MŰEGYETEM 1782

Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Measurement and Information Systems

FedLinked: A client-wise distilled representation based semi-supervised collaborative multi-task learning scheme

Scientific Students' Association Report

Author:

Attila Kádár

Advisor:

Dániel Hadházi

2021

Contents

Kivonat

Az elosztott környezetben történő multitaszk tanulás egy meglehetősen fontos probléma a federált tanulás területén belül. Számos gyakorlati alkalmazás során emberi vagy gépi szakértők törekszenek különböző, de nagy mértékben korreláló feladatok (taszkok) megoldására. Ez alapján teljesítményük kollaboráció bevezetésével javítható, figyelembe véve azt, hogy privát adatkészleteiket üzleti és adatvédelmi okokból nem oszthatják meg a többi résztvevővel. A szakirodalomban már publikált federált multitaszk megoldások az alábbi hiányosságokkal rendelkeznek:

- A multitaszk tanuláshoz szükséges tudás transzfer sok esetben a modell paraméterek vagy gradiensek megosztásával történik, ami nem megvalósítható olyan problémák esetén, ahol a privacy elvek miatt a használt modell üzleti titoknak minősül.
- Egyes módszerek esetében a tudás transzfer során nincs figyelembe véve a taszkok közötti korreláció, ezzel hátráltatva a ténylegesen hasonló feladatokat tanuló modellek közötti fokozott információ áramlást.

A félig ellenőrzött tanulás manapság is aktívan kutatott terület, melyet legfőképp a reprezentáció tanulás során felhasználható, ingyen hozzáférhető, nagy mennyiségű, címkézetlen adathalmazok felhasználása motivál. Az ismert félig ellenőrzött federált tanulási módszerek jelentős része úgynevezett pseudo labeling technikát alkalmaz, ezzel szemben az általunk javasolt megoldással bemutatjuk, hogyan lehet a címkézetlen adatokat a kollaborációban résztvevő kliensek közötti tudás megosztás megvalósítására felhasználni. Ezen munka eredménye a FedLinked eljárás, mely az előbb említett három tématerület (federált -, multitaszk -, és félig ellenőrzött tanulás) előnyeit kovácsolja össze egy komplex eljárás-blokk formájában, illetve képes az előbbiekben említett feltételeknek, megszorításoknak is eleget tenni. A FedLinked egy adatvédelmet megőrző, kollaboratív tanulási eljárás, működése a résztvevő kliensek keraszthasznosságával súlyozott reprezentáció regularizáción alapszik. Mivel a tanult taszkok klienspáronként nem diszjunktak, továbbá azok hasonlósága is feladatpáronként eltér, ezért a disztilláció előtt reperezentáció átképzést is bevezetünk, ezzel növelve a multitaszk tanulás hatékonyságát. A FedLinked eljárást kiértékeljük képosztályozási problémákon, továbbá teljesítményét összehasonlítjuk multitaszk környezetben egy FedAvg alapú módszerrel, illetve az egyszerű, nem kollaboratív tanulási eljárással. Emellett a javasolt FedLinked eljárás részletes elemzését is biztosítjuk.

Abstract

Multi-task learning in distributed environments is an important problem in the field of federated learning. In many practical problems, human or machine experts try to solve different, but highly correlated tasks, thus their performance can be improved through collaboration, but their private datasets cannot be shared with other participants because of business and privacy issues.

The already published federated multitask learning methods have the following deficiencies:

- Knowledge transfer for multitask learning is often done by sharing model parameters or gradients, which is not feasible for problems where model parameters must be kept secret due to privacy principles
- Many federated multitask methods does not take into account the correlation between task, which degrades the effectivity of knowledge transfer between models that learn similar tasks.

Semi-supervised learning is also actively researched nowadays, which is motivated by the utilization of the usually free, large set of unlabeled data in concept and representation learning. Most of the known semi-supervised federated learning methods utilize pseudo labeling techniques for clientwise self teaching, whereas in this paper we show how these unlabeled samples can be utilized in the knowledge sharing of the participants (without labeling them) during their collaboration.

We propose a method, called FedLinked, which combines the capabilities of these area (federated, multi-task and semi-supervised learning) related algorithms in the form of a complex solution-block and is able to cope with the above described constraints. FedLinked is a privacy preserving collaborative learning method, which is implemented through the regularization of representation learning based on the cross-utility of participating clients. Since the learned tasks are not disjoint per client pair, and their similarity varies by task pairs, we also introduced and utilize representation transormation before knowledge distillation to increase the efficiency of multitask learning.

We evaluate FedLinked on image classification problems and compare its performance to FedAvg based and non-cooperating clients based solutions in multi-task scenarios. Besides, a detailed analysis of our proposed method is provided.

Chapter 1

Introduction

Nowadays, the use of intelligent service provider devices is becoming increasingly common both in industry and among end users. Examples include the widespread use of smart, wearable devices, the emergence of self-driving cars or even machine learning-based services used internally in larger companies. In many cases, the size and the diversity of locally available training dataset proves to be a bottleneck in the performance of these services. This motivates collaboration between machine learning systems with different local datasets, but have the same or at least similar tasks. A major barrier to this type of cooperation is that in most cases participants cannot share sensitive but valuable information about their local datasets without violating certain privacy principles. The concept of federated learning offers solutions to these problems by enabling effective collaboration between participating clients under the orchestration of a central server, while respecting the necessary privacy principles.

