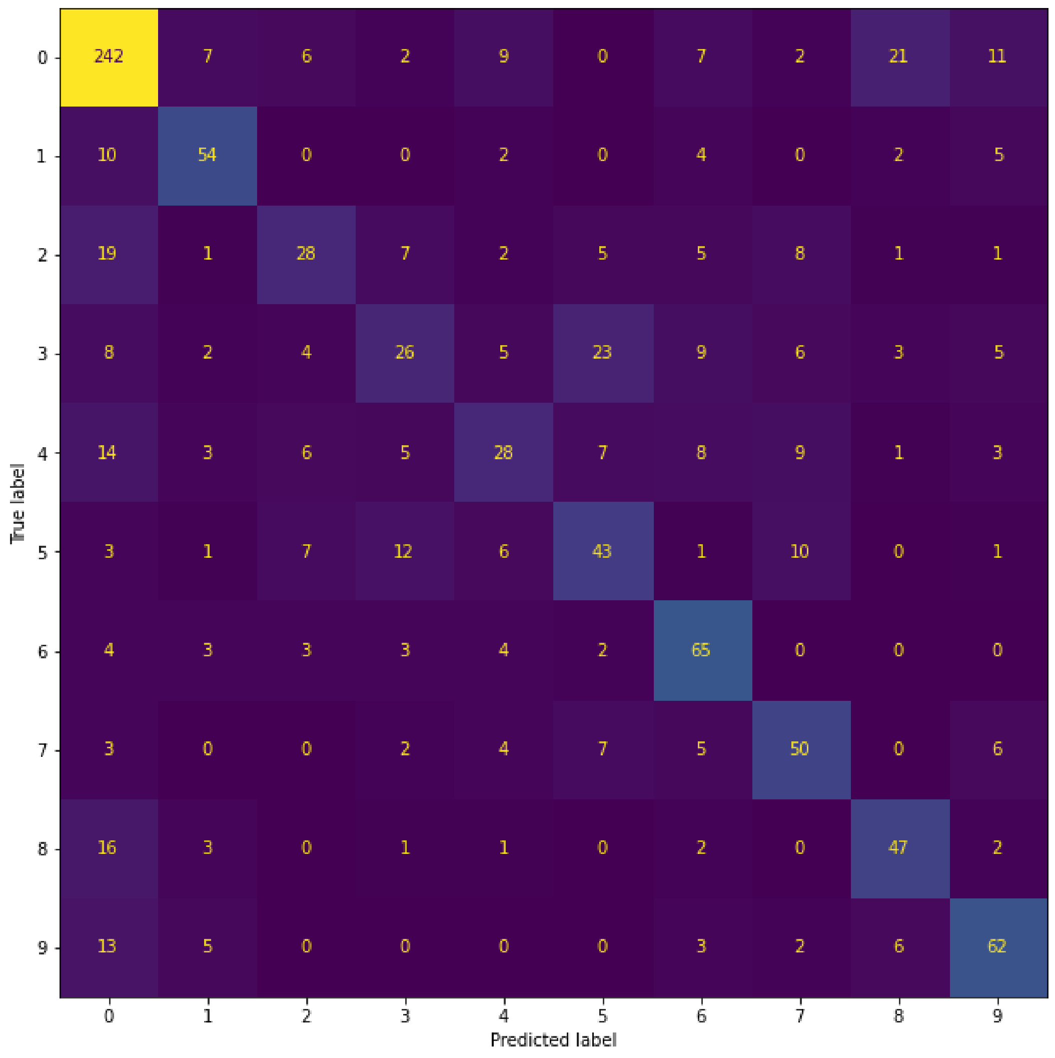
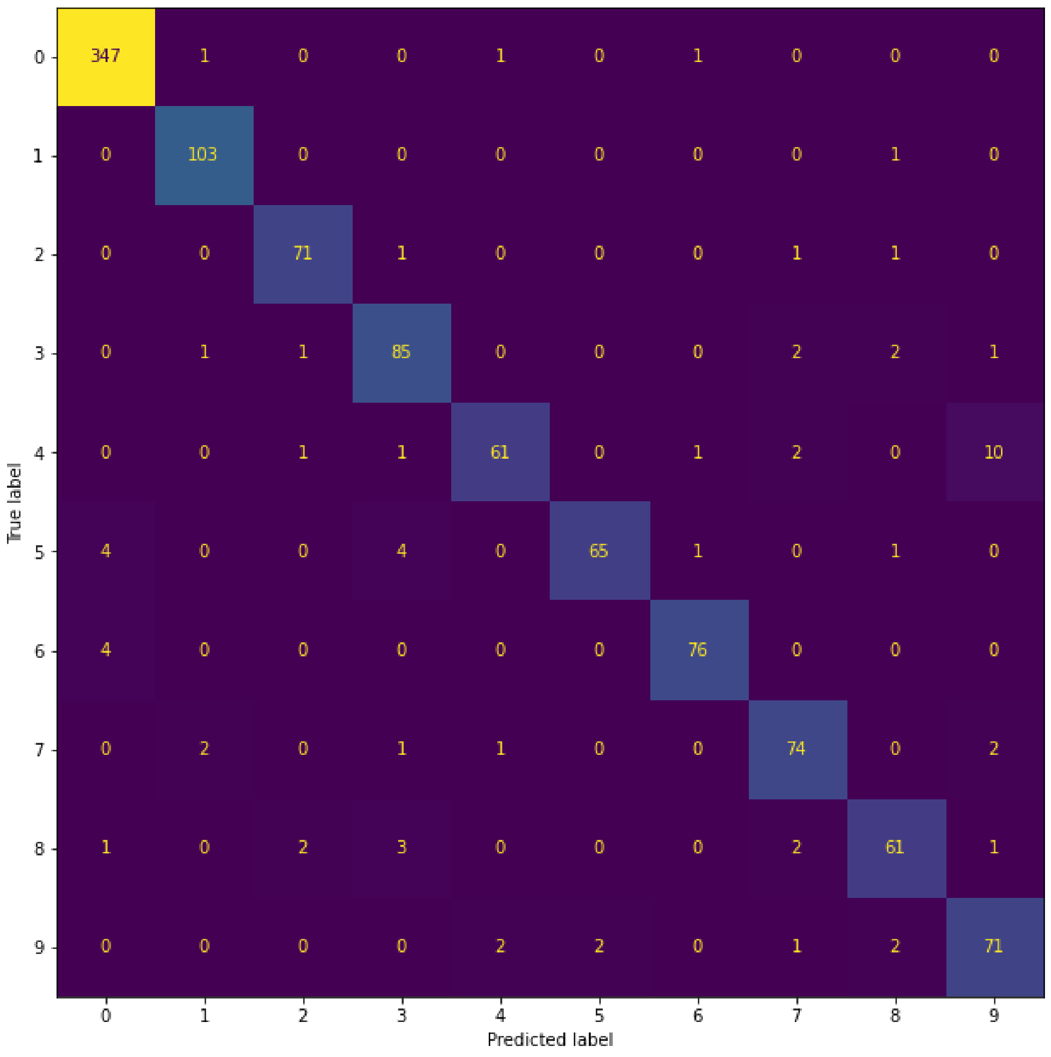


Figure 6.5: Confusion matrix produced from the same client using the multi-model approach - CIFAR10 data

Figure 6.5 and Figure 6.6 shows the confusion matrices produced from the outputs of one of the clients. The diagonal across shows the amount of correct predictions. There's a clear bias towards the label 0 in both examples, this is intentional as both clients represent the specialist for the class labeled as 0 in both examples. However, what it also shows is that one is more often correct than the other, this is due to the complexity of the data as explained earlier.



	0	1	2	3	4	5	6	7	8	9
0	347	1	0	0	1	0	1	0	0	0
1	0	103	0	0	0	0	0	0	1	0
2	0	0	71	1	0	0	0	1	1	0
3	0	1	1	85	0	0	0	2	2	1
4	0	0	1	1	61	0	1	2	0	10
5	4	0	0	4	0	65	1	0	1	0
6	4	0	0	0	0	0	76	0	0	0
7	0	2	0	1	1	0	0	74	0	2
8	1	0	2	3	0	0	0	2	61	1
9	0	0	0	0	2	2	0	1	2	71

Figure 6.6: Confusion matrix produced from the same client using the multi-model approach - MNIST data

Multi-model	Local model	Global model	Extra fragments
0.619	0.614	0.599	3
0.625	0.601	0.624	3
0.567	0.564	0.537	3
0.575	0.558	0.572	1
0.591	0.572	0.579	2
0.597	0.578	0.580	2
0.634	0.616	0.599	2
0.597	0.590	0.576	3
0.643	0.630	0.608	4
0.594	0.583	0.604	1

Table 6.2: The resulting accuracy from the final communication round for each client - CIFAR10

Table 6.2 and Table 6.3 shows the resulting accuracy scores from each client at the last communication round. It shows that it varies between the local and global model which one performs the best on the client's validation dataset, which is expected with the data distribution of each client being unique. The degree of non-i.d.d. varies between the clients, and its reflected in the results - the clients where the global model is outperforming the local model is where the data is more balanced and vice versa. The data distribution for each client is randomly generated, however for this run the distribution is documented at the far right column and while it is not immediately obvious, over several runs the pattern is consistent with the distribution - in this distribution it shows that the biggest discrepancy in the local models favor is when the distribution is the most unbalanced. However, the table also shows that the multi-model approach outperforms both models in all clients by a small margin, the reason for why the small margin will be addressed in the discussion section, but what can be said now is the reason why it is outperforming both models. The reason for the multi-model approach outperforming both models is simple, it is because this approach is very simple in concept, and effective. It combines the predictions from both models, and decides which prediction to use based on a class-wise metric, which in this case is the f-measure the models score on each class.

Multi-model	Local model	Global model	Extra fragments
0.940	0.939	0.936	4
0.917	0.912	0.917	1
0.919	0.917	0.914	4
0.921	0.919	0.921	1
0.914	0.911	0.902	3
0.923	0.926	0.927	2
0.941	0.937	0.942	2
0.941	0.944	0.939	3
0.924	0.925	0.922	4
0.944	0.942	0.939	3

Table 6.3: The resulting accuracy from the final communication round for each client - MNIST

6.2.3 Weighted averaged approach

The experiment using the weighted averaging approach was set up to be as similar as possible to both the centralized and the multi-model approach. This means that it also had ten clients with the same parameters as the ones mentioned in subsection 6.2.2.

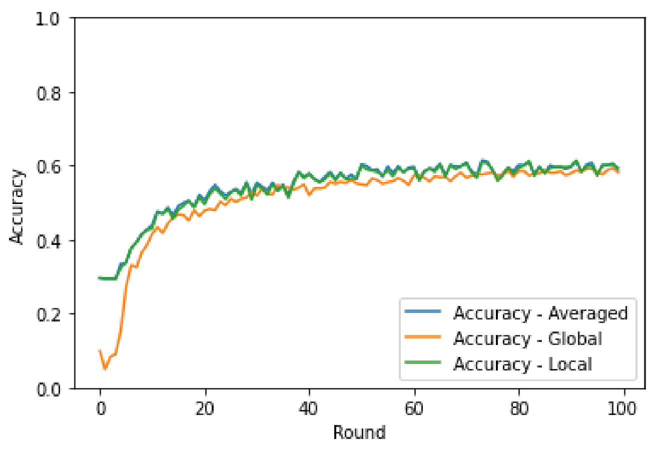


Figure 6.7: One of the clients from the weighted averaging approach - CIFAR10 data

Figure 6.8 and Figure 6.7 show a likeness to either of the other models in terms of accuracy. From the plots alone it's hard to say if there is any change at all in the performance as it is similar enough to not show any change at the current scale. To be able to tell at all if there is any increase at all in accuracy a simple visualization is not sufficient to draw a comparison between the models in this case.

