Federated Learning for Predicting the Next Node in Action Flows

(extended abstract of the MSc dissertation)

Daniel Francisco Lopes
Departamento de Engenharia Informática
Instituto Superior Técnico

Advisors: Professor Luís Rodrigues and João Nadkarni

Abstract

Federated learning is a machine learning approach that allows different clients to collaboratively train a common model without sharing their data sets. We focus on centralized federated learning, where a central server collects contributions from individual clients, merges these contributions, and disseminates the results to all clients. Since clients have different data and classify data differently, there is a trade-off between the generality of the common model and the personalization of the classification results. Current approaches rely on using a combination of a global model, common to all clients, and multiple local models, that support personalization. In this work, we report the results of a study, where we have applied some of these approaches to a concrete use case, namely the *Service Studio* platform from OUTSYSTEMS, where Graph Neural Networks help programmers in the development of applications. Furthermore, we explore two different approaches which merge some of the state-of-the-art algorithms so as to develop the best model for all the different clients. Our results show that one of the proposed approaches manages to achieve similar performance to the best-performing algorithms for all the classes of clients and can even outperform previous algorithms for some classes of clients.

1 Introduction

Machine Learning (ML) is an area of Artificial Intelligence (AI) that studies how to build a model, from a given training data set, such that it can be used to predict an output given an input. Federated Learning (FL) is a particular case of ML where different entities collaborate to construct a common model without explicitly exchanging their data sets and compromising performance while, ideally, preserving the privacy of their data. Our research is driven by the requirements of OUTSYSTEMS, where FL is being explored as an alternative to the current fully centralized inference and training setup, in order to build a model intended to help programmers in their coding tasks.

In our work, we study the centralized FL approach, which uses a central server to keep a global model. The server periodically performs communication rounds with some clients (all or just a subset), to improve the global model with the help of the individual training data from each client. In each communication round, the selected clients receive the global model parameters from the central server (step 1), train this model with their private data (step 2), and send back to the server the resulting updates to the model (step 3). The server then aggregates all the received local updates to generate a global update (step 4) to improve the global model. This procedure is repeated over various communication rounds.

FL has many challenges. First, the communication rounds may consume significant processing and network resources and should be made as efficient as possible. Second, keeping data at the clients may not be enough to preserve privacy, as it may be possible to infer the content of the training data from the updates to the model. Third, a faulty or malicious client may attempt to bias or poison the global model. Lastly, clients may have different data and different classification preferences, which creates the need for maintaining personalized models, in combination with a common general model.

In this work, we are mainly concerned with the last challenge, particularly, in techniques that can offer clients personalized models, while still benefiting from FL. Current approaches for personalization rely on using a global model, common to all clients, which is then adapted to each client's data, or on splitting the model in two and having a shared global part and multiple more specific parts, each one tailored and maintained exclusively by each client. We survey the state-of-the-art solutions for FL and identify unexplored alternatives for training in this personalized setup that are worth exploring. Based on these findings, we propose to implement and evaluate some of the existent solutions and our two new variants, *FedHybridAvgLG* and *FedHybridAvgLGDual*.

We experiment and evaluate these new variants in the context of the *Service Studio* platform from OUTSYSTEMS. *Service Studio* is a low-code platform that allows users to design and manage systems and applications in a simple and efficient manner through a visual and interactive user interface. In this platform, among other things, the user defines the application logic by creating a flow of actions. These actions can be of several types, for instance, "`if`", "`for`" or "`assign`" (many other actions related to, for example, user interface development and data management, are possible). In this platform, ML is used to give recommendations to the users about which actions should be added next to an action flow.

An action flow can be modelled as a graph where the actions are nodes and the edges represent the flow from action to action. The graph can then be used as input to a specific type

of ML neural network model architecture, a Graph Neural Network (GNN), which is specialized in interpreting graphs and making predictions on them. In our case, the model predicts, from a finite set of possible node types, which are the most probable to be added next to the graph. This prediction is then used by *Service Studio* to recommend possible next actions to the user. The use of FL in this context is relevant because it allows the model to be trained using contributions from various clients while ensuring that information about the applications being developed remains private.

Our results show that one of the newly proposed approaches achieves a performance similar to the top-performing algorithms, for each class of clients.

2 Studied Algorithms

2.1 Federated Learning

In its simplest form, the creation of ML models assumes that all the data of the clients is shared during the training phase. FL allows different clients to collaborate so as to construct a shared model without the need to share their private data, therefore preserving data privacy.

The most common approach to achieve FL consists in using a central server to orchestrate the coordination among clients. This architecture is described by Bonawitz et al. [1]. The protocol proceeds in rounds of communication where, in each round, the server selects a set of clients to participate. When the round starts, the server sends the parameters of the current global model to each participant. Afterwards, each participant independently trains the model received, using its own data set, obtaining a local model. The client then sends an update back to the server which reflects the changes that have been locally applied to the global model. Finally, the central server collects the updates from different clients, performs a weighted aggregation considering the size of each client's training data set, and uses the resulting global update to derive the new global model to be used in the following round. This aggregation method is defined as Federated Averaging (FedAvg) [7].

It is possible to define different FL categories, according to the way the data is partitioned, the way clients communicate and the scale of federation [3, 4, 10]. In terms of the data partitioning, if all the clients store the same features about each data subject (in our case action flows) but each data subject has all its features stored in a single client, the partition is *horizontal*; if the information about a single data subject is spread across clients and each client stores a different feature about the data subject, the partition is *vertical*; if each client stores different features of different data subjects, the partition is *hybrid*. In terms of the communication architecture, it can be classified as centralized (the previously described case), or decentralized (if the clients directly communicate with each other without needing a central server). Finally, in terms of the scale of federation it can be classified as "cross-silos" or "cross-device", depending on whether the clients represent organizations or individual devices. In this work, we consider a horizontally partitioned environment, with centralized architecture and "cross-silo" federation, where the clients are the organizations using the *Service Studio* platform.