The term *federated learning* (FL) was introduced by [17]. The authors focused on a federated setting with a large number of mobile devices with limited computational power and availability. This approach has proven useful in numerous practical applications, for example Google Gboard [7] or certain features of Android Messages. After Google pioneered many benefits of centralised federated learning systems, the field became even more popular. Apple's natural language processing algorithm called "Hey Siri" also utilizes federated learning methods. In other use cases, federated methods that utilize orders of magnitude fewer, but more reliable clients are preferred. In such cases, larger companies, organisations and groups of companies are seeking to apply secure collaborative learning for example in financial risk forecasting or pharmaceutical research [21]. The FedLinked approach, proposed in this thesis, is more suited to this type of large enterprise tasks, in which the privacy compliance is more critical than usual.

In case of several problems, clients participating in federated learning have to solve different tasks, which may correlate more or less. Therefore multi-task learning (MTL) approaches [4] can be useful in these federated learning scenarios. Researches in medical imaging [15] is an example, where collaboration is desirable, however HIPAA is regulating the use and protection of health information.

In certain cases, where sufficient amount of data is available, labelling them can still be an expensive task in terms of time and resources. To solve these type of problems, semi-supervised learning(SSL) solutions [24] help to improve the performance of machine learning models by using all the labeled and unlabeled data. To the best of our knowledge, not many published methods are available at the intersection of these three areas (federated learning, multi-task learning, semi-supervised learning), despite the many practical applications mentioned above. In this thesis we propose a method called FedLinked, that combines the capabilities of these area related algorithms in the form of a complex solution-block. The proposed method is designed to be efficient for federated multitask learning problems where neither datasets nor model parameters can be shared due to strict privacy rules. Beside utilizing the whole solution, each component of FedLinked can be used separately for collaborative learning tasks.

The remainder of this thesis is organized as follows: Section 2 briefly overviews related, commonly used federated, multi-task and SSL approaches, highlighting their advantages and shortcomings. In Section 3 we present our proposed method - FedLinked - and all of its components. Section 4 contains the description of datasets and models we use to evaluate FedLinked. Section 5 provides the description of our experiments including comparison between FedLinked and other approaches and a detailed analysis of the experimental result. In Section 6 we provide an ablation study, where impact of the main components are examined in terms of model performance. Finally the concluding remarks are presented in Section 7.

Chapter 2

Related work

2.1 Cross-device versus cross-silo federated learning

Two well-known and widely used federated learning schemes are Cross-device and Cross-silo federated learning. The main characteristics of the Cross-device training process are the following:

- A large number (up to 10^{10}) of low-performance clients (usually mobile or IoT devices) train a central model to solve certain problems (usually computer vision, natural language processing or sensor data analysis)
- Each client has its own local, private dataset and cannot access the private data of other clients
- A central server controls the federated learning process, but it does not see the private datasets of the clients.
- Clients are highly unreliable: a large proportion of participants (even more than 5%) may become unavailable during a learning round[1] (e.g. due to a power or mains failure power problems).

In contrast to Cross-device FL, the main features of the Cross-silo FL setting are the following:

- Typically 2 to 100 clients are involved in the learning process. These clients are mostly large companies or organisations (e.g. medical or financial) or geo-distributed datacenters.
- Each client trains the central model on an isolated, relatively large local dataset (data silo).
- Similar to the Cross-device FL, the process is controlled by a central server.
- Unlike in Cross-device FL, the relatively small number of clients are reliable, failures or drop outs during training are rare.

[1]In the centralised case, a learning round is a sequence of steps during which clients perform a few epochs of local training and exchange information with the central controller (or, in the decentralised case, between each other)

Note that these two FL schemes are only two of the relatively commonly used methods, but there are a number of other approaches. The FedLinked method we propose has mostly the characteristics of the Cross-silo FL scheme.

2.2 Fully decentralized / Peer-to-peer learning

In the previously presented Cross-device and Cross-silo FL environments, a central controller (server) manages the whole federated learning and at the end of the training process it stores the model collaboratively trained by the clients. However, there are applications where it is not justified or even feasible to have a central server: it can potentially be a single point of failure or bottleneck for a large number of participating clients. In addition, in some cases, due to privacy principles and data security issues, it is not allowed for an external partner (the server itself) to be involved in the learning process. These constraints can also arise even in such federated multitasking problems (for which we propose a solution by FedLinked), where the participating clients collaboratively train each other's model, but locally each client's tasks differ (even in number), so the goal is not to build a global model, but to improve the performance of each local model, with as little information sharing as possible.

Fully decentralised learning (e.g [11], [20]) helps with problems where clients need to communicate directly with each other. To increase efficiency, this communication/information sharing does not necessarily need to happen between every client pair.

It is worth noting that even in decentralised case, it may sometimes be necessary to use a central controller at the beginning of the process to calibrate the clients, e.g. to determine which specific algorithm and what hyperparameter values the clients should use or which client pairs should exchange information, etc.

2.3 Preserving the privacy of client data

Federated learning involves the sharing of information between clients or between client and server in order to make collaboration effective. This must be done in a way that takes into account all the privacy principles regarding the local client data and model parameters, i.e. no sensitive information should be shared.