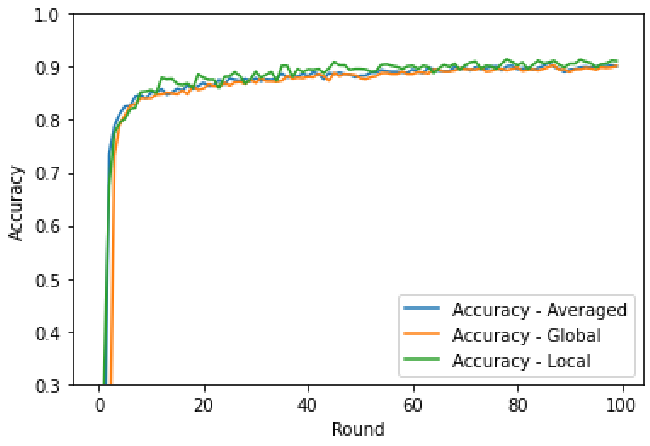


Figure 6.8: One of the clients from the weighted averaging approach - MNIST data

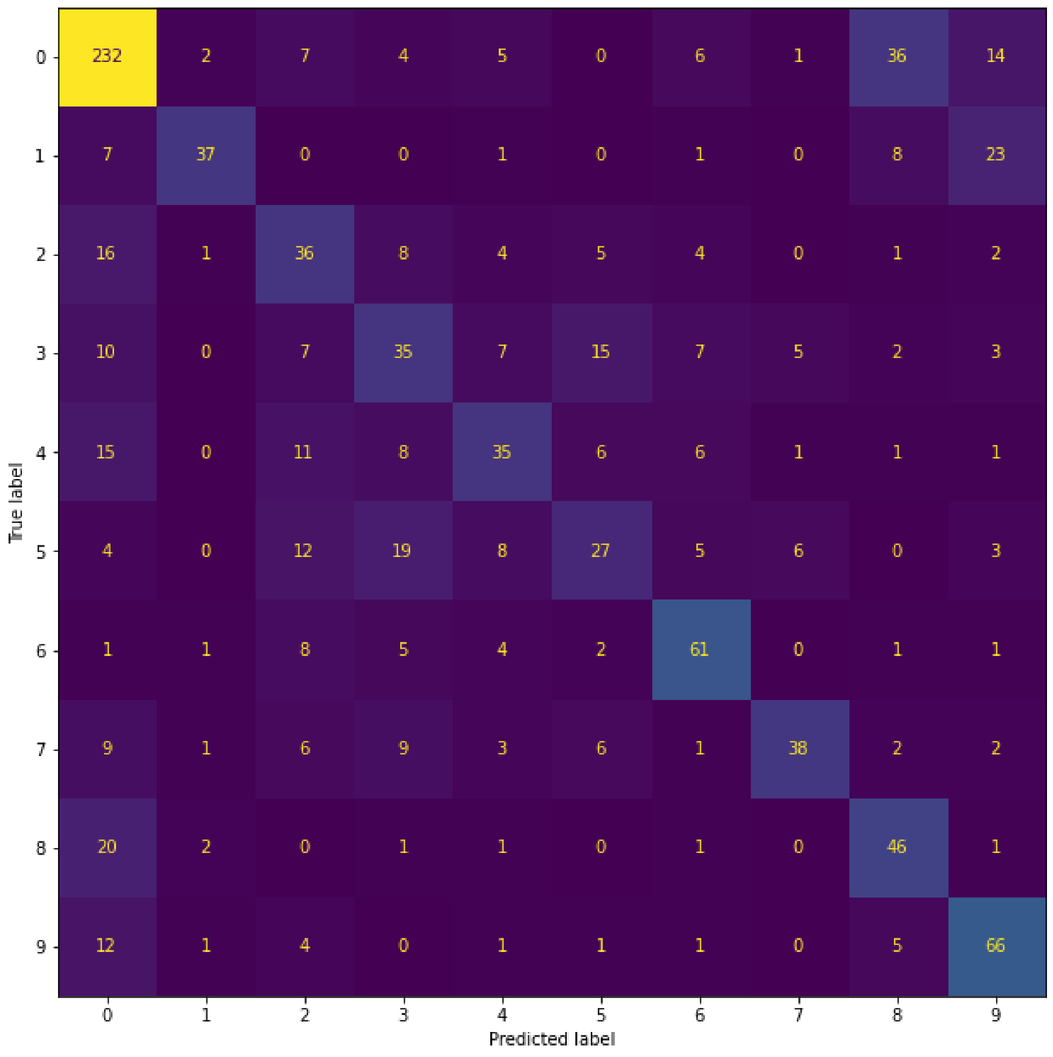
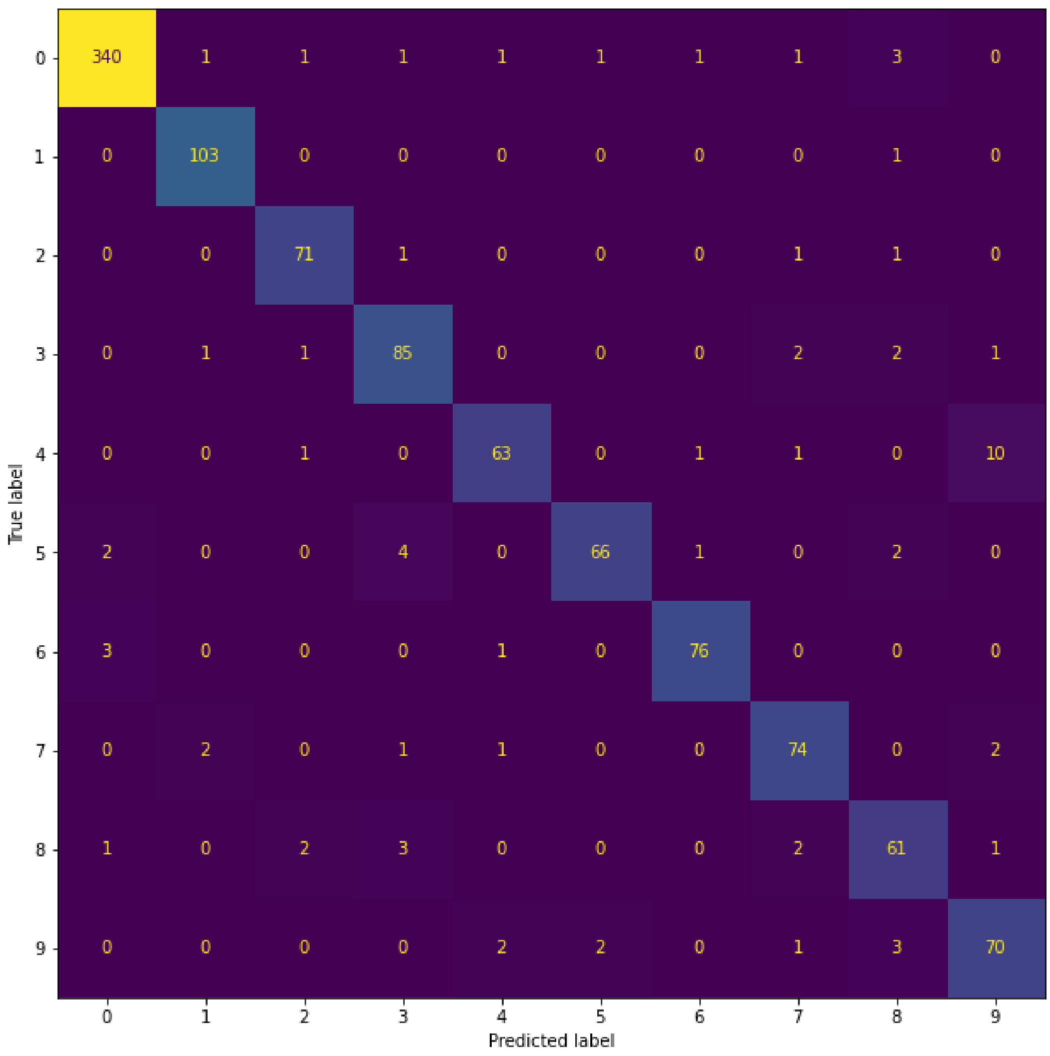


Figure 6.9: One of the clients from the weighted averaging approach - CIFAR10 data

Figure 6.9 and Figure 6.10 show the confusion matrices of the predictions of the models that specialize in the class labeled as 0. In Figure 6.9 the model seems to be slightly confusing the labels 0, 1, 8 and 9. The reason for this is hard to say, due to not knowing exactly which image it is predicting wrong - however, the class names for these labels are airplane, automobile, ship and truck, respectively, from this it seems probable that the model is being confused by shared features between these classes. However, both matrices visualize the expected results, which is a bias towards the specialization and how much more representation the specialized class has compared to the other classes as the matrices are generated using the local validation sets.



True label \ Predicted label	0	1	2	3	4	5	6	7	8	9
0	340	1	1	1	1	1	1	1	3	0
1	0	103	0	0	0	0	0	0	1	0
2	0	0	71	1	0	0	0	1	1	0
3	0	1	1	85	0	0	0	2	2	1
4	0	0	1	0	63	0	1	1	0	10
5	2	0	0	4	0	66	1	0	2	0
6	3	0	0	0	1	0	76	0	0	0
7	0	2	0	1	1	0	0	74	0	2
8	1	0	2	3	0	0	0	2	61	1
9	0	0	0	0	2	2	0	1	3	70

Figure 6.10: One of the clients from the weighted averaging approach - MNIST data

Weighted average	Local model	Global model	Extra fragments
0.587	0.594	0.581	3
0.608	0.602	0.603	3
0.553	0.558	0.530	3
0.571	0.572	0.581	1
0.601	0.604	0.574	2
0.597	0.596	0.590	2
0.597	0.601	0.603	2
0.614	0.618	0.603	3
0.637	0.641	0.624	4
0.561	0.562	0.565	1

Table 6.4: The resulting accuracy from the final communication round for each client - CIFAR10 data

As the plots do not show any difference in performance, it is better to directly look at the numbers in Table 6.4. From the table it is visible that this approach is consistently outperforming the global ever so slightly, however it is also being consistently slightly outperformed by the local model. It can be seen that this is the case for both datasets in Table 6.5 as well.

Weighted average	Local model	Global model	Extra fragments
0.935	0.939	0.936	4
0.912	0.912	0.917	1
0.913	0.917	0.914	4
0.924	0.919	0.921	1
0.901	0.911	0.902	3
0.928	0.925	0.927	2
0.942	0.937	0.942	2
0.937	0.944	0.939	3
0.927	0.925	0.922	4
0.946	0.942	0.934	3

Table 6.5: The resulting accuracy from the final communication round for each client - MNIST data

Chapter 7

Discussion

In this chapter the various methodology, implementation and results will be discussed. I will talk about what I think went well, various processes that went behind the more impactful decisions and finally what can be improved. First I will discuss how I think the chosen methodology worked out for this thesis, then the decisions and thoughts behind the implementation and its design and then the results and how it turned out. Finally at the end of this chapter I will discuss to what extent the research questions proposed in section 1.1 has been answered.

7.1 Chosen methodology

The first two months of this thesis was used for exploration as federated learning was a somewhat new topic to me and I needed the time to get properly familiarized with the current research directions and so on. In retrospect all that time wasn't that well spent and it would've been better if I started working properly earlier and had some extra time to work on the thesis, however it did help me gain a better perspective on the topic - it is difficult to say if that perspective was all that helpful in the long run and if the time would have been better spent being focused.

The chosen working methodology for this thesis was agile inspired, however due to team size it's not practical to implement every aspect of it. Most of the work was split up into its own dedicated sprints and in retrospect these should have been longer, as some of the work did not get finished in its dedicated time slot, which results in either the plan getting further and further behind, or I will be forced to complete a lot of leftover work at the end. In reality, a mix of the two occurred. The last two weeks of the plan, where there were no specific planned objective were necessary, as it was all used to finish the remaining work from the previous weeks.

However, the agile approach was not all bad with the shorter sprints. The weekly meetings with the supervisor were very helpful in maintaining focus due to forcing me to reflect on what has been done, and what I would have to do next.

7.2 Implementation

Starting on this project, I already knew of several of the tools I had to use, such as Python and TensorFlow. However, starting out, I first attempted the implementation with a fully functioning federated learning framework, and after some time of testing out the implementation in this framework and trying to make it work I realised it was unnecessary and only added a lot of extra work and burden to learn more tools, furthermore, it was severely limited by available hardware, as the framework did not simulate the FL process, it fully performed it in parallel. The decision to keep the FL framework would limit the clients to a measly five clients at most with the most basic model architecture, and I would have to learn the ins and outs of the framework to be able to implement the code to function as I wished. Therefore, I made the decision to completely get rid of the framework, this was decided after discussing it with the supervisor of the project. It was concluded that it was unnecessary and only added complications.

With the removal of the framework the the barriers between each client had to be programmed in and self-managed, which is explained in section 5.5. However, with the removal of the framework, it also removed some key features of FL, which is the communication and with the removal of the communication, every client in the network can be considered reliable, as there is no chance for it to drop out without it being specifically programmed into the client. These details were considered to be not that important, as first of all, the task at hand is to improve the client-side accuracy, and with clients dropping out, those naturally become irrelevant regardless. Secondly one of the main issues of clients dropping out is that it will cause unbalanced data during aggregation and that it might create bias during client selection. This was not an issue for this task, as the data distribution is designed to be unbalanced to begin with and there is no client selection mechanism due to the low number of clients, every client will be selected every communication round.