2.2 Personalized Federated Learning

FL introduces several challenges in different areas, including the privacy of client data, robustness against attacks during the model training phase, communication efficiency and model performance. In this thesis, we focus on the challenge of maximizing the performance of the model. This typically involves using mechanisms of model personalization [9]. Namely, the necessity of creating specific models for each client, which may include global components (which benefit from the contribution of all the clients) and specialized components (adjusted to the data set of each client).

Concretely, we focus on algorithms based on the parameter decoupling technique, which divide the model into two parts, the representation or "body" and the classifier or "head". The body is composed of the first layers of the model and is responsible for extracting the data features, and the head is composed of the last layers of the model and is responsible for classifying the data from its features. Depending on the algorithm one of the parts is global and the other is local and specialized to each client. Clients can train both parts but the updates exchanged with the server correspond only to the global part. Three relevant examples of these approaches are Local Global Federated Averaging (LG-FedAvg) [5], Federated Representation Learning (FedRep) [2] and Federated Averaging with Body Aggregation and Body Update (FedBABU) [8]. Figure 1 summarizes the algorithms and includes the FedAvg algorithm (Figure 1(a)), covered in the previous section. The joint chains indicate the training is joint, that is, both the head and the body are updated in the same training step. We provide in the next paragraphs a brief description of the remaining three algorithms.

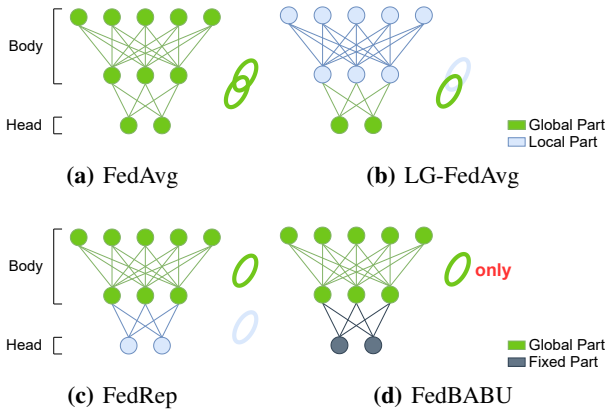


Figure 1. Models according to the algorithms FedAvg, LG-FedAvg, FedRep and FedBABU.

LG-FedAvg is one of the algorithms based on the parameter decoupling technique. In the specific case of this algorithm,

the classifier is shared with the server and the representation is specialized for each client. Thus, this algorithm personalizes the body such that it can extract the features of the data for each client and shares the head in order to obtain a classifier that works for every client. This allows each client to have its own type of data, for instance, one client can have images while the other can have text. Therefore, after receiving the head of the model from the server, the client associates its local body to obtain the local model which is, afterwards, trained jointly, that is, performing a sequence of local epochs and updating both the head and the body simultaneously. After training, the client sends to the server the updates referring to the head. Figure 1(b) illustrates the training procedure and the role of each model part.

FedRep is an algorithm which takes a different approach to the LG-FedAvg algorithm. The authors of the algorithm argue that results from centralized ML indicate that data shares a common representation of its features and that the heterogeneity resides in the classifications. For example, an image of a dog is represented equally in two clients, however, one client can classify the dog as ugly while another can classify it as beautiful. Therefore, the representation is shared with the server and the classifier is specialized for each client. By sharing the body, the algorithm tries to obtain a global representation for all the clients while keeping the head local allows for the classifications to be specialized. Another difference between FedRep and LG-FedAvg resides in the way the training is performed. While in LG-FedAvg the body and the head are updated simultaneously and for the same number of rounds, in FedRep the head is fully trained first and only afterwards is the body trained, furthermore, the number of training rounds between the head and the body may differ. Therefore, FedRep is more flexible since it allows the parts of the model to be trained for a different number of rounds, which can be useful when we want to personalize the head further by performing more training rounds than the body. Figure 1(c) illustrates the training procedure of the algorithm FedRep.

FedBABU is another algorithm based on the parameter decoupling technique. Figure 1(d) summarizes the algorithm. Similarly to FedRep, FedBABU shares the body, such that, a good representation of the data is collaboratively created by the clients. The authors studied the FedAvg algorithm to understand why an increase in the performance of the global model does not necessarily mean that fine-tuning it further increases the performance. They came to the conclusion that aggregating the head introduces unnecessary noise to the global model, as the classification is a specificity of each client. Therefore, FedBABU leverages a shared fixed global head to train the body in each client, focusing on creating a good generalized global representation. Then, and only during evaluation, the head is personalized through fine-tuning.

Table 1 summarizes the main characteristics of the three personalization algorithms. In terms of notation, E indicates the number of local training rounds, E_H the number of local head training rounds and E_B the number of local body training rounds.

Table 1. Comparison between the different FL personalization algorithms.

Algorithm	Exchanged Part	Custom Part	SGD steps per Client	Local Training Procedure	Fine-Tuning (FT Part)
LG-FedAvg	head	body	E	full model	optional after training (full model)
FedRep	body	head	$E_H + E_B$	head first (w/ global body) body last (w/ trained head)	optional after training (full model)
FedBABU	body	head	E	body only (w/ fixed head)	required after training (head or full)

3 Federated Learning in the context of OUTSYSTEMS

3.1 *Service Studio*

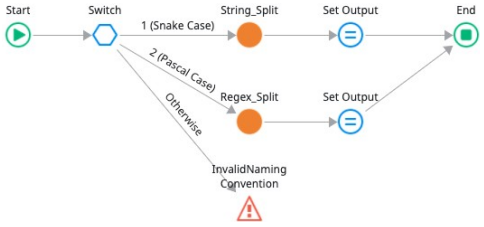


Figure 2. *Service Studio* Action Flow for splitting a string into multiple tokens from a given naming convention.