As the FL process involves multiple participants, the vulnerability of the system increases. There may be a case where a client's device is compromised by an attacker and the messages received from the server are leaked. Similarly, data stored on the server (including model or client data) can fall into unauthorised hands. This raises a key question for federated learning: *How to achieve effective collaboration without clients sharing their private data and without the trained model(s) falling into unauthorized hands?*

Many federated ([17], [12]) and federated multitask [22] solutions have been published in the literature that use various methods to protect the privacy of data stored by clients during collaboration, but to the best of our knowledge, there are very few federated multitask learning methods that can provide efficient collaboration without sharing either private training data or local model parameters. As it was mentioned earlier in the introduction, our proposed FedLinked method is designed to be efficiently applicable even to those kind of federated multitask learning problems where sharing private datasets or model parameters is not possible. Additionally, in our case, the clients can use networks with different architecture as well.

2.4 Commonly used federated learning algorithms

One of the most widely used methods in the field of federated learning is FedAvg [17]. This approach provides a solution to typical federated learning problems where the goal is to train a single global model in a distributed manner on the local (private) datasets of participating clients. In each learning round, each client first optimizes an instance of the current global model on their own datasets, then the updated parameters are averaged to form the updated global model on the coordinating server. Algorithm 1 summarizes the main steps of FedAvg according to [17].

Algorithm 1 Federated Averaging (FedAvg)

```
Server executes:
1: initialize $x_0$
2: for t = 1, 2, ..., T do
3:     $S_t \leftarrow$ (random set of M clients)
4:     for each client $i \in S_t$ in parallel do do
5:         $x^i_{t+1} \leftarrow ClientUpdate(i, x_t)$
6:     end for
7:     $x_{t+1} \leftarrow \sum_{k=1}^{M} \frac{1}{M} x^k_{t+1}$
8: end for

ClientUpdate($i, x$):
   for j = 1, ..., K do
2:     $x \leftarrow x - \eta \nabla_x f(x; z)$, where $z \sim P_i$
   end for
4: return $x$
```

This algorithm has proven to be quite useful for many cross-device problems [6], but it cannot be used in certain cases where sharing local gradients is not possible due to privacy reasons, or each client trains an individual model on non-IID datasets. A possible solution to this kind of problem is provided by FedMD [12] method. After training each client's local model to convergence on their local and public datasets, this algorithm uses knowledge distillation [8] by computing class scores on the public dataset and sending the result to the coordinating server in each round of federated learning. The average of the class scores (also called consensus) is used by each client for further fine tuning. Algorithm 2 summarizes the main steps of FedMD according to [12].

Algorithm 2 FedMD

```
Input: Public dataset $D_0$, private datasets $D_k$, independently designed models $f_k,\ k = 1, ..m$
Output: Trained model $f_k$
1: Transfer learning: Each client trains $f_k$ to convergence on the public $D_0$ and then on its private $D_k$ datasets.
2: for t = 1, 2, ..., P do
3:     Communicate: Each party computes the class scores $f_k(x_i^0),\ x_i^0 \in D_0$ on the public dataset, and transmits the result to a central server
4:     Aggregate: The server computes an updated consensus, which is an average: $\overline{f}(x_i^0) = \frac{1}{m}\sum_k f_k(x_i^0)$.
5:     Distribute: Each client downloads the updated consensus $\overline{f}(x_i^0)$.
6:     Digest: Each client trains its model $f_k$ to approach the consensus $\overline{f}$ on the public dataset $D_0$.
7:     Revisit: Each client trains its model $f_k$ on its own private data for a few epochs..
8: end for
```

One advantage of this method is that it provides a higher generalization capability for local models than separated learning. On the other hand, it is not adequate in many cases, when different clients learn different tasks. Our proposed method can handle these cases by estimating utility scores between different clients learning different tasks.

2.5 Multitask learning methods

Regarding multitask learning, we mainly overview methods used in computer vision, since in this thesis we present the practical application of the FedLinked method on image processing problems. Multitask machine learning architectures mostly partition the architecture of the models into two main components: task-specific and shared component. This partitioning is done in a way that assures better generalization ability through sharing, while minimizing negative transfer. A survey by [5] summarizes several approaches based on this principle, the first being the shared trunk or hard parameter sharing method (Figure 2.1).

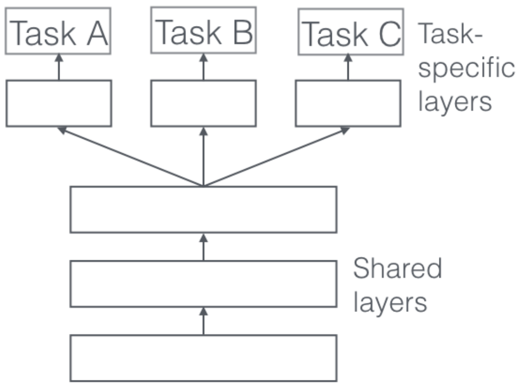


Figure 2.1: Hard parameter sharing structure [25]

In shared trunk architectures a global feature extractor shared with all tasks is followed by an individual output branch for each task. The use of this concept appears in several

MTL related publications (e.g. [13]), but it is difficult to integrate into federated learning environments, where the same feature extractor is not necessarily optimal for all clients due to differences in tasks. Unlike Shared-trunk, the Cross-talk (soft parameter sharing) approach has separate networks for each task, with information flow between parallel layers in the task networks, referred to as *cross-talk*. An example of the Cross-talk architecture can be seen on Figure 2.2.