7.3 Results

7.3.1 Multi-model approach

The results show that there is a slight improvement in client-side accuracy, but what does that mean really? Given the pieces that these experiments consist of it has not been elaborated upon what each individual piece means and how it affects the final result. Breaking it down, starting with the data distribution, it has already been explained what it means for the aggregation method. However, for the client it will cause the model to have a bias towards certain classes due to over-representation in the local dataset. This bias is intentional as data produced by people will naturally be biased towards an individual's preferences and habits and creating bias in the model through non-i.i.d. data is an attempt to simulate that. Which brings up the next piece of the experiments, which is the aggregation method.

Federated averaging will over several rounds cause the local models to become more and more similar to each other, slowly ironing out the quirks of each local model. This will diminish the effect of the data distribution in the later rounds - however it will still have a slight effect on the accuracy regardless of the round, which can be seen in the results section. This happens because the global model received after the aggregation process is getting re-trained on the local dataset, but a single epoch is not enough to completely fit it to the local data. This means that the two models used for predictions is the returned global model and an iteration of said global model after an epoch of training on the local data. From this it's possible to see that the accuracy of the clients can be further boosted by incorporating previous models, it's possible to further include every model from the first model to the last. While this could potentially significantly boost the accuracy, it's also a question if the accuracy boost is worth the decrease in efficiency regarding time spent per training round, as time spent per round will only grow with the increase in model counts. This process could of course be optimized in several ways, such as simply saving the evaluation results every round, however at some point it is simply inefficient to keep including models, but this issue is outside the scope of this thesis.

All these aforementioned pieces of the experiment will affect the overall performance of the final local model, however it won't change the actual results by a large margin while federated averaging remains as the aggregation method, as it is still a form of averaging. Federated averaging could be used as an aggregation method for client-focused federated learning, however steps to mitigate the generalization of the global model must be implemented, as discovered in section 3.1, clustering could be one of these methods, where similar clients would create subnetworks that produce their own global models, rather than trying to generalize everything. Another approach could be to mix federated learning and centralized machine learning, this could be achieved by using the multi-model approach used in this thesis with a centralized machine learning model and the global model produced by federated averaging. However, for that to be an alternative, each client must host enough data to be able to produce a centralized model that performs well enough. This removes one of the advantages of federated learning, which is pooling the knowledge from several clients with smaller datasets to simulate a huge dataset. With these things in mind, it shows that the multi-model approach is a valid and flexible way to achieve client-side optimization in federated learning.

7.3.2 Weighted averaging

A weighted average between the local model and the global model benefits more from federated averaging, as opposed to the multi-model approach. This is simply due to the similarity between the local and global model because of federated averaging, as a huge discrepancy between those two would produce a worse result due to differences causing more noise in the product. The main reason for including a weighted averaging method is due to it being similar to how federated averaging works to an extent. Federated average could be considered a weighted averaging method, where the weights are calculated by the amount of data samples in each client and in total. Whereas in the local weighted averaging, this methodology can be translated to measuring the impact of the local data relative to the amount of global data samples - meaning the more data on the client, the more weight for that model.

From subsection 6.2.3 it is apparent that this approach fails to outperform the local model - thus making it a downgrade and eliminates the reason for using federated learning in the first place, as the reason for participating in a FL network is to gain a model that can outperform the local model on the local data. From this it can be concluded that it is not worth using in its current state. However, there is room for improvements for this approach - the calculated weights can be calculated in a different way that can preserve the relationship between the resulting model and the intended bias of the model. Currently this is done by using the amount of samples on the client versus the amount of global samples, but this is not the only way to do this. Something which was not attempted was to produce a weight that can factor in the class-wise distribution of the local dataset, but could result in an improved result. However, with only two models being combined - this is unlikely to have a significant effect. What could result in an improved result could be to include several of the past models - both local and global in the averaging.

7.4 Research questions

While the research questions were more of a tool to guide the research of this thesis, answering them is still important - to showcase that the objective of the thesis was accomplished.

7.4.1 What are the effects of non-i.d.d. data in federated learning?

The main detriment of having non-i.d.d. data is that it affects the efficiency of the training process of the FL network. As mentioned in subsection 3.1.1 it will affect the overall result of the final model and have an impact on the bias of the global model. However, these factors really only apply if the goal is to obtain a general model to begin with.

7.4.2 How are the effects of non-i.d.d. data alleviated in federated learning?

In section 3.1 there are several ways to alleviate the effects of non-i.d.d. data. First and foremost - ways to balance the data out appears to be the most frequent and efficient way to deal with unbalanced distributions. Clustering, as discussed in subsection 3.1.6 is a method to group together similarly balanced data. Furthermore, methods that implement shared datasets or way to selectively choose clients for aggregation are common approaches.

7.4.3 What are the methods for personalization in federated learning?

In section 3.1 several possible approaches to this issue has been proposed already. All of the approaches being previous ML techniques that has already been proved to be effective outside of FL. Furthermore, this thesis introduces two separate approaches that successfully perform personalization on the client-side.

7.4.4 Why or why not is personalizing important?

As was explained in subsection 3.1.7 when there is a lack of personalization in FL there is a better performance on average in multiple environments, however the performance in a single environment is degraded. It was further explained that in domains that require highly personalized models due to discrepancies between individual use-cases.

Chapter 8

Conclusion

In this thesis the topic of personalizing the global model generated in a federated learning network to a client has been researched, and specifically two kinds of methods have been tested out and confirmed to have an effect on the performance. Firstly the research gathered through performing a literature review. The literature review showed that there is a necessity for this issue to be addressed for it to be purposed in certain domains, such as the health domain where the need for efficient and personalized models are a necessity due to variances in an individual's needs. Furthermore, it showed that there are several approaches to this issue and these approaches have been categorized in section 3.1, namely within the subdomains of machine learning known as meta-learning, ensemble learning, pruning, clustering and local optimization. After exploring these topics two approaches have been proposed in this thesis, firstly, an approach inspired by federated averaging, but at client-level and secondly, an approach inspired by ensemble learning, more specifically, temporal ensembling - which aims to combine the output of two iterations of the model. The methodology that was used in order to obtain the results of section 3.1 and decisions prior to implementing the proof-of-concept is elaborated upon in chapter 4. The details of these implementations are outlined in chapter 5, where the design of the implementation is elaborated upon. Finally the results generated from the proposed approaches show that there is an increase in performance in comparison to the global model in the local environment - which was the goal of this thesis, as it proves that these two approaches can be used personalize the global model. Furthermore, there are some suggestions in ways to improve the approaches - as it is flexible in nature and components of it can be replaced with other metrics that might improve performance. There are also some suggested ways to improve the performance in terms of accuracy without changing the metrics as well. This could be done with the multi-model approach through trying different aggregation methods on the server side. It could also be improved through involving more models and including their predictions in the combination of predictions.

To conclude, this thesis has summarized the current approaches to client personalization in the realm of federated learning, as well as introduce two approaches that shows an improvement in accuracy. The multi-model approach from this thesis has also shown that it successfully managed to merge the strengths of both models in regards to maintaining the local model's specialization and the global model's generalization which are two qualities that are important for different reasons, however the focus has been on maintaining the specialization as this is a vital when applying FL to applications where a high degree of individuality is important, such as in the health sector. This approach successfully used principles from ensemble learning, which is a common approach to this topic as discussed in subsection 3.1.2. The averaging approach however, did not yield as much promise from the experiments as the other one, it was not based on any of the approaches researched in the literature review, it was however based on federated averaging which is one of the main aggregation algorithms. While the shown improvement is not numerically high, the approach is sound and could be further improved with the proposed future work and generate better results.

Bibliography

[1] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong and Q. He, 'A Comprehensive Survey on Transfer Learning,' *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, Conference Name: Proceedings of the IEEE, ISSN: 1558-2256. DOI: `10.1109/JPROC.2020.3004555`.

[2] D. Blalock, J. J. G. Ortiz, J. Frankle and J. Guttag, 'What is the State of Neural Network Pruning?' en, p. 18,

[3] O. Sagi and L. Rokach, 'Ensemble learning: A survey,' en, *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 4, e1249, 2018, _eprint: https://onlinelibrary.wiley.com/doi/pdf ISSN: 1942-4795. DOI: `10.1002/widm.1249`. [Online]. Available: `https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249`.

[4] A. Nichol, J. Achiam and J. Schulman, *On First-Order Meta-Learning Algorithms*, en, Number: arXiv:1803.02999 arXiv:1803.02999 [cs], Oct. 2018. [Online]. Available: `http://arxiv.org/abs/1803.02999`.

[5] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu and S. Zhao, 'Advances and Open Problems in Federated Learning,' en, *arXiv:1912.04977 [cs, stat]*, Mar. 2021, arXiv: 1912.04977. [Online]. Available: `http://arxiv.org/abs/1912.04977`.

[6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin and V. Chandra, 'Federated Learning with Non-IID Data,' en, *arXiv:1806.00582 [cs, stat]*, Jun. 2018, arXiv: 1806.00582. [Online]. Available: `http://arxiv.org/abs/1806.00582`.

[7] H. Wang, Z. Kaplan, D. Niu and B. Li, 'Optimizing Federated Learning on Non-IID Data with Reinforcement Learning,' in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, ISSN: 2641-9874, Jul. 2020, pp. 1698–1707. DOI: `10.1109/INFOCOM41043.2020.9155494`.

[8] W. Zhang, X. Wang, P. Zhou, W. Wu and X. Zhang, 'Client Selection for Federated Learning With Non-IID Data in Mobile Edge Computing,' *IEEE Access*, vol. 9, pp. 24 462–24 474, 2021, Conference Name: IEEE Access, ISSN: 2169-3536. DOI: `10.1109/ACCESS.2021.3056919`.

[9] H. B. McMahan, E. Moore, D. Ramage and S. Hampson, 'Communication-Efficient Learning of Deep Networks from Decentralized Data,' en, p. 10,

[10] L. Zhu, Z. Liu and S. Han, 'Deep Leakage from Gradients,' en, p. 11,

[11] V. Kulkarni, M. Kulkarni and A. Pant, 'Survey of Personalization Techniques for Federated Learning,' en, *arXiv:2003.08673 [cs, stat]*, Mar. 2020, arXiv: 2003.08673. [Online]. Available: `http://arxiv.org/abs/2003.08673`.

[12] K. Kopparapu, E. Lin and J. Zhao, 'FedCD: Improving Performance in non-IID Federated Learning,' en, *arXiv:2006.09637 [cs, stat]*, Jul. 2020, arXiv: 2006.09637. [Online]. Available: `http://arxiv.org/abs/2006.09637`.

[13] J. Xiao, C. Du, Z. Duan and W. Guo, 'A Novel Server-side Aggregation Strategy for Federated Learning in Non-IID situations,' in *2021 20th International Symposium on Parallel and Distributed Computing (ISPDC)*, ISSN: 2379-5352, Jul. 2021, pp. 17–24. DOI: `10.1109/ISPDC52870.2021.9521631`.

[14] C. Briggs, Z. Fan and P. Andras, 'Federated learning with hierarchical clustering of local updates to improve training on non-IID data,' in *2020 International Joint Conference on Neural Networks (IJCNN)*, ISSN: 2161-4407, Jul. 2020, pp. 1–9. DOI: `10.1109/IJCNN48605.2020.9207469`.

[15] S. Vahidian, M. Morafah and B. Lin, 'Personalized Federated Learning by Structured and Unstructured Pruning under Data Heterogeneity,' in *2021 IEEE 41st International Conference on Distributed Computing Systems Workshops (ICDCSW)*, ISSN: 2332-5666, Jul. 2021, pp. 27–34. DOI: `10.1109/ICDCSW53096.2021.00012`.

[16] J. Xie, X. Yin, X. Zhang, J. Chen and Q. Wen, 'Personalized Federated Learning with Gradient Similarity,' in *2021 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, ISSN: 2576-8964, Dec. 2021, pp. 268–271. DOI: `10.1109/ICCWAMTIP53232.2021.9674055`.

[17] C. Yang, Q. Wang, M. Xu, Z. Chen, K. Bian, Y. Liu and X. Liu, 'Characterizing Impacts of Heterogeneity in Federated Learning upon Large-Scale Smartphone Data,' en, in *Proceedings of the Web Conference 2021*, Ljubljana Slovenia: ACM, Apr. 2021, pp. 935–946, ISBN: 978-1-4503-8312-7. DOI: `10.1145/3442381.3449851`. [Online]. Available: `https://dl.acm.org/doi/10.1145/3442381.3449851`.