Service Studio is a platform developed by OUTSYSTEMS, which intends to help programmers develop their applications in a simple way, without the need to write code. As such, the programmers simply need to create a chain of actions, called an action flow, which represents the logic of the application. As an example, Figure 2 shows an OUTSYSTEMS Action Flow for splitting a string formatted in a given naming convention. The flow leverages a "switch" action to select the initial string naming convention format, either snake case (condition 1) or pascal case (condition 2), otherwise, it raises an exception. For the pascal case, first a "server action" is performed to split the string by capital letters and then the output is set. For example, for the input string "FederatedLearning" in the Pascal naming convention, this flow outputs "Federated Learning".

Action Flows can be modelled as graphs and can be classified as directed weakly connected graphs. The nodes represent the actions and are connected through edges, which represent the flow between two actions. Each edge is directed, meaning that there exists a flow relation between a source node and a destination node. Each node has its own attributes which represent characteristics or features of the action. For example, all nodes have a kind that indicates the type of the

Table 2. Comparison between models for different clients.

	Number of Action Flows	Accuracy (%) Local Model	Accuracy (%) Centralized Model
Client A	47,711	75.41	65.79
Client B	60	24.14	58.62

action, e.g., "switch", "assign", "if", and so on. Edges also have attributes that represent the characteristics of the flow, for example for a "switch" action one of the edge attribute indicates the condition the edge corresponds to.

ML is used in this platform to give recommendations to the users whenever the user tries to add a new action by suggesting some possible next actions to be added to the action flow. These suggestions are obtained using a ML model based on GNNs. In particular, the model's objective is to predict one of the nodes' attributes: the node "kind".

Currently, this model is trained in a centralized fashion, that is, the OUTSYSTEMS clients need to share their data with a centralized server, such that, a model can be trained on the data from all the clients. However, this approach has two setbacks. Firstly, it requires the clients to share their personal data which may contain sensitive information, thus raising privacy concerns. Secondly, the obtained model is shared across all the clients, which means the predictions might not be the most adequate as there is no personalization. Another more naïve approach would be having each OUTSYSTEMS client develop their own local model, trained only with each client's data. However, this would require each client to have enough data to be able to train its own model, which is not always the case.

Table 2 highlights the advantages and disadvantages, in terms of the quality of the recommendations, for the usage of local models in relation to the usage of a single global model, calculated from the data of 881 clients, resorting to two distinct clients. Client A has a long usage history of the platform, therefore, it already has a large data set. As such, it has enough data points to construct a local model which offers great accuracy and is specialized to its business model. On the other hand, client B is relatively new to the platform, thus it has a small data set. This client clearly benefits from the usage of a centralized model.

One way to allow the creation of collaborative models without the need to share private data is by using FL. Furthermore, as seen in Section 2.2, there are approaches which focus on personalizing FL models to each client. This allows the creation of models which are trained with the data of various clients without sharing client data, while also being personalized to each client's data. In this work, we propose two possible approaches, each combining two different algorithms from the ones covered previously, we deem these algorithms hybrid algorithms (the reasoning behind the chosen algorithms for the proposed approaches will become clearer in Section 4.2).

3.2 FedHybridAvgLG

The algorithm ***Fed****erated* ***Hybrid*** *Fed****Avg LG****-FedAvg (FedHybridAvgLG)* is an hybrid algorithm which attempts to merge the FedAvg and LG-FedAvg algorithms. Smaller clients, who do not have sufficient data for personalization, leverage the FedAvg algorithm as it generates more general models since the full model is shared by all clients. Larger clients, which are those who have enough data for personalization, leverage the LG-FedAvg algorithm as it allows personalization by specializing the body of the client.

3.2.1 Small Clients. For smaller clients, which do not have enough data to personalize the model, the algorithm *FedHybridAvgLG* works as the FedAvg algorithm, hence Figure 1(a) can be used to illustrate the scheme of a model for a small client. Thus, in each communication round, after receiving the model parameters, the smaller clients train the body and the head of the model jointly to obtain the trained local model. Afterwards, the client update is sent to the server, this update contains the parameters of the full trained local model.

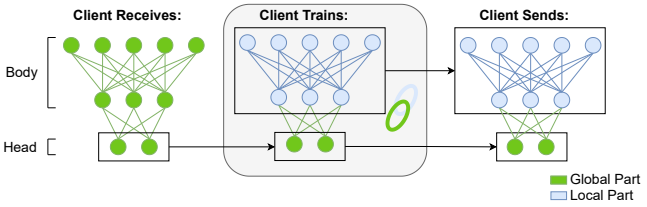


Figure 3. Example of a model for large clients in the *FedHybridAvgLG* algorithm.

3.2.2 Large Clients. Figure 3 illustrates the scheme of a model for a large client in the *FedHybridAvgLG* algorithm. For these clients, which have a large enough data set to personalize a model, *FedHybridAvgLG* leverages the LG-FedAvg algorithm. Hence, in each communication round, after receiving the model parameters, the clients only update their model head keeping their local body (instead of the whole model as in small clients) and then train both the local body and the received head jointly to obtain the trained local model. Afterwards, the client update is sent to the server. The local update contains both the trained head (as in LG-FedAvg), but also the local body (contrarily to LG-FedAvg).

3.3 FedHybridAvgLGDual

Fed*erated* ***Hybrid*** *Fed****Avg LG****-FedAvg* ***Dual*** *Model (FedHybridAvgLGDual)* is an approach different from the previous one, that requires the larger clients to calculate two different models (hence the dual in the name).

3.3.1 Small Clients. The procedure for smaller clients is exactly the same as in the *FedHybridAvgLG* algorithm. So the smaller clients simply receive the full model parameters from the server, train the model locally and send back the trained parameters. Figure 1(a), illustrates this procedure.