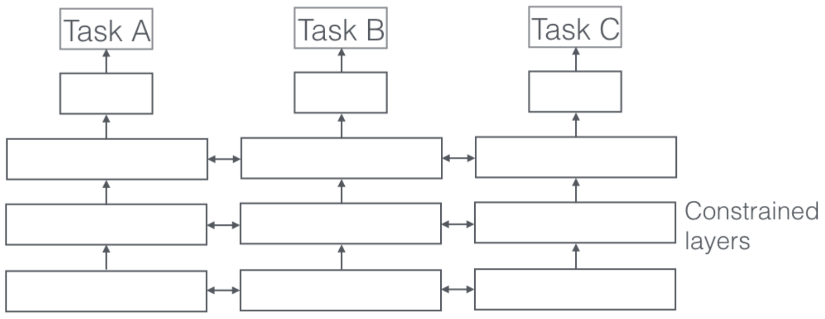


Figure 2.2: Cross-talk (soft parameter sharing) structure [25]

[18] provides one possible implementation of Cross-talk approach. Like FedLinked, this approach is able to take into account the similarities between tasks, thus increasing the effectiveness of collaboration. The other MTL methods mentioned by [5], such as Predictive Distillation or Task Routing, are not feasible in federated schemes since they produce output for multiple tasks from the same given input. Further general MTL approaches are described by [26]. Feature learning approaches are based on the following hypothesis: if models share their internal representation among themselves and work together to form a common representation that carries information about all of the tasks, then this potentially can improve the generalization capability of each model on their local tasks (by modelling the data manifold more accurately). In federated learning environment this idea is realized by the previously mentioned FedMD algorithm. One disadvantage of Feature Learning is assuming that all tasks are similar to each other, which is often not the case, especially in distributed systems. In contrast, Task Relation Learning approaches (e.g. [3]) seek to identify similarities between tasks and use this information for collaboration. All the methods overviewed by [26] estimate the similarity of each task based on the datasets associated with them. However, this is not always feasible due to privacy reasons. Another possible federated multitask learning approach is MOCHA [22], which also takes into account the similarity of the tasks during collaboration, but it has the following disadvantages: the method assumes that every client learns only one task, however in our use case, every client may learn multiple tasks. Additionally this method violates important privacy principles, because it estimates task similarities based on shared model parameters.

Our proposed method seeks to overcome the aforementioned weaknesses of these approaches by realizing a privacy preserving federated multitask learning scheme.

2.6 Semi-supervised learning algorithms

Utilizing the non-labeled training data in image classification problems can also increase the performance of the solution (also in federated environments). [2] describes a possible application of semi-supervised learning in a federated learning environment. The authors propose the use of pseudo-labeling technique to increase the size of local datasets on client side, thus creating a more robust global model. Another approach [14] locally optimizes the joint loss function calculated from both labeled and unlabeled data using the well-known Mean Teacher [23] SSL technique. Just like FedSem, this method aggregates the learned parameters by clients to form the updated global model. The proposed FedLinked differ from these approaches in the way of utilizing the unlabeled data. It uses a global, unlabeled dataset directly to share knowledge between the clients, which is critical in order to preserve the privacy of the local datasets. Finally, we highlight the Distillation-based DS-FL method [9] which, similarly to FedMD, computes global logits by aggregating the logits computed by clients on unlabeled data. The authors propose a new logit aggregation method, called Entropy Reduction Aggregation (ERA). DS-FL approach assumes that clients learn identical tasks on data sets not necessarily from an i.i.d. distribution. In contrast, FedLinked estimates which tasks might be useful to others and does not even require that these tasks produce same output on the same input.

Chapter 3

Proposed method

In this section, we first give a formal definition of the federated multitasking problem set that is the motivation of the proposed FedLinked method. After this, the proposed method is described (with theoretical background aspects and the implementation considerations).

3.1 Problem definition

The proposed FedLinked algorithm realizes collaboration in a federated environment between separate local models which learn similar or eventually identical, overlapping tasks, and estimates the similarities between them. In order to formally define this problem and describe our proposed solution, we first introduce the following notations:

- K: Number of clients participating in collaboration
- $X^{(i)} = \{(x_j^{(i)}, d_j^{(i)})\}$: Private, local dataset of the i^{th} client. $x_j^{(i)}$ denotes the j^{th} input data and $d_j^{(i)}$ is the corresponding output label.
- $X^{(0)} = \{x_j^{(0)}\}$: the public, global, unlabeled dataset (available to all participants). It is used during collaborative representation regularization.
- θ_i : Parameters of the i^{th} client.
- $f^{(r)}(\cdot, \theta)$: Representation of the input data as a function of θ parameters immediately after applying the feature extractor layers. The size of this representation vector must be the same for all clients[1].
- $A_{k,j}$: Utility of the k^{th} client with respect to the j^{th} client. These client-pair specific weights determine the regularization of representations in the collaborating phase.
- $f_i^*(\cdot)$: Regularizer representation of the i^{th} client calculated from the representations and the utility of other clients with respect to the i^{th} one.

The subtasks to be solved: determining initial θ_i parameter vectors in function of $X^{(i)}$ datasets; based on $f^{(r)}(X^{(0)}, \theta_i)$ representations, estimating matrix A and finally using all these representations and utility values to find the optimal θ_i parameters of all the local models.

[1]θ_i parameter codes the structure of the i-th client too, but these structures are fixed during training.