[18] J. Wu, Q. Liu, Z. Huang, Y. Ning, H. Wang, E. Chen, J. Yi and B. Zhou, 'Hierarchical Personalized Federated Learning for User Modeling,' en, in *Proceedings of the Web Conference 2021*, Ljubljana Slovenia: ACM, Apr. 2021, pp. 957–968, ISBN: 978-1-4503-8312-7. DOI: `10.1145/3442381.3449926`. [Online]. Available: `https://dl.acm.org/doi/10.1145/3442381.3449926`.

[19] A. Xue, X. Zhu, J. Wang and J. Xiao, 'Diversified Point Cloud Classification Using Personalized Federated Learning,' in *2021 International Joint Conference on Neural Networks (IJCNN)*, ISSN: 2161-4407, Jul. 2021, pp. 1–8. DOI: `10.1109/IJCNN52387.2021.9533496`.

[20] B. Guo, Y. Mei, D. Xiao, W. Wu, Y. Yin and H. Chang, *PFL-MoE: Personalized Federated Learning Based on Mixture of Experts*, en, Number: arXiv:2012.15589 arXiv:2012.15589 [cs], Dec. 2020. [Online]. Available: `http://arxiv.org/abs/2012.15589`.

[21] K. Thonglek, K. Takahashi, K. Ichikawa, H. Iida and C. Nakasan, 'Federated Learning of Neural Network Models with Heterogeneous Structures,' in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2020, pp. 735–740. DOI: `10.1109/ICMLA51294.2020.00120`.

[22] X.-C. Li, D.-C. Zhan, Y. Shao, B. Li and S. Song, 'FedPHP: Federated Personalization with Inherited Private Models,' en, in *Machine Learning and Knowledge Discovery in Databases. Research Track*, N. Oliver, F. Pérez-Cruz, S. Kramer, J. Read and J. A. Lozano, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2021, pp. 587–602, ISBN: 978-3-030-86486-6. DOI: `10.1007/978-3-030-86486-6_36`.

[23] Y. Chen, X. Qin, J. Wang, C. Yu and W. Gao, 'FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare,' *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, Jul. 2020, Conference Name: IEEE Intelligent Systems, ISSN: 1941-1294. DOI: `10.1109/MIS.2020.2988604`.

[24] T. A. Khoa, D.-V. Nguyen, M.-S. Dao and K. Zettsu, 'Fed xData: A Federated Learning Framework for Enabling Contextual Health Monitoring in a Cloud-Edge Network,' in *2021 IEEE International Conference on Big Data (Big Data)*, Dec. 2021, pp. 4979–4988. DOI: `10.1109/BigData52589.2021.9671536`.

[25] J. Guo, U. Kurup and M. Shah, 'Efficacy of Model Fine-Tuning for Personalized Dynamic Gesture Recognition,' en, in *Deep Learning for Human Activity Recognition*, X. Li, M. Wu, Z. Chen and L. Zhang, Eds., ser. Communications in Computer and Information Science, Singapore: Springer, 2021, pp. 99–110, ISBN: 9789811605758. DOI: `10.1007/978-981-16-0575-8_8`.

[26] A. Jalalirad, M. Scavuzzo, C. Capota and M. Sprague, 'A Simple and Efficient Federated Recommender System,' en, in *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies - BDCAT '19*, Auckland, New Zealand: ACM Press, 2019, pp. 53–58, ISBN: 978-1-4503-7016-5. DOI: 10.1145/3365109.3368788. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3365109.3368788.

[27] Q. Wang, H. Yin, T. Chen, J. Yu, A. Zhou and X. Zhang, *Fast-adapting and Privacy-preserving Federated Recommender System*, en, Number: arXiv:2104.00919 arXiv:2104.00919 [cs], Sep. 2021. [Online]. Available: http://arxiv.org/abs/2104.00919.

[28] R. Balakrishnan, M. Akdeniz, S. Dhakal, A. Anand, A. Zeira and N. Himayat, 'Resource Management and Model Personalization for Federated Learning over Wireless Edge Networks,' en, *JSAN*, vol. 10, no. 1, p. 17, Feb. 2021, ISSN: 2224-2708. DOI: 10.3390/jsan10010017. [Online]. Available: https://www.mdpi.com/2224-2708/10/1/17.

[29] B. Xiong, X. Yang, F. Qi and C. Xu, 'A unified framework for multi-modal federated learning,' en, *Neurocomputing*, vol. 480, pp. 110–118, Apr. 2022, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2022.01.063. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231222000820.

[30] C. Finn, P. Abbeel and S. Levine, 'Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks,' en, p. 10,

[31] Y. Liu, Y. Zhao, G. Zhou and K. Xu, 'FedPrune: Personalized and Communication-Efficient Federated Learning on Non-IID Data,' en, in *Neural Information Processing*, T. Mantoro, M. Lee, M. A. Ayu, K. W. Wong and A. N. Hidayanto, Eds., ser. Communications in Computer and Information Science, Cham: Springer International Publishing, 2021, pp. 430–437, ISBN: 978-3-030-92307-5. DOI: 10.1007/978-3-030-92307-5_50.

[32] A. Li, J. Sun, B. Wang, L. Duan, S. Li, Y. Chen and H. Li, 'LotteryFL: Empower Edge Intelligence with Personalized and Communication-Efficient Federated Learning,' in *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, Dec. 2021, pp. 68–79. DOI: 10.1145/3453142.3492909.

[33] J. Frankle and M. Carbin, *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*, en, Number: arXiv:1803.03635 arXiv:1803.03635 [cs], Mar. 2019. [Online]. Available: http://arxiv.org/abs/1803.03635.

[34] Y. Qin and M. Kondo, 'MLMG: Multi-Local and Multi-Global Model Aggregation for Federated Learning,' en, in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, Kassel, Germany: IEEE, Mar. 2021, pp. 565–571, ISBN: 978-1-66540-424-2. DOI: 10.1109/PerComWorkshops51409.2021.9431011. [Online]. Available: https://ieeexplore.ieee.org/document/9431011/.

[35] Y. Luo, X. Liu and J. Xiu, 'Energy-efficient Clustering to Address Data Heterogeneity in Federated Learning,' in *ICC 2021 - IEEE International Conference on Communications*, ISSN: 1938-1883, Jun. 2021, pp. 1–6. DOI: 10.1109/ICC42927.2021.9500901.

[36] H. Cho, A. Mathur and F. Kawsar, 'Device or User: Rethinking Federated Learning in Personal-Scale Multi-Device Environments,' en, in *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, Coimbra Portugal: ACM, Nov. 2021, pp. 446–452, ISBN: 978-1-4503-9097-2. DOI: 10.1145/3485730.3493449. [Online]. Available: https://dl.acm.org/doi/10.1145/3485730.3493449.

[37] P. Wu, T. Imbiriba, J. Park, S. Kim and P. Closas, 'Personalized Federated Learning over non-IID Data for Indoor Localization,' in *2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, ISSN: 1948-3252, Sep. 2021, pp. 421–425. DOI: 10.1109/SPAWC51858.2021.9593115.

[38] M. Mestoukirdi, M. Zecchin, D. Gesbert, Q. Li and N. Gresset, *User-Centric Federated Learning*, en, Number: arXiv:2110.09869 arXiv:2110.09869 [cs, eess, math], Oct. 2021. [Online]. Available: http://arxiv.org/abs/2110.09869.

[39] V. Kelli, V. Argyriou, T. Lagkas, G. Fragulis, E. Grigoriou and P. Sarigiannidis, 'IDS for Industrial Applications: A Federated Learning Approach with Active Personalization,' en, *Sensors*, vol. 21, no. 20, p. 6743, Jan. 2021, Number: 20 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 1424-8220. DOI: 10.3390/s21206743. [Online]. Available: https://www.mdpi.com/1424-8220/21/20/6743.

[40] K. Kirsten, B. Pfitzner, L. Löper and B. Arnrich, 'Sensor-Based Obsessive-Compulsive Disorder Detection With Personalised Federated Learning,' in *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2021, pp. 333–339. DOI: 10.1109/ICMLA52953.2021.00058.

[41] Q. Wu, X. Chen, Z. Zhou and J. Zhang, 'FedHome: Cloud-Edge based Personalized Federated Learning for In-Home Health Monitoring,' *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020, Conference Name: IEEE Transactions on Mobile Computing, ISSN: 1558-0660. DOI: 10.1109/TMC.2020.3045266.

[42] M. J. Page, D. Moher, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan, R. Chou, J. Glanville, J. M. Grimshaw, A. Hróbjartsson, M. M. Lalu, T. Li, E. W. Loder, E. Mayo-Wilson, S. McDonald, L. A. McGuinness, L. A. Stewart, J. Thomas, A. C. Tricco, V. A. Welch, P. Whiting and J. E. McKenzie, 'PRISMA 2020 explanation and elaboration: Updated guidance and exemplars for reporting systematic reviews,' en, *BMJ*, n160, Mar. 2021, ISSN: 1756-1833. DOI:

10.1136/bmj.n160. [Online]. Available: https://www.bmj.com/lookup/doi/10.1136/bmj.n160.

[43] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan, R. Chou, J. Glanville, J. M. Grimshaw, A. Hróbjartsson, M. M. Lalu, T. Li, E. W. Loder, E. Mayo-Wilson, S. McDonald, L. A. McGuinness, L. A. Stewart, J. Thomas, A. C. Tricco, V. A. Welch, P. Whiting and D. Moher, 'The PRISMA 2020 statement: An updated guideline for reporting systematic reviews,' en, *BMJ*, n71, Mar. 2021, ISSN: 1756-1833. DOI: 10.1136/bmj.n71. [Online]. Available: https://www.bmj.com/lookup/doi/10.1136/bmj.n71.

Appendix A

Link to code

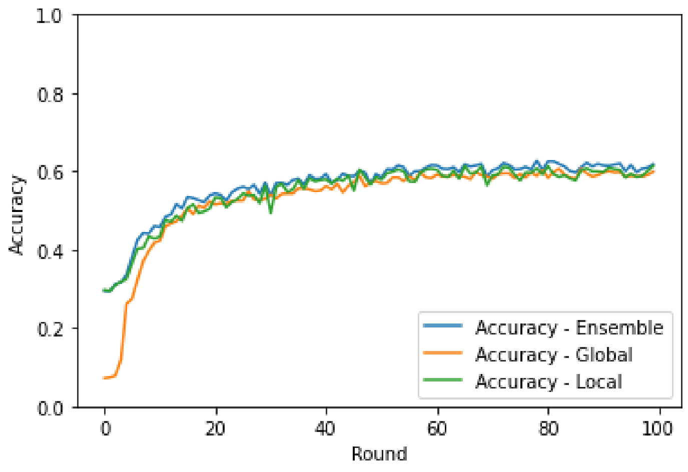
The repository for the code can be found here:
`https://www.github.com/erikhny/FLCU`