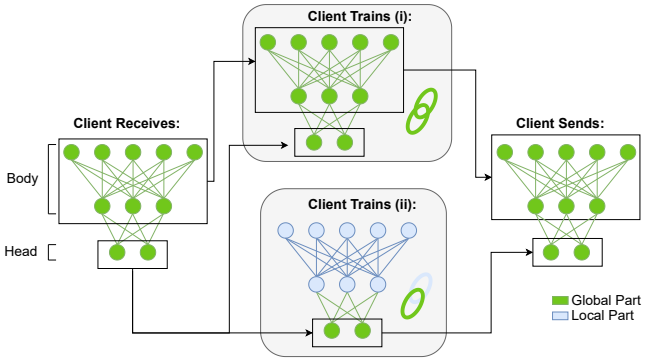


Figure 4. Example of a model for large clients in the *FedHybridAvgLGDual* algorithm.

3.3.2 Large Clients. For larger clients, this algorithm leverages two models, one based on the FedAvg algorithm and one based on the LG-FedAvg algorithm, Figure 4 illustrates the procedure for this algorithm. (i) After receiving the global model parameters, the client trains the full model received jointly simulating FedAvg. (ii) Then, the client keeps a local body which is trained jointly with the global head received from the server. After training both models, the client sends the body trained from (i) and the head trained from (ii) to the server, keeping the local body from (ii).

3.4 Client-Size Categorization

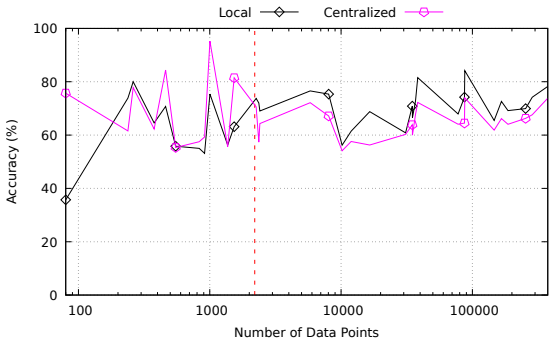


Figure 5. Accuracy of local and centralized models by the number of total client data points.

Hybrid algorithms combine two different FL algorithms, having each client perform one or the other depending on a given threshold. In order to define the threshold for considering a client small, we used the data from Figure 5. This Figure illustrates the average validation accuracy from the last 5 training rounds of a total of 30 rounds for the local models and the centralized model, for each one of 33 selected clients. The procedure for choosing these clients is described in Section 4.1. The horizontal axis is in logarithmic scale. From this graph, we can identify a point where the clients start to have enough data to personalize a model to their use case, thus starting to prefer using local models instead of the centralized model, which is more general.

In this graph, we can see a point at approximately 2200 data points (marked in the figure with a red dashed line) where to the left the centralized model generally achieves better performance (except for a couple of points) and to the right the performance of local models is superior. Hence, to the left of this point, the clients do not have a sufficient amount of data to personalize a model to their use case and, to the right, the clients start to have a large enough data set which allows them to create a model specialized to their data.

This threshold of 2200 data points draws the line between clients preferring a more general model, such as the centralized model, and a more personalized model, like the local model. Therefore, we defined this value as the threshold for considering a client as a small client.

3.5 Discussion

The two previously proposed hybrid algorithms present significantly different approaches. *FedHybridAvgLG* does not require the larger clients to train two different models, therefore, it is less costly computationally. Also, since this algorithm maintains a local body in the larger clients which is further specialized every round and is sent to the server, it is expected that the aggregated body in the server will be more specialized than when using the FedAvg algorithm, where the clients train a global body every round. The same happens with the global head, which will also be more specialized due to the larger clients training it with a local body.

FedHybridAvgLGDual uses a more specialized aggregated head than FedAvg, since the head sent by the larger clients to the server was trained with a local specialized body. Also, contrarily to *FedHybridAvgLG*, the body sent by the larger clients is not the local specialized body and is instead a body calculated from the received global body, thus the global body is more general than in *FedHybridAvgLG*. Furthermore, since the aggregation of the global head is performed with the heads from the smaller clients, which are not as specialized as they were trained with a more general global body, it is to be expected that the aggregated heads of *FedHybridAvgLG* and *FedHybridAvgLGDual* are less specialized than LG-FedAvg.

Lastly, since *FedHybridAvgLGDual* requires more computation from the larger clients, this algorithm only becomes viable if the obtained models provide some advantage.

4 Experimental Study

4.1 Experimental Setup

In order to evaluate each one of the federated algorithms, we leveraged an OUTSYSTEMS data set composed of the code developed by 881 clients. From this data set, we selected 33 clients for evaluation. Each client maintains the data relative to one organization which uses the *Service Studio* platform, that is, it keeps all the action flows of that organization.

To extract the 33 clients, the clients were partitioned according to their number of action flows. The first partition includes all the clients with less than 64 flows and all the

following partitions increase exponentially in size by a factor of 2, creating 11 partitions in total. Afterwards, 3 clients were randomly selected from each partition.

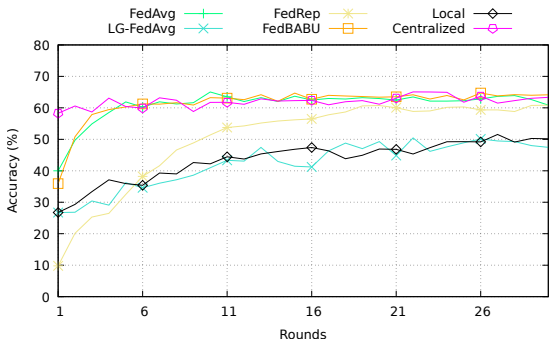
The evaluation was performed in the AWS cloud where each client was run on a separate instance. For the federated algorithms, 30 communication rounds were performed and for each one all the 33 clients participated, that is, there was no client selection since in the case of OUTSYSTEMS there are no communication or hardware restrictions. The local models were obtained using the data of each one of the 33 clients and the centralized model using the data of all the 33 clients in a single instance.