3.2 Structure of FedLinked method

FedLinked realizes collaborative learning in multi-task environment through the regularisation of the representations of the participants (by utilizing the cross-utility between them). During regularisation, clients who learn correlating tasks, have a greater influence on each other's representations. This collaboration is realised in a way that neither the client's private data, nor their parameter vectors, nor the list of the learnt tasks need to be shared with the server or other clients.

Collaborative learning consists a series of learning rounds. One learning round has two main phases: training local models on their private datasets and representation regularization by the public dataset based on the cross-utility coefficients estimated by the coordinating server. The steps performed during one learning round in FedLinked framework can be seen on Figure 3.1: firstly, clients train local models separately and send $f^{(r)}(X^{(0)}, \theta_i)$ representation vectors to the server, which updates the cross-utility coefficients. Based on these coefficients and public representations, the server determines and sends the proposed representations client-wise, which is then approximated locally during regularization step.

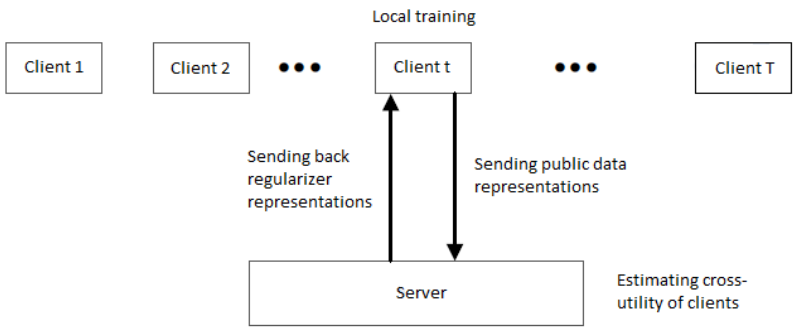

Local training
Client 1
Client 2
Client t
Client T
Sending back regularizer representations
Sending public data representations
Server
Estimating cross-utility of clients

Figure 3.1: FedLinked framework

In the remainder of this section, we describe in detail all the components of a learning round.

3.3 Representation regularisation, cross-utility estimation

The central component of FedLinked is the task utility-based knowledge distillation realized by representation regularization. This component assumes that collaborating models with similar tasks should use similar representations in order to achieve better generalisation capability (by better modelling common parts of the data manifolds) beside minimising local train loss. The estimation of the cross-utility values, the representation regularization and local training is done by minimizing the following objective function:

$$\begin{aligned} L(A, \{\theta_i\}) = & \sum_{k \neq j} \sum_{x \in X^{(0)}} A_{k,j} \cdot \|f^{(r)}(x, \theta_k) - f^{(r)}(x, \theta_j)\|_2^2 + \beta \sum_j loss(f, \theta_j, X^{(j)}) \\ & + \gamma \sum_j \|\theta_j\|_2^2 \\ & \text{subject to: } \sum_{k \neq j} A_{k,j}^2 = \eta : \forall j \end{aligned} \tag{3.1}$$

Optimization of the objective function L is done in function of the primal variables : θ_j and $\{A_{k,j}\}_{k\neq j}$ for $\forall j, k \in 1, K$. The first term of the objective function is responsible for cross-utility estimation and for representation regularization. Intuitively, when $A_{k,j}$ takes a high positive value, the distance of the representations supposed to be smaller, since the two models are presumably learning tasks which requires similar representations.

The second term represents the local supervised learning, during which clients minimize their own loss function on their private, labeled datasets. The experiments presented in the next section were carried out on classification problems, thus using Categorical-Cross Entropy loss function as the second term of the object function.

Finally an L_2 regularisation term is included in order to control the complexity of the local models. By setting the value of β and γ, the weight of local training and L_2 regularization can be controlled.

The constraint $\sum_{k\neq j} A^2_{k,j} = \eta$, $\forall j$ is used to control the values of cross-utility coefficients. Therefore the value of η specifies the strength of representation based regularization (thus the collaboration).

As in many federated learning algorithms, FedLinked uses Alternating Optimization (AO) in order to minimize the first term and last two terms of the objective function. This approach has the advantage that clients do not have to share their private data with the server, since the optimization can be done locally, while the determination of the cross-utility coefficients is done by the server. The first phase of alternation (optimizing the first term) is also done in two steps using block coordinate descent optimization: first optimizing it in function of cross-utility matrix A, using θ_j parameters computed in the previous iteration, then optimizing only over θ_j for $\forall j \in \{1, K\}$ (considering $A_{k,j}$ as constants) .

The practical implementation of the main component of FedLinked is carried out by performing the following steps:

0. Pre-training of all models on their private, labeled datasets. This is equivalent to optimising the second term of the objective function for all clients.

1. Each client sends its representations computed on the public data points to the server:

$$\alpha(j, x) = f^{(r)}(x, \theta_j) : \forall x \in X^{(0)}, \forall j \tag{3.2}$$

2. Given the public representations, the server performs the first step of representation based collaboration to estimate cross-utility coefficients:

$$\begin{aligned} \{A_{k,j}\}^* &= \arg\min_{A_{k,j}} \sum_{k\neq j} \sum_{x\in X^{(0)}} A_{k,j} \cdot \|\alpha(k, x) - \alpha(j, x)\|_2^2 \\ \text{subject to:} & \sum_{k\neq j} A^2_{k,j} = \eta : \forall j \end{aligned} \tag{3.3}$$

The minimization of the squared distance of the representations can be approximated by maximizing their inner product (Equation 3.4) if the deviation of the norm of the representation vectors is small. Therefore we introduced the L2 regularization of the parameter vectors for this purpose.