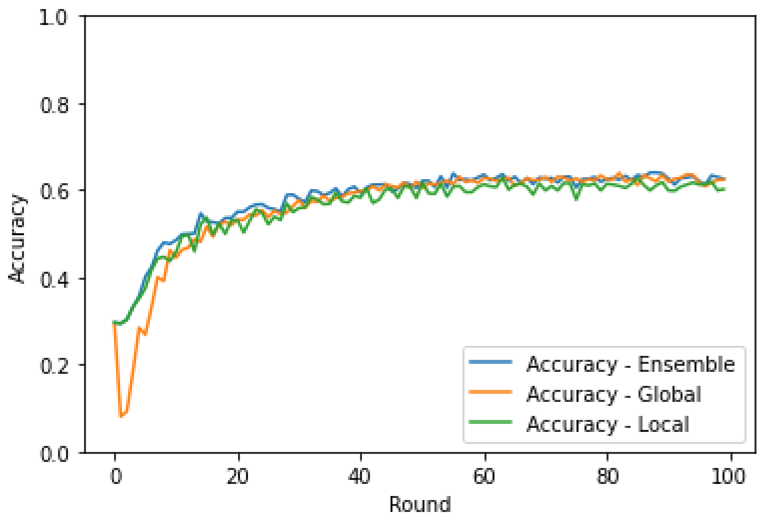
Appendix B

Additional figures

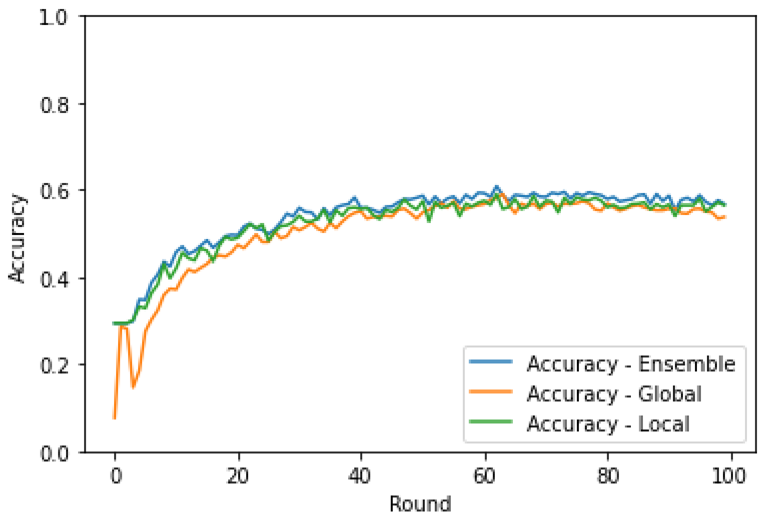
B.1 Multi-model figures

B.1.1 CIFAR-10

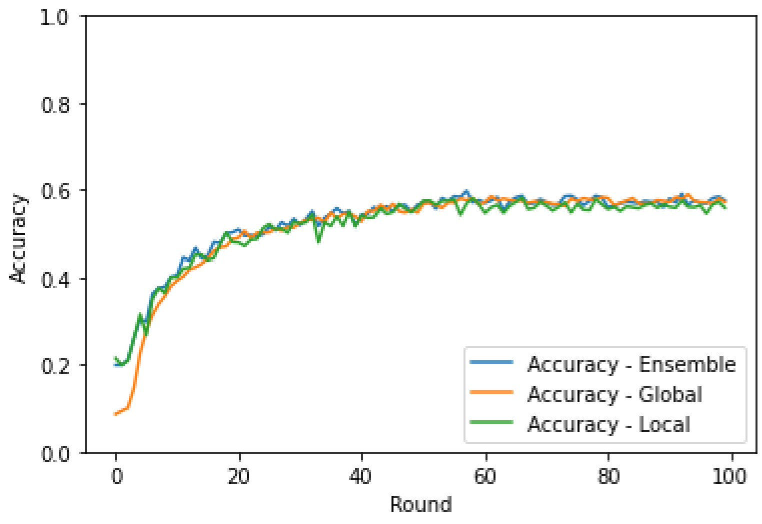


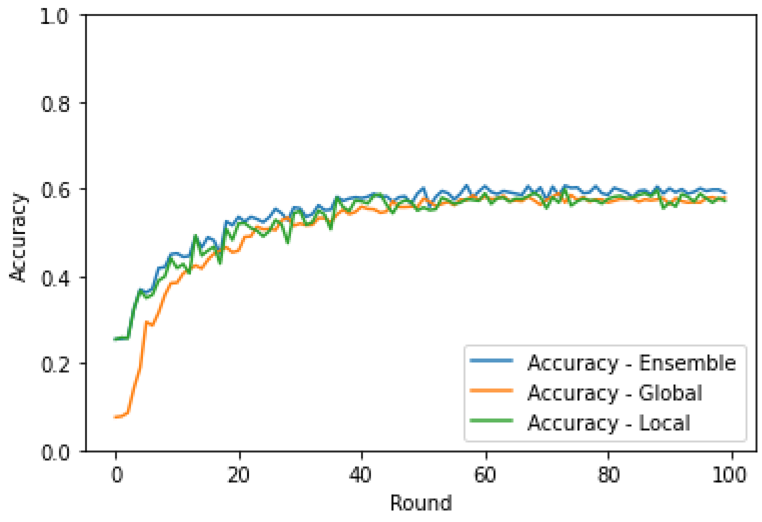


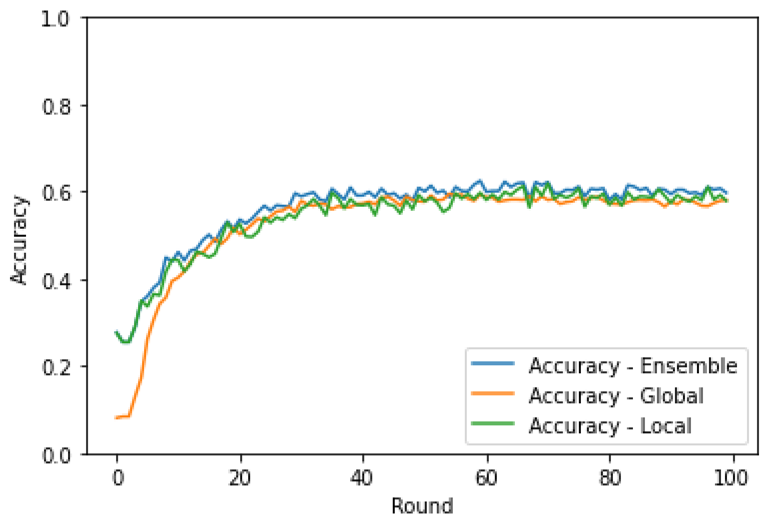
Accuracy
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local

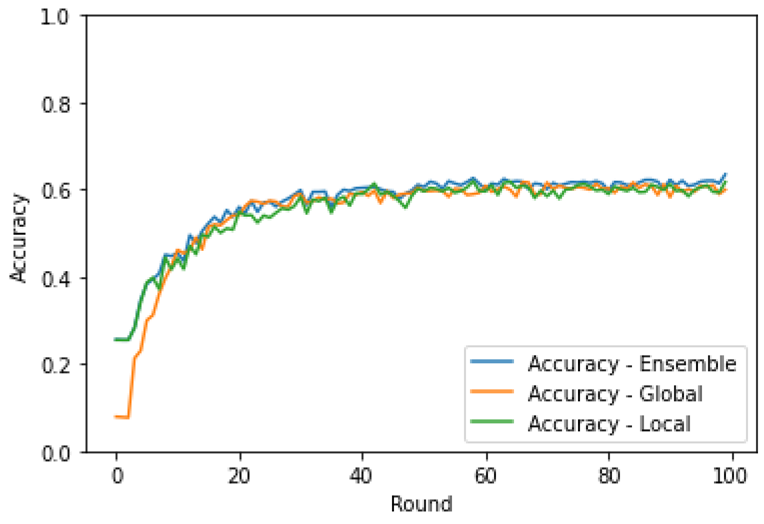


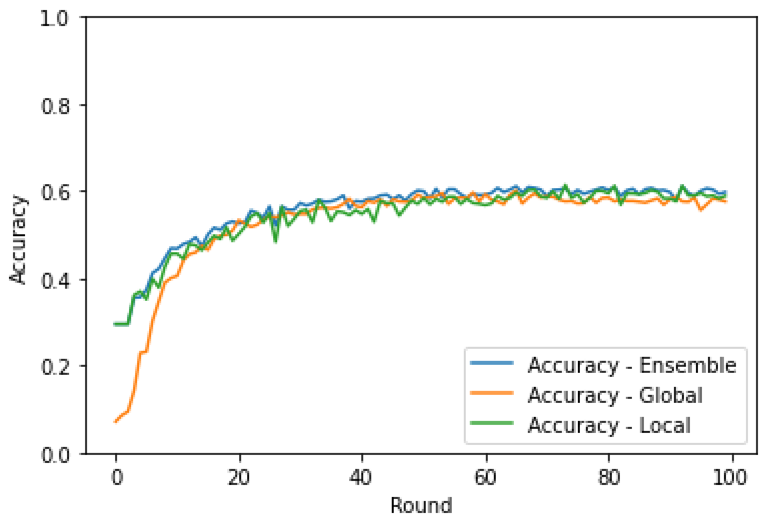
Accuracy
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local

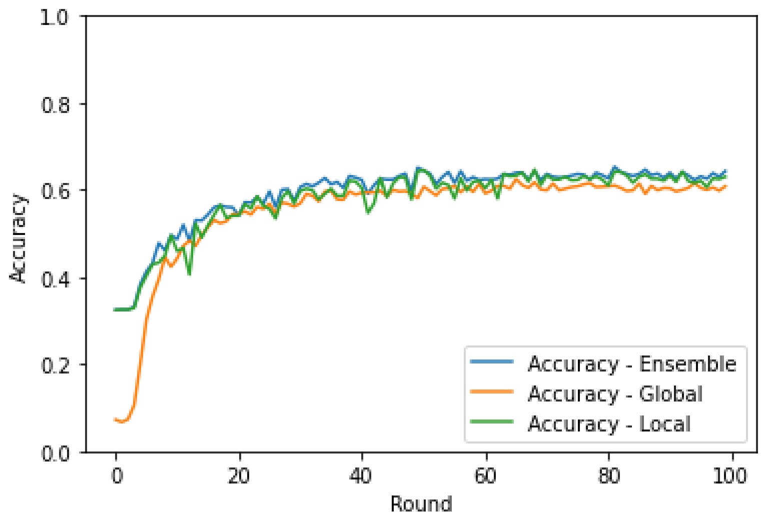

1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local

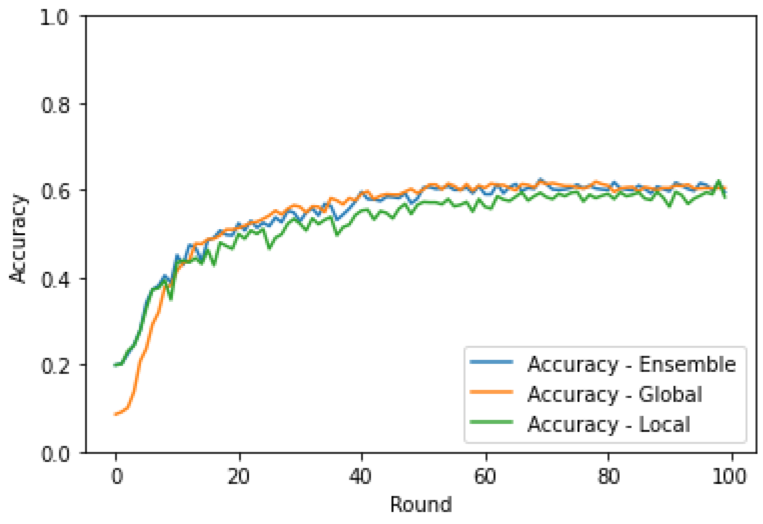

1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local


Accuracy
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local

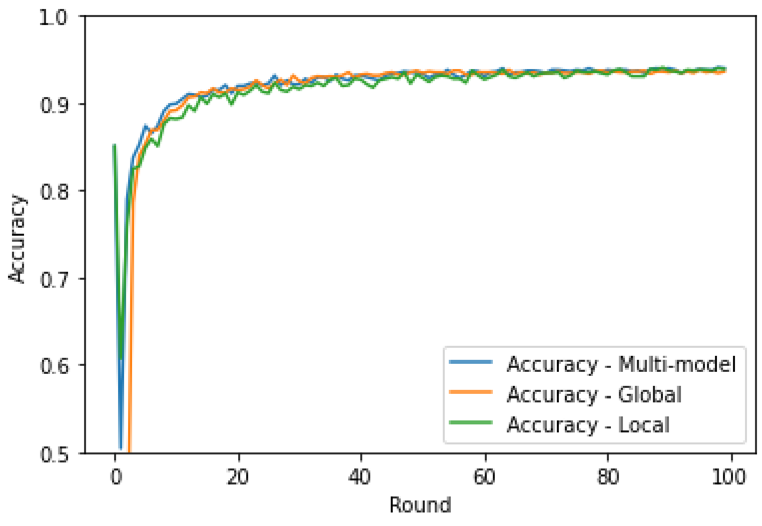

Accuracy
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local

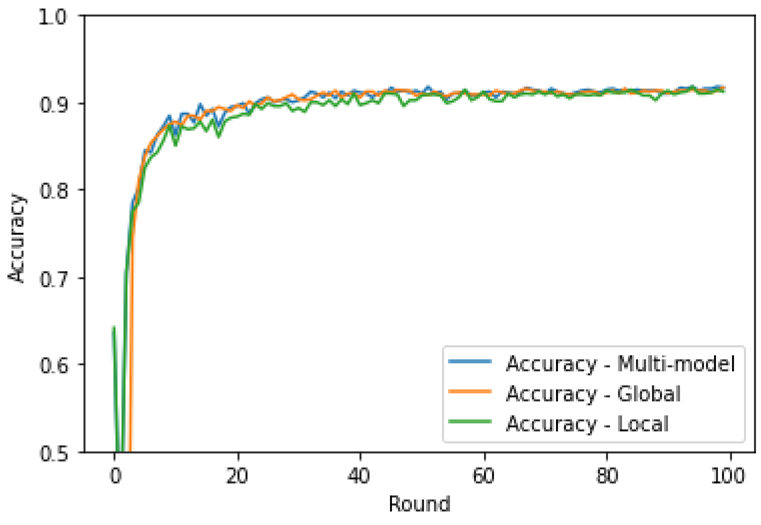

1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local


Accuracy
Round
Accuracy - Ensemble
Accuracy - Global
Accuracy - Local
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100