4.1.1 Model Performance. Since the clients' data set is balanced, the performance of the obtained models was evaluated using the accuracy of the models in each client's test data set, that is, the percentage of correct predictions over the total predictions. In the analysis of the results we split the clients into three groups (large clients are split into intermediate and big clients):

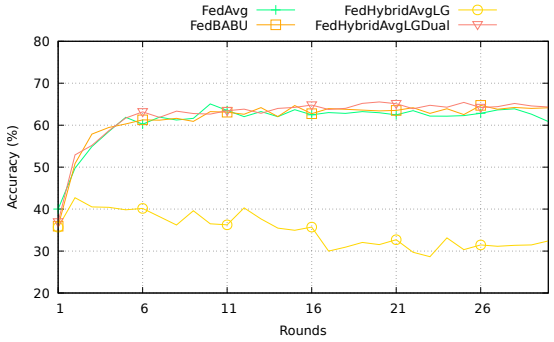
- Clients with a small number of data points (until 2200 data points, as explained in Section 3.4);
- Clients with an intermediate number of data points (between 2200 and 31700 data points). The value 31700 was obtained from the percentiles of the total number of data points for the 881 clients and it corresponds to the 75% percentile, and;
- Clients with a big number of data points (above 75% percentile, that is above 31700 data points).

4.2 Node Kind Prediction Task

In this section, we analyse the experimental results for the node kind prediction task. A more complete experimental evaluation which includes experiments with a more complex prediction task, with the prediction of novel actions and with the variation of some hyperparameters are available in the thesis [6]. In Section 4.2.1, we analyse the results for the algorithms proposed in the literature (FedAvg, LG-FedAvg, FedRep and FedBABU) and the centralized and local models. Then, we analyse the performance of our hybrid proposals in Section 4.2.2. In order to facilitate the interpretation of the results, we created two different graphs for each type of client, one which contains the literature approaches (includes the federated algorithms from the literature and the local and centralized algorithms) and another which contains the hybrid algorithms (includes the hybrid algorithms and, for comparison purposes, the literature algorithms which they intend to replicate as well as the best-performing algorithms from the literature graph for that group of clients). In the case of the FedAvg and LG-FedAvg algorithms, a single local training round was performed. For the FedRep algorithm, one local training round for the head and one for the body were performed. For FedBABU, a single local body training round was performed and since we wanted to provide the same test environment for every algorithm, no fine-tuning was performed, meaning that for this specific experience there is no personalization mechanism for FedBABU. As such a fixed classifier was used. Therefore, we performed a separate experience where we fine-tune the models for a single round before evaluation. The results of this experience are presented in Section 4.2.3.



(a) Literature Algorithms



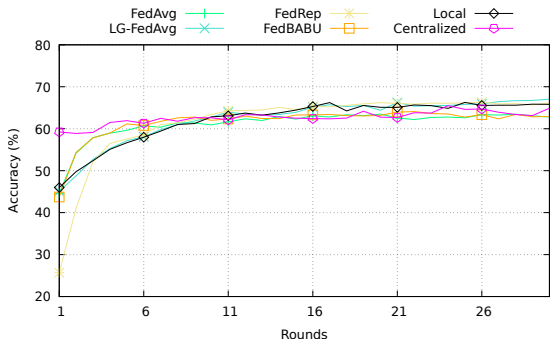
(b) Hybrid Algorithms

Figure 6. Accuracy of the various models for small clients for the node kind prediction task.

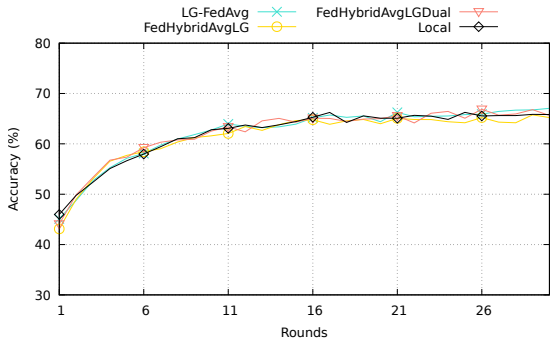
4.2.1 Literature Algorithms.

Performance for Small Clients. Figure 6(a) shows the evolution of the average accuracy for the small clients throughout the training/communication rounds for each one of the algorithms. In terms of the federated models from the literature, we can see that the FedAvg and FedBABU algorithms are the ones which obtain the best accuracy (with a slight advantage from FedBABU), followed by the FedRep algorithm and lastly by the LG-FedAvg, meaning that personalizing the head is preferable to personalizing the body. The LG-FedAvg algorithm achieves the worst performance, a fact that can be justified by the few data points of the client which do not allow for proper personalization of the body.

We can also check that the centralized model achieves worse accuracy than both FedAvg and FedBABU in a considerable amount of rounds. Lastly, we see that the local models are inferior to both the centralized model and the FedAvg, FedBABU and FedRep algorithms, which shows the importance of client collaboration for smaller clients.

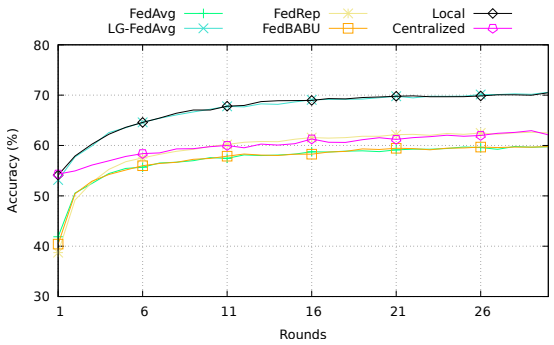


(a) Literature Algorithms

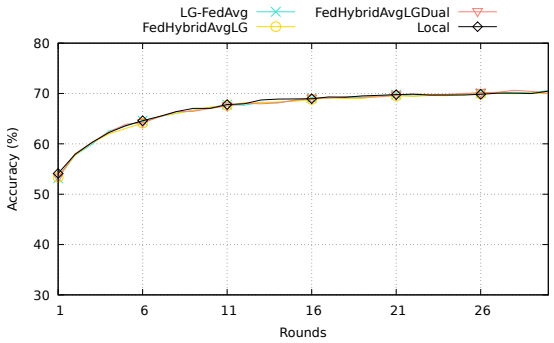


(b) Hybrid Algorithms

Figure 7. Accuracy of the various models for intermediate clients for the node kind prediction task.