$$
\begin{aligned}
&\min . -\sum_{k \neq j} \sum_{x \in X^{(0)}} A_{k,j} \cdot \langle \alpha(k,x), \alpha(j,x) \rangle \\
&\text{subject to: } \sum_{k \neq j} A_{k,j}^2 = \eta \ , \forall j
\end{aligned}
\tag{3.4}
$$

Based on this observation, the optimal value of cross-utility coefficients ($\{A_{k,j}\}^*$) can be approximated in closed form using Lagrange duality. Considering the condition $\sum_{k \neq j} A_{k,j}^2 = \eta : \forall j$, the solution of the optimization problem is as follows:

$$
\{A_{j,k}\}^* = \frac{\sum_{x \in X^{(0)}} \langle \alpha(j,x), \alpha(k,x) \rangle}{\sqrt{(\sum_{u \neq v} (\sum_{x \in X^{(0)}} \langle \alpha(u,x), \alpha(v,x) \rangle)^2)/\eta}} \tag{3.5}
$$

3. Given the current representations and the utility coefficients, clients locally perform the second step of the representation based collaboration. In order to avoid any kind of privacy violation, for each client j, the server sends the client-wise regularizer representation $f_j^*(X^{(0)}) = \frac{\sum_{k \neq j} (A_{k,j} \cdot \alpha(k, X^{(0)}))}{\sum_{k \neq j} (A_{k,j})}$ in one step instead of sending cross-utility matrix A and representation matrices $(\alpha(k, X^{(0)}) : \forall k \neq j)$ one by one. Therefore the j^{th} client optimizes the following function:

$$
\{\theta_j\}^* = \arg\min_{\theta_j} \| f^{(r)}(X^{(0)}, \theta_j) - f_j^*(X^{(0)}) \|_2^2 \tag{3.6}
$$

4. Finally, the second phase of alternating optimisation, the local supervised learning takes place. Then the next learning round begins with step 1.

These steps are consistent with the process shown on Figure 3.1. Note that the sign of the cross-utility coefficients can be negative, which can be exploited if the representation vectors can contain negative activations. If representation vectors are non-negative, then the optimal $A_{j,k}$ coefficients are definitely non-negative.

3.4 Transforming representations

In practice, the tasks learned by clients are not necessarily disjoint, and their similarity also varies from task pair to task pair. For this reason, a direct comparison of the representations computed by different models may not be informative (e.g. it is possible that the i^{th} activation of one client's representation matches with the j^{th} activation of another client's representation, where $i \neq j$). For this reason, we propose an algorithm to transform between representations of client pairs before performing knowledge distillation. A linear transformation can be applied on representation vectors, where multiplication by the matrix $B_{k,j}$ transforms representation of the k^{th} client for the j^{th} client. Using this transforming component, the objective function in Equation 3.1 is modified as follows:

$$
\begin{aligned}
L(A, \{\theta_i\}) = &\sum_{k \neq j} \sum_{x \in X^{(0)}} A_{k,j} \cdot \|B_{k,j} \cdot f^{(r)}(x, \theta_k) - f^{(r)}(x, \theta_j)\|_2^2 + \beta \sum_j loss(f, \theta_j, X^{(j)}) \\
&+ \gamma \sum_j \|\theta_j\|_2^2 \\
&\text{subject to: } \sum_{k \neq j} A_{k,j}^2 = \eta : \forall j \; ; \; B_{k,j}^T \cdot B_{k,j} = I : \forall k \neq j
\end{aligned}
\tag{3.7}
$$

The constraint $B_{k,j}^T \cdot B_{k,j} = I$ ensures that the norm of the representation vectors are not changed by the transformation. The optimal transformation matrices maximize Equation 3.8, since the maximum of the cross-similarity coefficients can only be obtained for solutions which maximize Eq. 3.8.

$$
\{B_{k,j}\}^* = \underset{B_{k,j}}{\arg\max} \sum_{k \neq j} \sum_{x \in X^{(0)}} \langle B_{k,j} \cdot f^{(r)}(x, \theta_k), f^{(r)}(x, \theta_j) \rangle \tag{3.8}
$$

Using the definition of the trace of a matrix, the expression can be rewritten as:

$$
\begin{aligned}
\sum_{x \in X^{(0)}} \langle B_{k,j} \cdot f^{(r)}(x, \theta_k), f^{(r)}(x, \theta_j) \rangle &= \sum_{x \in X^{(0)}} trace(B_{k,j} \cdot f^{(r)}(x, \theta_k) \cdot f^{(r)}(x, \theta_j)^T) \\
&= trace(B_{k,j} \cdot \sum_{x \in X^{(0)}} f^{(r)}(x, \theta_k) \cdot f^{(r)}(x, \theta_j)^T)
\end{aligned}
\tag{3.9}
$$

Let $R_{k,j} = \sum_{x \in X^{(0)}} f^{(r)}(x, \theta_k) \cdot f^{(r)}(x, \theta_j)^T$, furthermore let $(\Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T)$ and $(U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T)$ be the singular value decomposition of matrices $B_{k,j}$ and $R_{k,j}$ respectively.