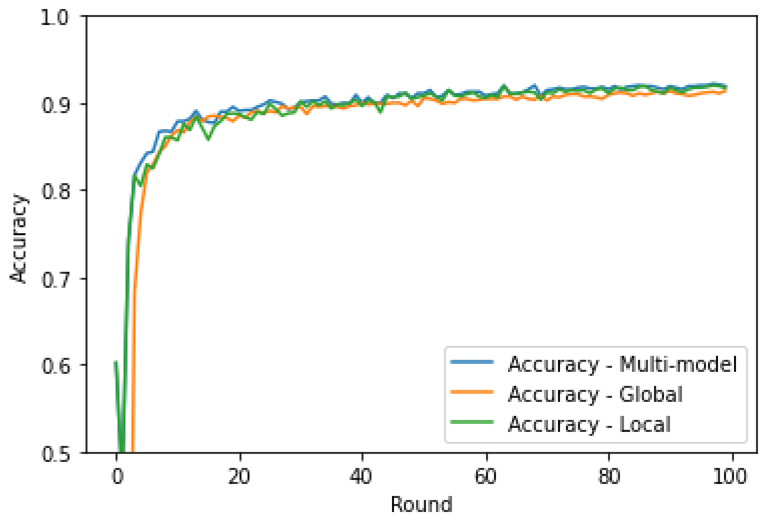
B.1.2 MNIST

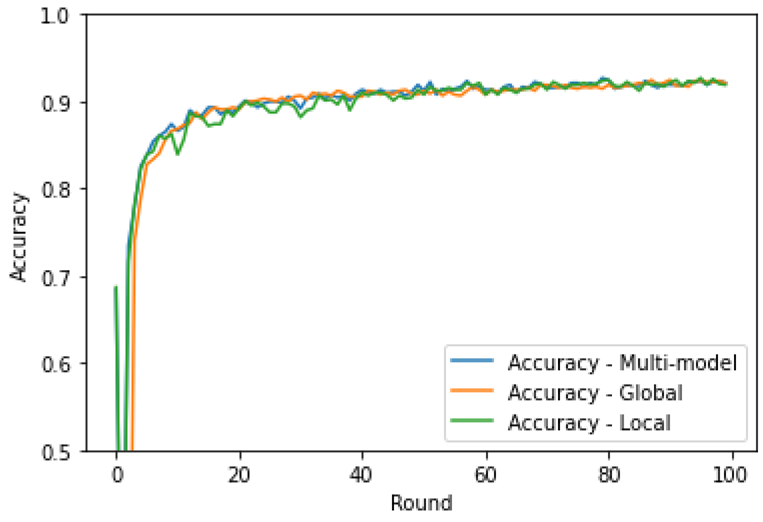


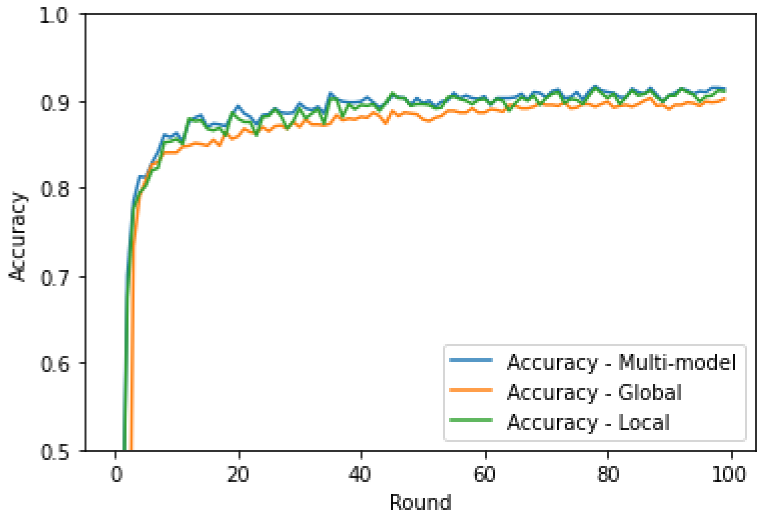
1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local

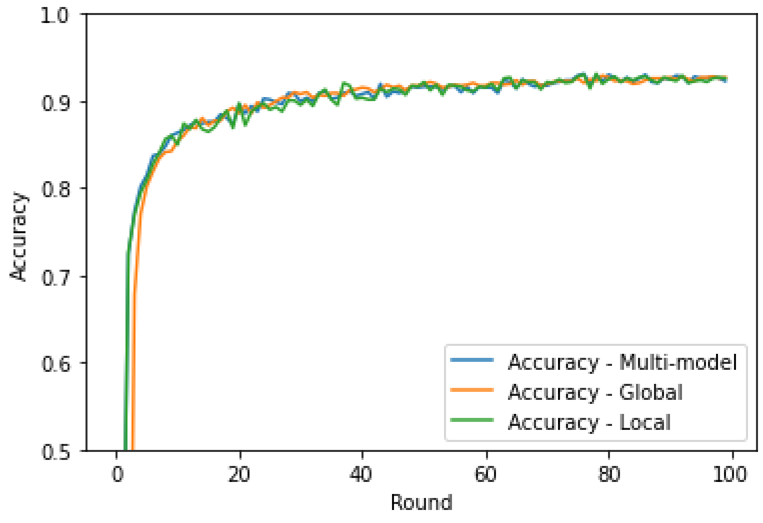


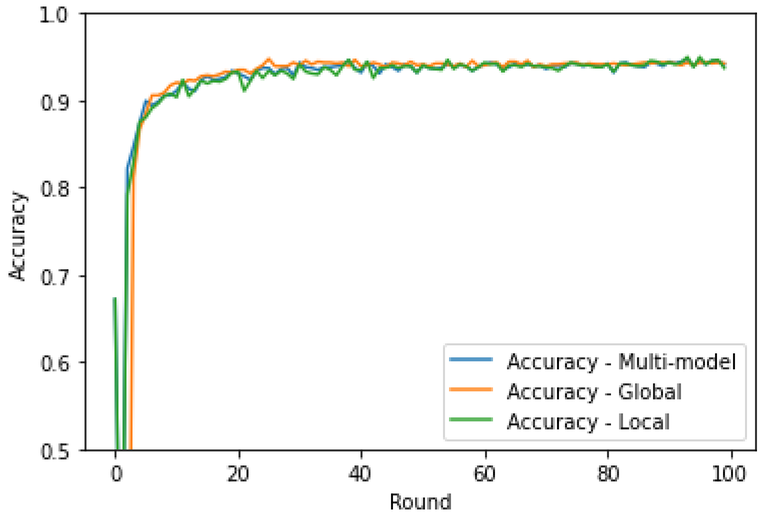
1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local

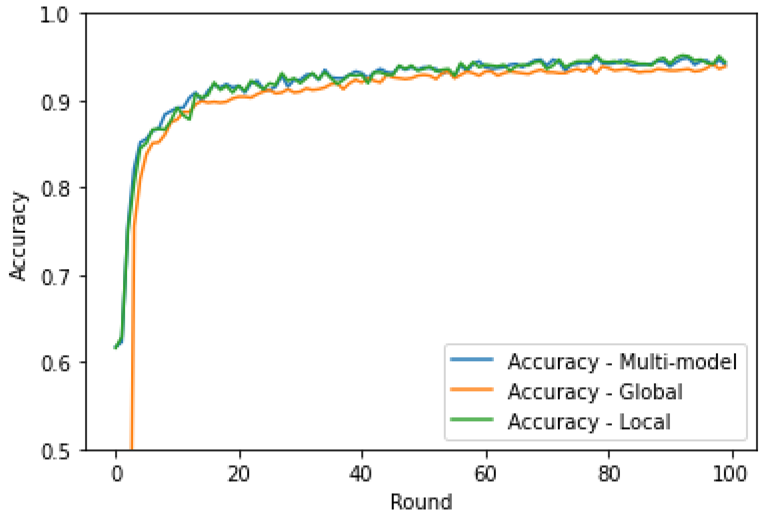

1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local

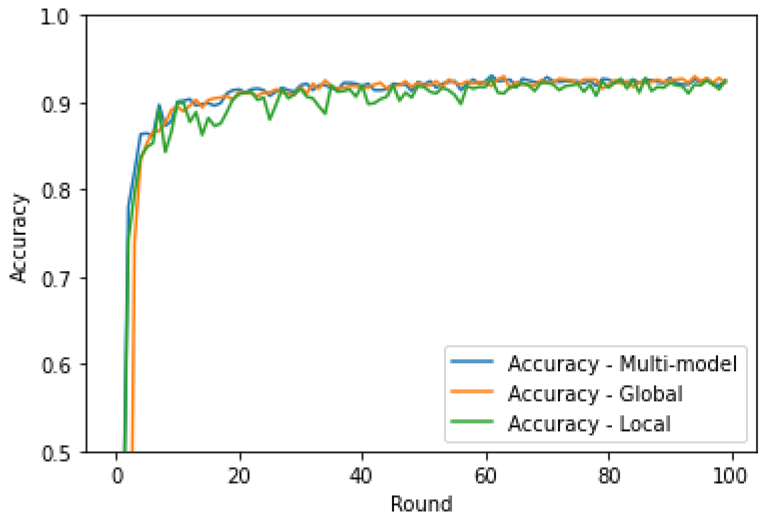

1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local


1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local

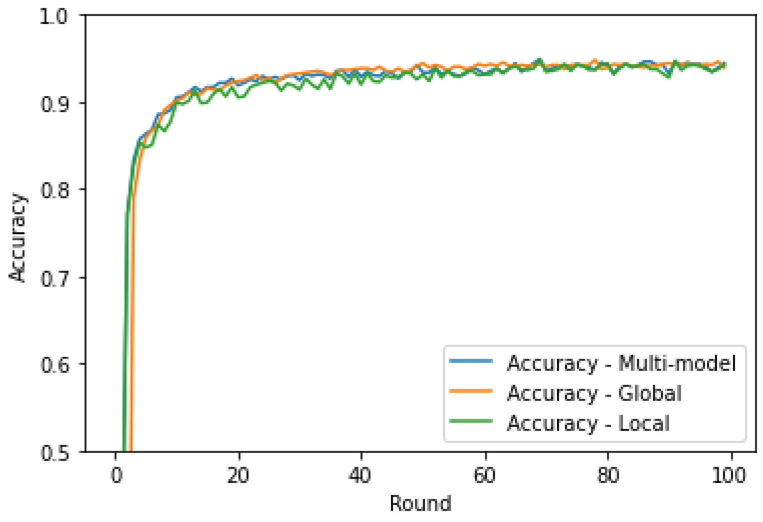

1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local


Accuracy
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local


Accuracy
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local



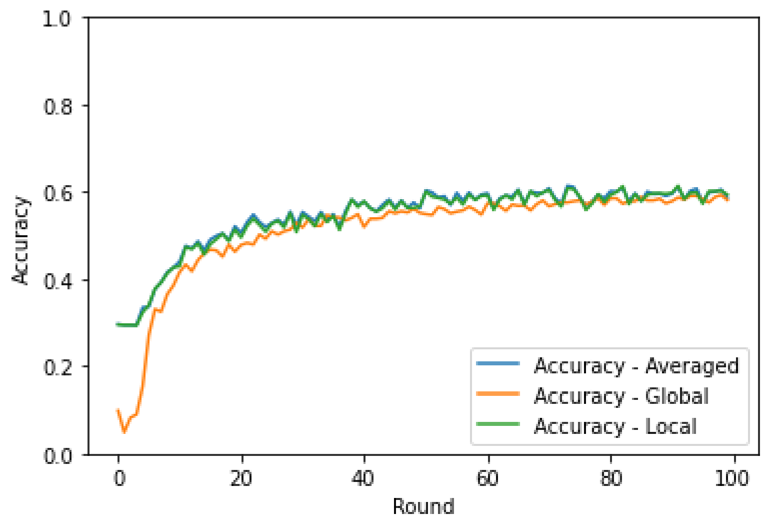
1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local