(a) Literature Algorithms



(b) Hybrid Algorithms

Figure 8. Accuracy of the various models for big clients for the node kind prediction task.

Performance for Intermediate Clients. Figure 7(a) illustrates the evolution of the average test accuracy for the clients with an intermediate number of data points. We can see greater proximity between the accuracies of the four federated algorithms. Furthermore, the personalization algorithms are superior to FedAvg and FedBABU. Also, FedBABU achieves slightly better performance than FedAvg in most of the rounds. Finally, note that towards the end of the training, LG-FedAvg and FedRep end up achieving better accuracy than both the local models (about 0.3% to 1% superior) and the centralized model (about 2% to 3% superior), with LG-FedAvg surpassing FedRep, meaning that for clients with more data, personalizing the body of the model is best.

Performance for Big Clients. Figure 8(a) illustrates the evolution of the average accuracy for big clients. In this case, we can see a tendency similar to the one of the intermediate clients, where the personalization algorithms are superior to FedAvg and FedBABU, which have similar performance. However, the LG-FedAvg algorithm is superior to the FedRep algorithm, meaning that personalizing the body is the best option for clients with a lot of data. Also, towards the end of the training where the body of LG-FedAvg starts to be more specialized, the accuracy becomes slightly superior to the one of the local model (difference of 0.1% to 0.2%) and the centralized model (difference of 7% to 8%).

Discussion of Results. From the obtained results we can conclude that there is no strategy that is the best for all types of clients. For clients with few data points, the personalization of the head is easier than the body, since the body typically has a greater number of parameters, therefore, it is harder to personalize. However, for these clients, either the collaboration on the full model or on the body but using a fixed head is preferable, since the low amount of data makes personalization ineffective. The FedAvg algorithm, which trains the whole model collaboratively, obtains results very close to those of the centralized model, being superior in a considerable amount of rounds. Also, FedBABU manages to achieve slightly better results than FedAvg, which means that collaboratively training the head might introduce some noise into the model and so it is preferable to train the model with a fixed head.

As the number of data points grows (intermediate and big clients) the data becomes specific and in sufficient quantity to train, individual client models. Therefore, the centralized model becomes inferior to local models and personalization algorithms are superior alternatives to FedAvg and FedBABU. Also, the personalization of the body offers greater results

than that of the head and actually, slightly superior to local models resulting in a difference of up to 1% in accuracy. This indicates that the collaboration on the head might help these larger clients classify some more general data points which are less specific to the client and the local model fails to classify. As such, we can conclude that for these clients, personalizing the representation is preferable to personalizing the classifier, which is somewhat surprising since the literature mentions that it is expected for the heterogeneity to reside in the classifier and not in the representation.

In environments where data privacy is required, the development of a hybrid approach between the FedAvg or FedBABU algorithms (for smaller clients) and the LG-FedAvg algorithm (for bigger clients) would allow bigger clients to collaborate in the construction of a federated model which would benefit the smaller clients without sharing their data, while also receiving a small boost in model performance when compared to local models. This reasoning is what motivated the development of the hybrid algorithms *FedHybridAvgLG* and *FedHybriAvgLGDual*, whose results will be covered next.

4.2.2 Proposed Hybrid Algorithms.

Performance for Small Clients. In Figure 6(b) we can verify the evolution of the performance for small clients for the hybrid algorithms. We can observe that *FedHybridAvgLG* is the algorithm which has the worse performance, and in fact, it gets worse over every communication round. On the other hand, *FedHybridAvgLGDual* manages to achieve the intended performance and obtain results similar to FedAvg.

Performance for Intermediate Clients. From Figure 7(b) we can observe that *FedHybridAvgLG* does not manage to match the performance of the LG-FedAvg algorithm, which was its intended goal. Also, *FedHybridAvgLGDual* although not overperforming LG-FedAvg in every round, it manages to surpass the performance of this algorithm in some rounds and it also achieves close results in the other rounds, as intended.

Performance for Big Clients. In Figure 8(b) the evolution of the accuracy for the big clients for each one of the hybrid algorithms can be observed. The *FedHybridAvgLG* algorithm manages to achieve similar results to the LG-FedAvg algorithm, nonetheless, it achieves inferior accuracy (difference of about 0.1% to 0.6% in accuracy). Lastly, *FedHybridAvgLGDual* also achieves similar results to the LG-FedAvg algorithm but it overperforms this algorithm in some rounds (difference of about 0.1% to 0.4% in accuracy), therefore, achieving its intended goal.

Discussion of Results. The algorithm *FedHybridAvgLG* underperformed in comparison to LG-FedAvg for the intermediate and big clients, and most importantly, to FedAvg for small clients where the difference between the two is considerable and kept getting worse after each communication round. We believe this results from each larger client sending its local body for aggregation. Since every round, each local body keeps getting more and more specialized in its own unique way, the resulting aggregated model is not of use because each body "pulls" in its own direction.

The *FedHybridAvgLGDual* algorithm achieved the intended results, it achieved better performance than the FedAvg algorithm for the smaller clients and similar or better performance than the LG-FedAvg model (and consequently the local model, as they have identical performance) for the larger clients. For the smaller clients, the fact that the global head is trained with the local bodies of the larger clients means that it becomes more specialized while also managing to remain general enough not to affect the classifications of the more general data, leading to an improvement in performance. For the big clients, we have the opposite, as the small improvement in performance comes from the fact that the aggregated head contains the heads of the smaller clients which were trained with a more general body (remember that for smaller clients there is no local body), meaning the global head is more general than the one obtained from LG-FedAvg. This improves the classifications of the few data points that are more general and less specific to each client.