Equation 3.9 can then be further modified as follows:

$$
\begin{aligned}
trace(B_{k,j} \cdot \sum_{x \in X^{(0)}} f^{(r)}(x, \theta_k) \cdot f^{(r)}(x, \theta_j)^T) &= trace(B_{k,j} \cdot R_{k,j}) \\
= trace((\Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T) \cdot (U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T))&
\end{aligned}
\tag{3.10}
$$

Therefore, using the cyclic property of the trace operator and the orthonormality of the matrices in SVD decomposition, the optimization problem is equivalent to the following:

$$
\begin{aligned}
&\underset{\Gamma_{k,j}, \Lambda_{k,j}, \Phi_{k,j}}{\arg\max} \; trace(S_{k,j} \cdot V_{k,j}^T \cdot \Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T \cdot U_{k,j}) \\
&\text{subject to: } \Gamma_{k,j}^T \cdot \Gamma_{k,j} = I, \Phi_{k,j}^T \cdot \Phi_{k,j} = I, \Lambda_{k,j} = I)
\end{aligned}
\tag{3.11}
$$

If S is diagonal and all of its elements are non-negative, then for a given matrix X, the value of $trace(S \cdot X) = \sum_i S_{(i,i)} \cdot X_{(i,i)}$ will be maximal if the elements on the principal diagonal of X are maximized.

Furthermore, due to the constraints of the optimization, $V_{k,j}^T \cdot \Gamma_{k,j}$ and $\Phi_{k,j}^T \cdot U_{k,j}$ are also orthonormal matrices. We need to maximize the principal diagonal of their multiplications, which implies the choice $\Lambda_{k,j} = I$, taking the $B_{k,j}^T \cdot B_{k,j} = I$ constraint into account.

Then the matrix $V_{k,j}^T \cdot \Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T \cdot U_{k,j}$ is also orthonormal, with the elements on its principal diagonal being maximal if $V_{k,j}^T \cdot \Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T \cdot U_{k,j} = I$. This is satisfied if $\Gamma_{k,j} = V_{k,j}$, $\Phi_{k,j} = U_{k,j}$.

Thus the optimal transformation matrix can be expressed as follows:

$$B_{k,j} = V_{k,j} \cdot I \cdot U_{k,j}^T = V_{k,j} \cdot U_{k,j}^T$$
where $V_{k,j}$ and $U_{k,j}$ matrices are calculated from the SVD decomposition of R matrix (3.12)

This component can be easily incorporated into the algorithm presented at the end of Section 3.3 by performing the following steps:

0. Pre-training of all models on their private, labeled datasets. This is equivalent to optimising the second term of the objective function for all clients.
1. Each client sends its representations computed on the public data points to the server (Eq. 3.2).
2. Given the public representations, the server computes the transformation matrix and transformed representations:

$$\begin{aligned} R_{k,j} &= \sum_{x \in X^{(0)}} \alpha(k,x) \cdot \alpha(j,x)^T = U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T \\ B_{k,j} &= V_{k,j} \cdot U_{k,j}^T \\ \alpha(k,j,x)' &= B_{k,j} \cdot \alpha(k,x) \ , \forall x \in X^{(0)}, \forall k,j \ , \ k \neq j \end{aligned} \tag{3.13}$$

3. The server estimates cross-utility coefficients based on the transformed representations:

$$\{A_{j,k}\}^* = \frac{\sum_{x \in X^{(0)}} \langle \alpha(j,x), \alpha(k,j,x)' \rangle}{\sqrt{(\sum_{u \neq v} (\sum_{x \in X^{(0)}} \langle \alpha(u,x), \alpha(v,u,x)' \rangle))^2)/\eta}} \tag{3.14}$$

4. Given the regularizer representations, clients locally perform the second step of the representation based collaboration (similarly to Eq. 3.6 but $f_i^*(X^{(0)})$ is determined from the transformed representations).
5. Finally, the second phase of alternating optimisation, the local supervised learning takes place. Then the next learning round begins with step 1.

Note that it can be also shown, that the optimization of B matrices can be done separately from the optimization of A matrix.

3.5 Further modifications for enhance performance

Let the *representation trunk* of a model be the set of those layers that produce the representations used in the collaboration, and similarly let *task-specific layers* be those that produce output from representation. Right after representation regularization, the upper layers still reflect the state after the last local learning, in contrast to the lower layers. In order to resolve this kind of inconsistency, we propose *fine-tuning* of the upper layers after representation regularization. During fine-tuning, the lower layers are frozen and only the upper layers are trained on private dataset, then the final step of learning round - the local supervised learning - takes place on the unfrozen network.

Furthermore, to avoid overfitting, the use of early stopping during local supervised learning is recommended. Similarly, for all clients, we propose the use of a so-called *rollback* component during collaborative learning. Applying this component means performing the following steps: at the end of each learning round, the models are evaluated on their validation dataset and after the last learning round, the best performing parameters are back-loaded. The main purpose of rollback is that if a client stops benefiting from the collaboration after a certain number of rounds, it can still preserve its best performing parameters achieved during the federated learning without actually leaving the collaboration (which might have a negative impact on other participants).

Algorithm 3 summarizes the steps of the FedLinked procedure involving all components.