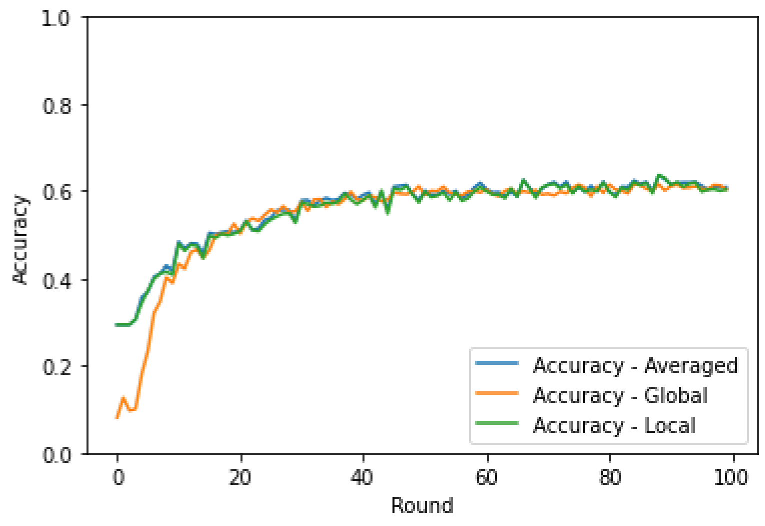


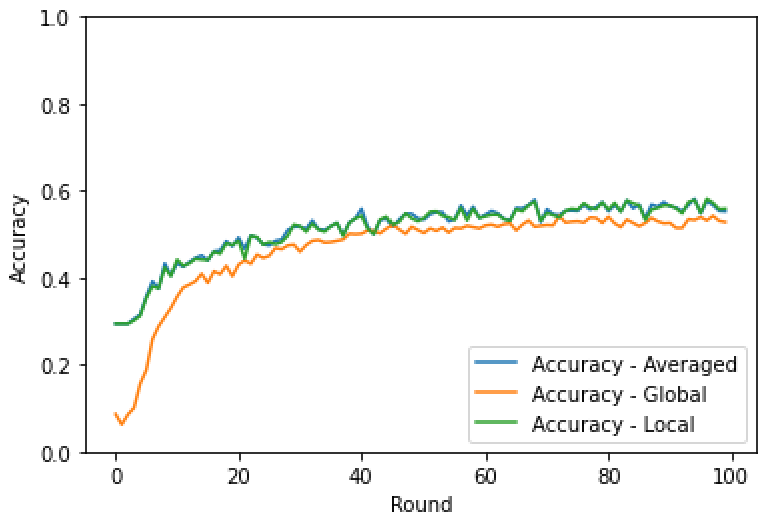
1.0
0.9
0.8
0.7
0.6
0.5
Accuracy
0
20
40
60
80
100
Round
Accuracy - Multi-model
Accuracy - Global
Accuracy - Local

B.2 Weighted averaging

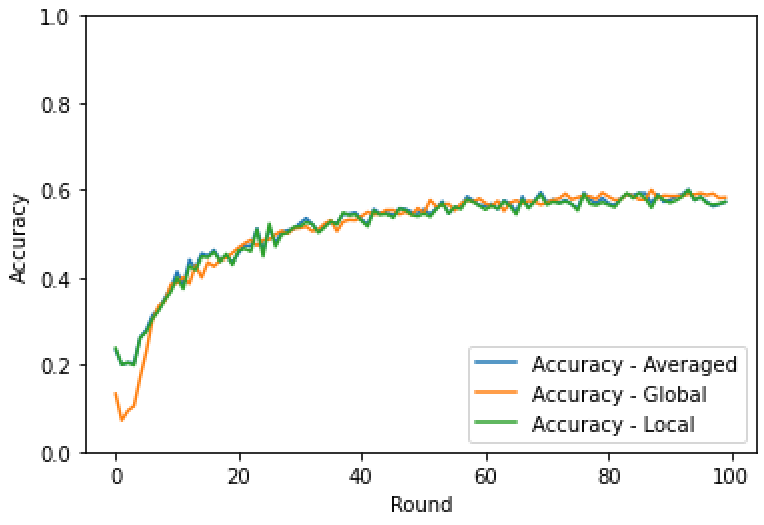
B.2.1 CIFAR-10



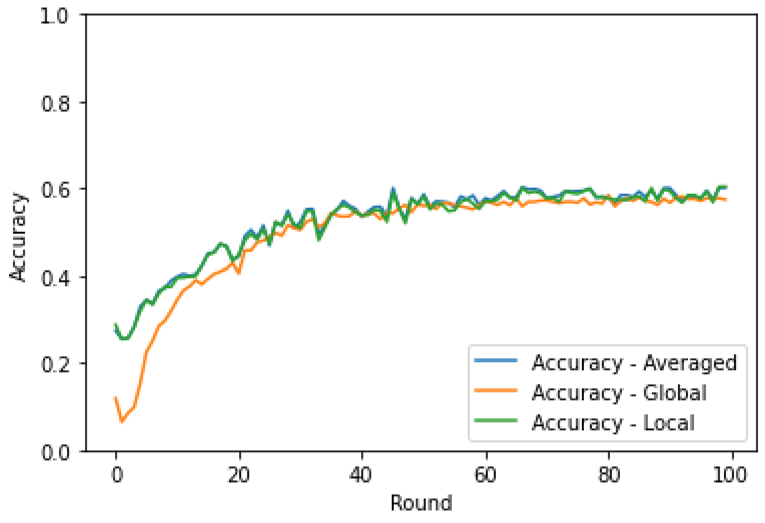


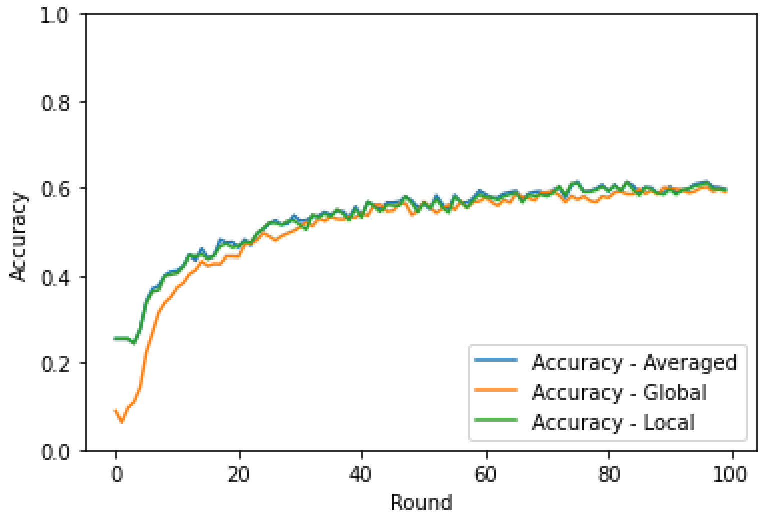


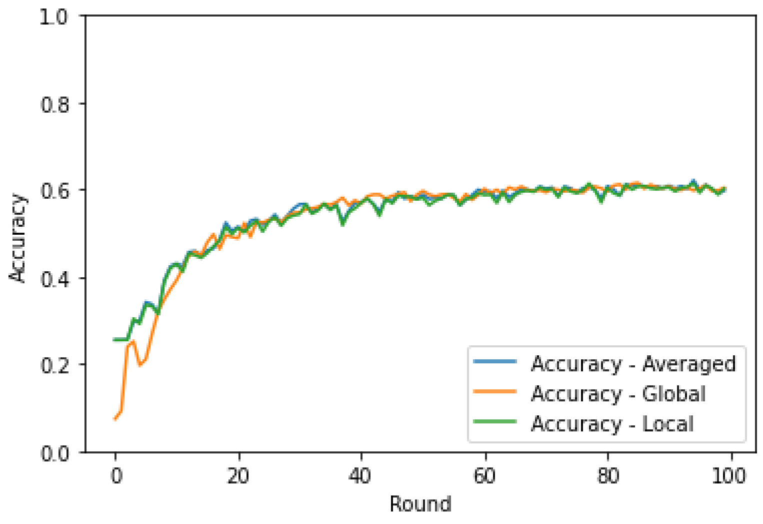
1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

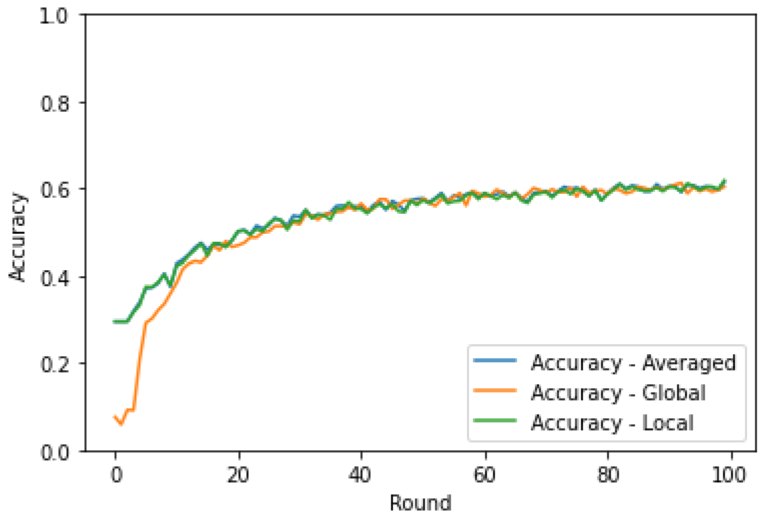


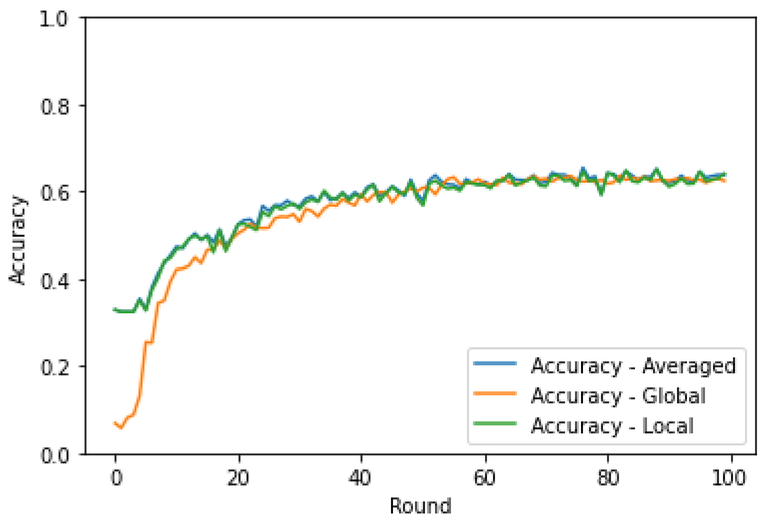
1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