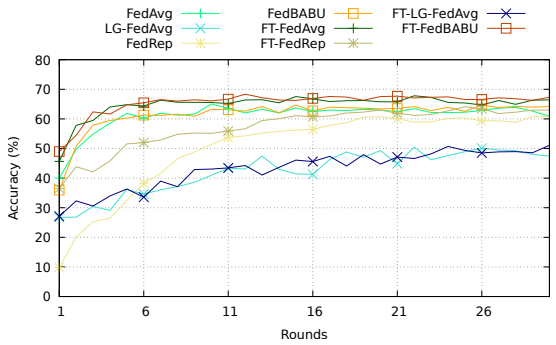
Since *FedHybridAvgLG* did not manage to achieve the intended results, in order to save test budget, we opted not to perform any further experiments with this algorithm.

4.2.3 Fine-Tuning. In order to test the influence of fine-tuning, we performed an experiment where we fine-tuned the models of the previous experiment for one round before evaluation. Fine-tuning was performed on the whole model.

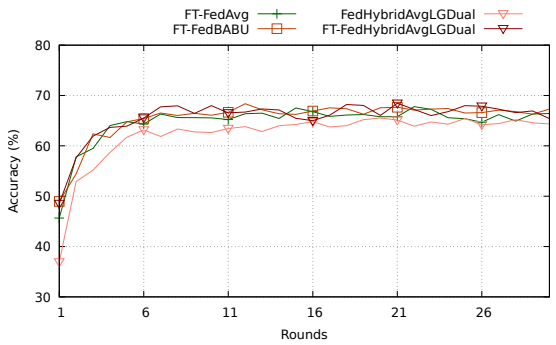
Performance for Small Clients. Figure 9 presents the results of the accuracy for small clients when fine-tuning the models for each algorithm. Similarly to the previous results, for ease of interpretation, we split the results into two graphs, thus, Figure 9(a) contains the results for the literature algorithms and Figure 9(b) contains the results for the hybrid algorithms. From the results we can verify that almost all algorithms benefit from fine-tuning before evaluation, meaning that a small personalization of the whole model can make the model adapt to the client's data. Only LG-FedAvg had no improvement after fine-tuning, which indicates that personalizing the head might not be useful for these clients.

Performance for Intermediate Clients. Figure 10 illustrates the performance for the intermediate clients with and without fine-tuning, Figure 10(a) contains the results for the literature algorithms and Figure 10(b) for the hybrid algorithms. It is possible to observe that both our approach and LG-FedAvg do not benefit from fine-tuning, which might imply that the personalization was already adequate before fine-tuning. However, the remaining algorithms manage to outperform both LG-FedAvg and our approach by quite a margin (3% to 6% in accuracy), something that indicates that intermediate clients still benefit from a more general approach which only needs an adaptation to the client data.

Performance for Big Clients. Figure 11 contains the results with and without fine-tuning for the big clients, Figure 11(a) contains the results for the literature algorithms and

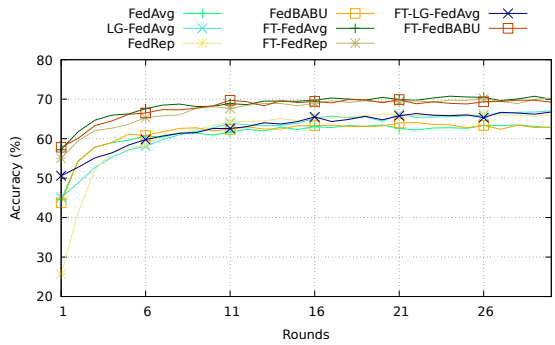


(a) Literature Algorithms

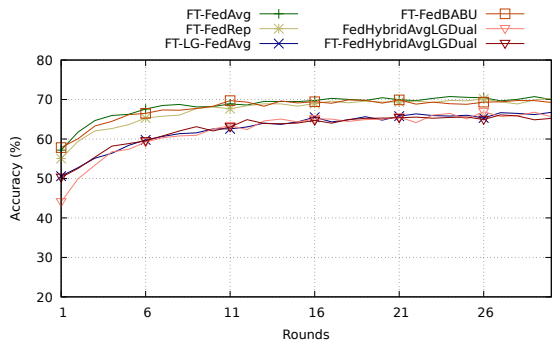


(b) Hybrid Algorithms

Figure 9. Accuracy of the various models for small clients for the node kind prediction task after fine-tuning.



(a) Literature Algorithms



(b) Hybrid Algorithms

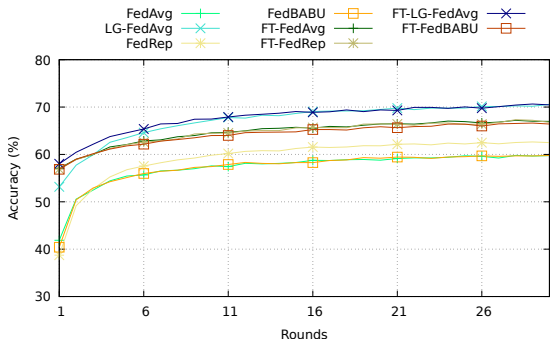
Figure 10. Accuracy of the various models for intermediate clients for the node kind prediction task after fine-tuning.

Figure 11(b) for the hybrid algorithms. There are some similarities and some differences from the previous clients. As for the similarities, once again fine-tuning our hybrid approach and LG-FedAvg does not produce any gains and in some rounds is prejudicial to the model performance. As for the differences, the remaining algorithms do not manage to overperform the former algorithms as they did for the intermediate clients, meaning the personalization of the body manages to capture the specificities of the client data more accurately.