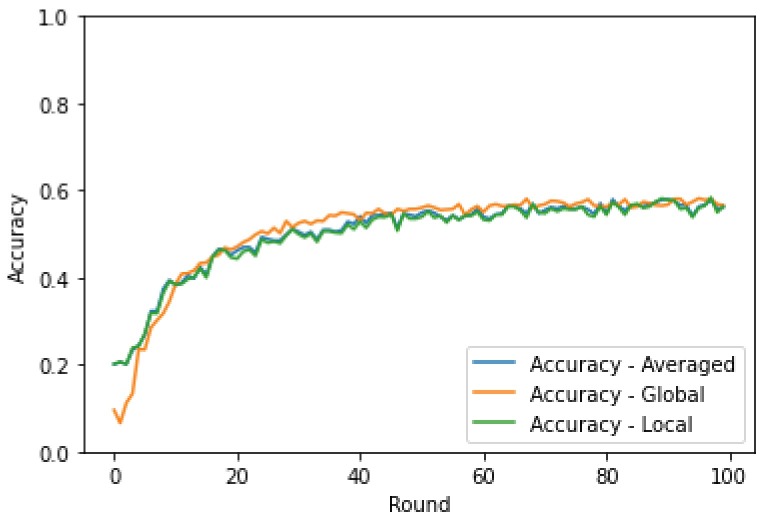

1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

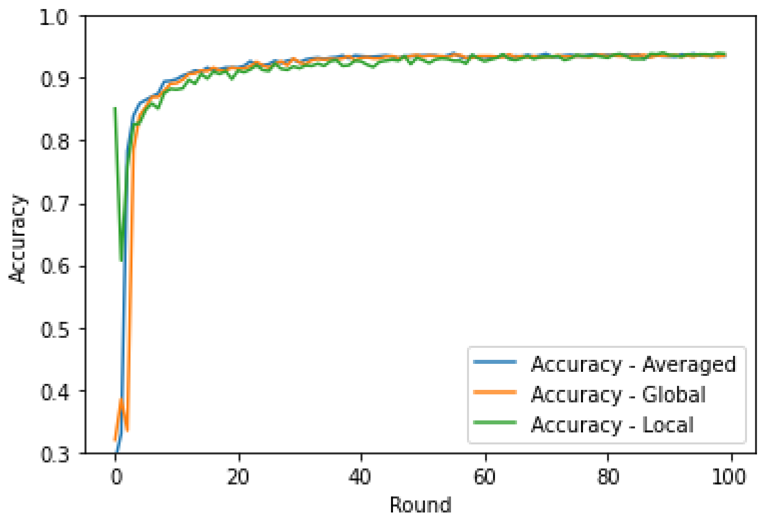

1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

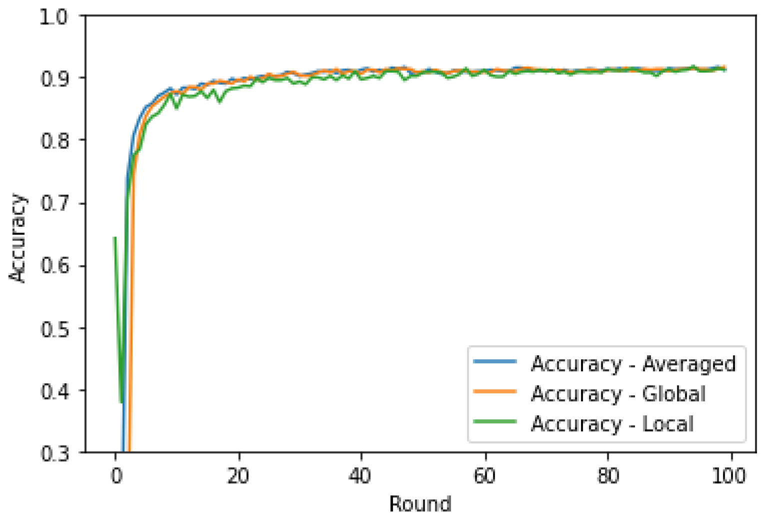

1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

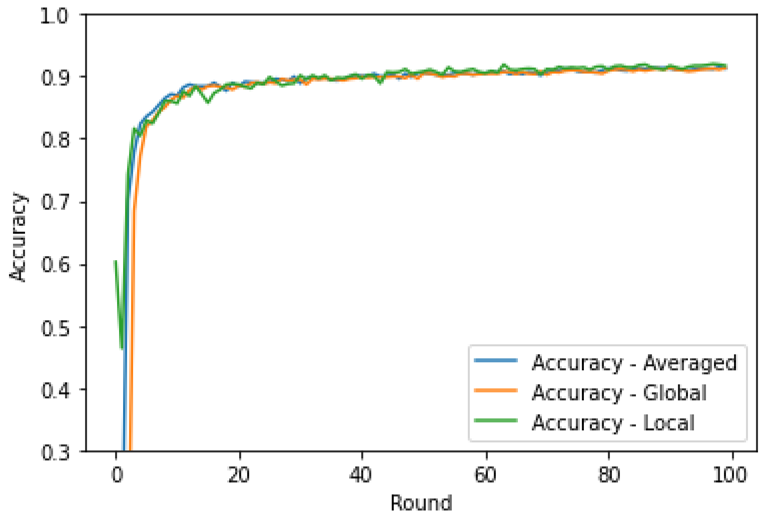

Accuracy
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

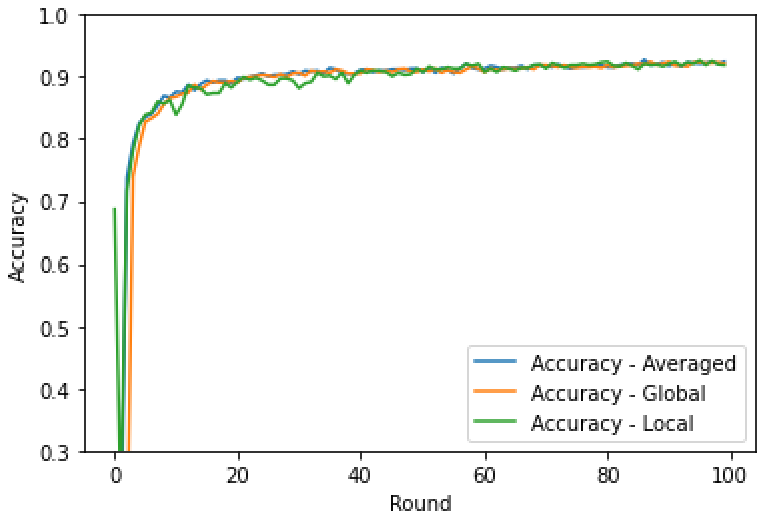

Accuracy
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

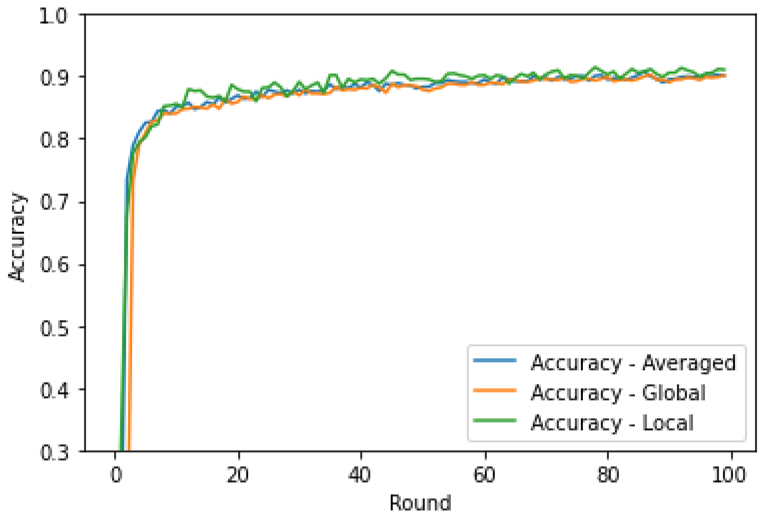
B.2.2 MNIST

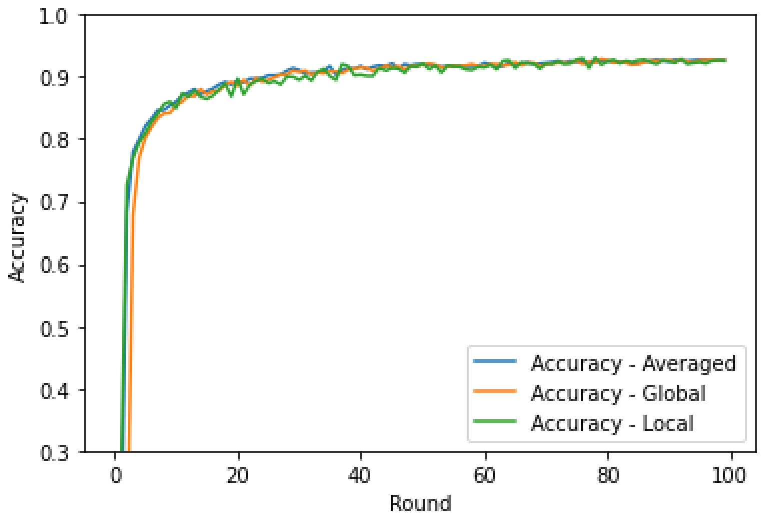

1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

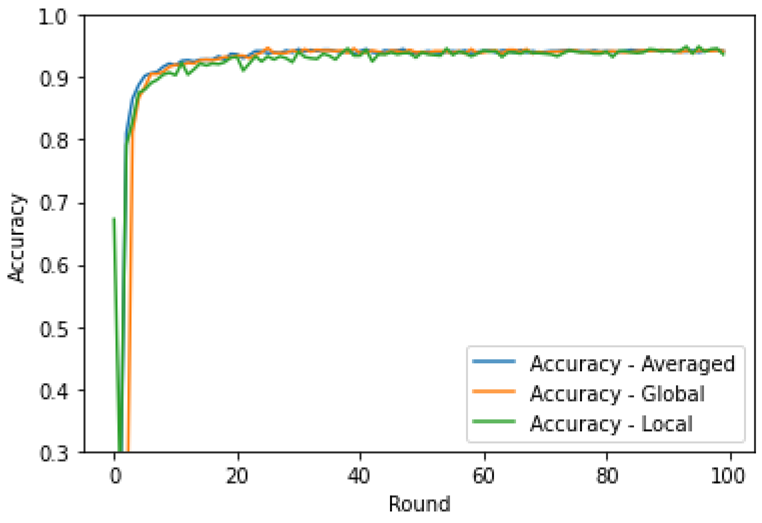

1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

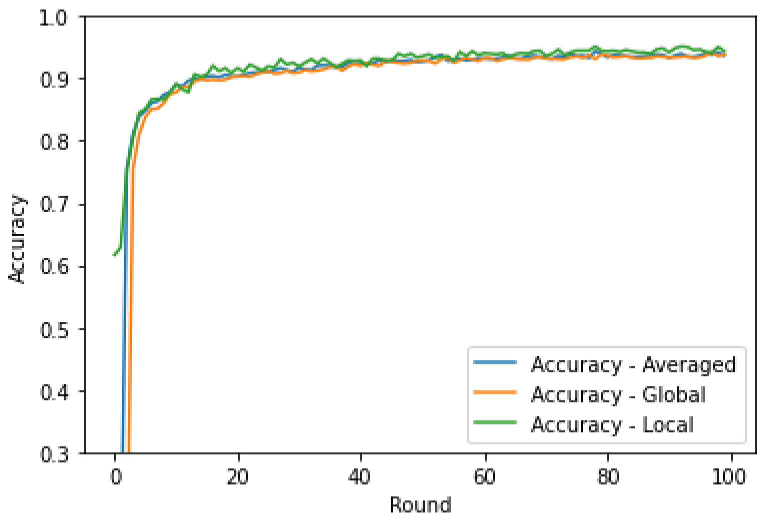

Accuracy
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

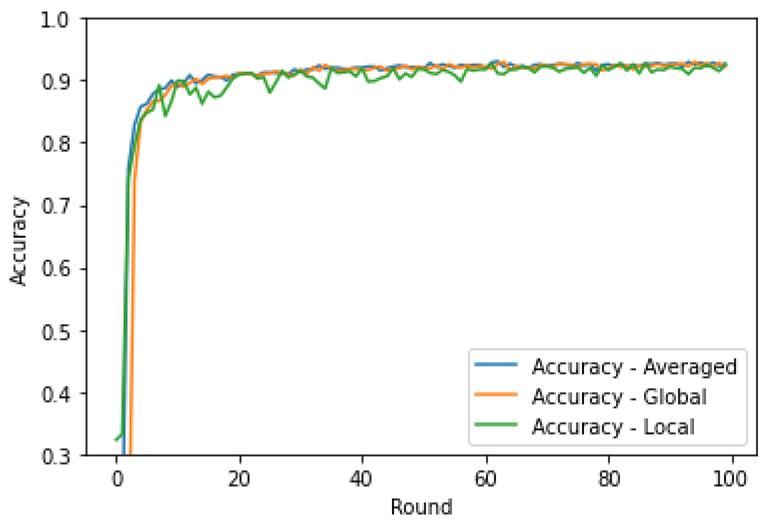

Accuracy
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local

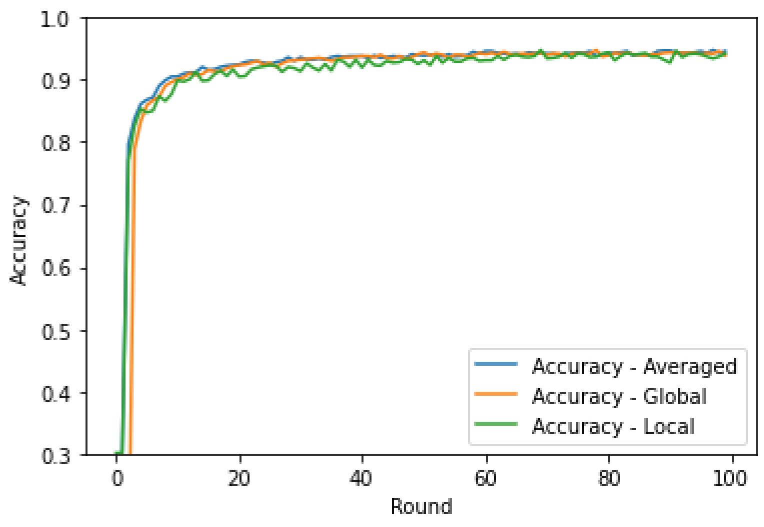

1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local


Accuracy
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local


Accuracy
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
20
40
60
80
100
Round
Accuracy - Averaged
Accuracy - Global
Accuracy - Local