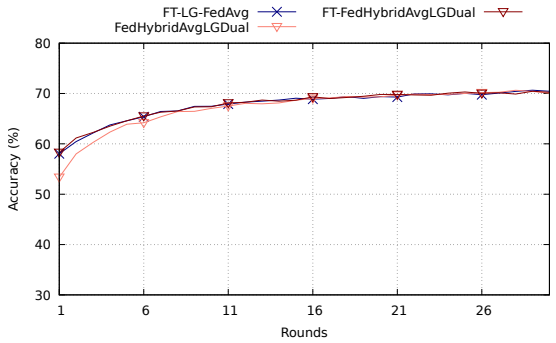
Discussion of Results. Firstly, it is interesting to notice that fine-tuned FedBABU achieves better results than fine-tuned FedAvg for smaller clients, but worse results for intermediate and big clients. The authors of FedBABU [8] argued that training the head introduced noise to the global model, as such, we can derive that the introduced noise in FedAvg is prejudicial for the smaller clients since it affects the generality of the model when these clients need a more general model. However, the noise results from the specific data of the larger clients, so when it is removed (as in FedBABU) it affects the quality of the predictions as the model becomes more general when larger clients need a more specific model.

Secondly, it is also interesting to note that the body might be the model part which best captures the specificity of the client data. By looking at the performance of the LG-FedAvg for all the clients and our hybrid approach for the intermediate and big clients (where it replicates the behaviour of LG-FedAvg), we can see that fine-tuning does not improve performance, in fact, in some rounds, it worsens the performance. This happens because the body has reached a point of personalization where more personalization simply has no effect (in the early rounds fine-tuning improved the accuracy, which does not happen in the later rounds) and the head by being more personalized stops classifying more general and out of the distribution client data as effectively, which can explain the drop in performance in some of the rounds after fine-tuning. Furthermore, if we look at the results of FedRep after fine-tuning and compare them to the ones of FedAvg after fine-tuning for the intermediate and big clients, we can see that fine-tuned FedAvg outperforms or matches fine-tuned FedRep which has a personalized head, hence the personalization of the body seems to be the key performance factor. Thus, it is preferable to have a more general head and a body which is personalized to the client data.

Lastly, fine-tuning FedAvg, FedRep and FedBABU in the intermediate clients achieved better performance than our approach and LG-FedAvg. This might mean that these clients, still do not have enough specific data to be preferable to personalize the body fully than having a more general model



(a) Literature Algorithms



(b) Hybrid Algorithms

Figure 11. Accuracy of the various models for big clients for the node kind prediction task after fine-tuning.

which is adapted to their data when fine-tuning. Thus, if fine-tuning is desired, it may be worth developing an approach similar to *FedHybridAvgLGDual* where the intermediate clients behave like the smaller clients except they fine-tune the model. However, the impact of the intermediate clients not participating in the LG-FedAvg part of the algorithm would have to be studied, since it might impact the performance of the model.

5 Conclusions and Future Work

In this thesis, we performed an experimental study to evaluate the viability of applying techniques of personalized FL to our use case, the *Service Studio* platform developed by OUTSYSTEMS. We surveyed some solutions proposed in the literature and evaluated them. The obtained results demonstrated that the amount of data of each client influences the performance of each algorithm, meaning there is no algorithm which works well for every client. Clients with fewer data prefer an algorithm which allows collaboration on the full model, as they do not have enough data to personalize part of the model. Clients with more data, prefer to collaborate on the head of the model and personalize the body. Hence, we also proposed and evaluated possible approaches that merge some of the studied algorithms, which we call hybrid algorithms. One of the proposed algorithms, *FedHybridAvgLGDual*, which merges the FedAvg and LG-FedAvg algorithms, proved to achieve similar performance to the top algorithms for all the types of clients for a task of predicting the kind of the next action to be added to an action flow. We also demonstrated that for this task fine-tuning our hybrid proposal only improves the performance for clients with a low amount of data.

Our proposed algorithm *FedHybridAvgLGDual* requires the clients with more data (intermediate and big clients) to calculate two different models in each communication round. Therefore, as future work, some other approaches might be explored to avoid having to calculate two different models. Also, it would be interesting to test our hybrid algorithm with other data sets (possibly even out of the scope of OUTSYSTEMS).

Acknowledgments

I am grateful to Filipe Assunção and Miguel Lopes for the fruitful discussions and comments during the preparation of this thesis. This work was done in the scope of a curricular internship at OUTSYSTEMS and was partially supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) as part of the project with reference UIDB/50021/2020.

References

[1] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. In *Proceedings of Machine Learning and Systems 2019, MLSys 2019*. mlsys.org, Stanford, CA, USA, 374–388.

[2] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. 2021. Exploiting Shared Representations for Personalized Federated Learning. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021 (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, Virtual Event, 2089–2099.

[3] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning* 14 (June 2021), 1–210.

[4] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. 2021. A survey on federated learning systems: vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering* (November 2021).

[5] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Think locally, act globally: Federated learning with local and global representations. *CoRR* abs/2001.01523 (July 2020).

[6] Daniel Lopes. 2022. *Federated Learning for Predicting the Next Node in Action Flows*. Master's thesis. Instituto Superior Técnico, Universidade de Lisboa.

[7] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agueray Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017 (Proceedings of Machine Learning Research, Vol. 54)*. PMLR, Fort Lauderdale, FL, USA, 1273–1282.

[8] Jaehoon Oh, Sangmook Kim, and Se-Young Yun. 2021. FedBABU: Towards Enhanced Representation for Federated Image Classification. *CoRR* abs/2106.06042 (June 2021).

[9] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2021. Towards personalized federated learning. *CoRR* abs/2103.00710 (March 2021).

[10] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (January 2019), 12:1–12:19.