Algorithm 3 FedLinked

```
Input: Public dataset $X^{(0)}$, private datasets $X_i$, independently designed models $c_i$ , $i = 1, ..K$
Output: Trained models $c_i$
1: $best\_round\_accuracy_i \leftarrow 0, \forall i = 1, 2, ..., K$
2: Each client pre-trains $c_i$ to convergence on their private $X_i$ datasets.
3: for r = 1, 2, ..., Rounds do
4:     for j = 1, 2, ..., K do
5:         $\alpha(j, X^{(0)}) \leftarrow f^{(r)}(X^{(0)}, \theta_j)$
6:     end for
7:     for $j, k = 1, 2, ..., K, j \neq k$ do          ▷ first step of representation regularization
8:         $R_{k,j} \leftarrow \sum_{x \in X^{(0)}} \alpha(k, x) \cdot \alpha(j, x)^T$
9:         $U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T \leftarrow R_{k,j}$
10:        $B_{k,j} \leftarrow V_{k,j} \cdot U_{k,j}^T$
11:        $\alpha(k, j, X^{(0)})' \leftarrow B_{k,j} \cdot \alpha(k, X^{(0)})$
12:        $\{A_{j,k}\}^* \leftarrow \frac{\sum_{x \in X^{(0)}} \langle \alpha(j,x), \alpha(k,j,x)' \rangle}{\sqrt{(\sum_{u \neq v} (\sum_{x \in X^{(0)}} \langle \alpha(u,x), \alpha(v,u,x)' \rangle))^2)/\eta}}$
13:    end for
14:    for j = 1, 2, ..., K do          ▷ second step of representation regularization
15:        $f_j^*(X^{(0)}) \leftarrow \frac{\sum_{k \neq j} (A_{k,j} \cdot \alpha(k,j,X^{(0)})')}{\sum_{k \neq j} (A_{k,j})}$
16:        $\{\theta_j\}^* \leftarrow \arg\min_{\theta_j} \|f^{(r)}(X^{(0)}, \theta_j) - f_j^*(X^{(0)})\|_2^2$
17:    end for
18:    Fine-tuning of the task-specific layers for each $c_i$ on $X_i$
19:    Each client locally trains its model (updates the whole $\theta_i$ parameter vectors)
       on $X_i$ for a few epochs (with early stopping).          ▷ $2^{nd}$ phase of AO
20:    for j = 1, 2, ..., K do
21:        $round\_accuracy_i \leftarrow evaluate(c_i)$
22:        if $round\_accuracy_i \geq best\_round\_accuracy_i$ then:
23:            $save\_model(c_i)$
24:        endif
25:    end for
26: end for
27: for i = 1, 2, ..., K do          ▷ Rollback
28:     $c_i \leftarrow load\_saved\_model(i)$
29: end for
```

Chapter 4

Datasets and learning environment

In this section, we briefly introduce the datasets and learning environment used to evaluate FedLinked.

4.1 Datasets

Each experiment is evaluated on both the CIFAR10 [10] image and the Street View House Numbers (SVHN) image classification datasets [19]. Both are divided in the following way: 25% of the samples are selected into the public dataset, while the remainders are partitioned between the three clients in a balanced, but not i.i.d. way. Each client has a train, two validation (for local early stopping and rollback) and test datasets (which are disjoint).

Table 4.1 shows numerical statistics on the partitioning of both datasets among clients. Columns Val_ES and Val_RB contain statistics about the validation sets used for local early stopping and rollback respectively. The datasets are distributed so that each data point is assigned to at most one client. Therefore the size of training datasets is reduced but due to this partitioning, the effect of knowledge transfer is not biased by same data points during the collaboration.

Table 4.1: Number of samples after splitting the datasets

	Train		Val_ES		Val_RB		Test		Public	
	CIF10	SVHN	CIF10	SVHN	CIF10	SVHN	CIF10	SVHN	CIF10	SVHN
M0	4374	5450	600	748	486	606	540	673	5500	9474
M1	3280	5918	450	812	365	658	405	731		
M2	4374	9346	600	1283	486	1039	540	1154		

4.2 Task partitioning among clients

During each experiment, three clients are trained. Both examined datasets consist of data points from 10 different classes. These are:

- **CIFAR10:** airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck
- **SVHN:** All the 10 digits

Our goal is to divide these tasks in a few-partner federated learning setting so that each of the groups listed below includes at least one client-pair:

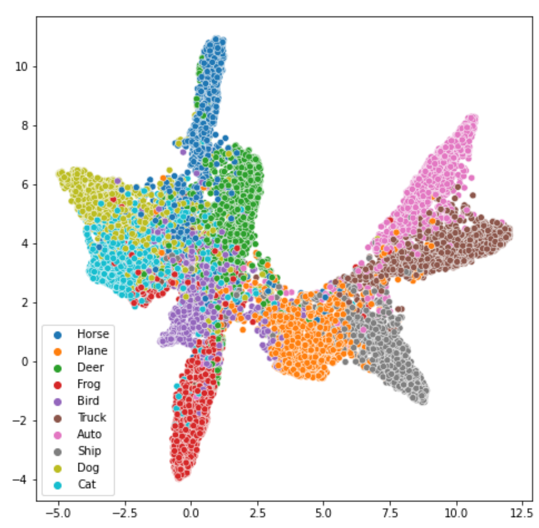
- client-pairs learning tasks whose representations might be useful to each other (in our case M0-M1 client-pair meets these conditions)
- client-pairs learning tasks for which the distance between their representation vectors is relatively large (in our case M0-M2 and M1-M2 client-pairs meet these conditions)

This kind of task partitioning allows us to examine the extent to which the FedLinked procedure can improve the performance of local models under varying degrees of correlation between tasks learned by client-pairs. Furthermore, it might help to answer the question of how the cross-utility of the tasks' representations are related to the similarities between the tasks estimated by human intuition.

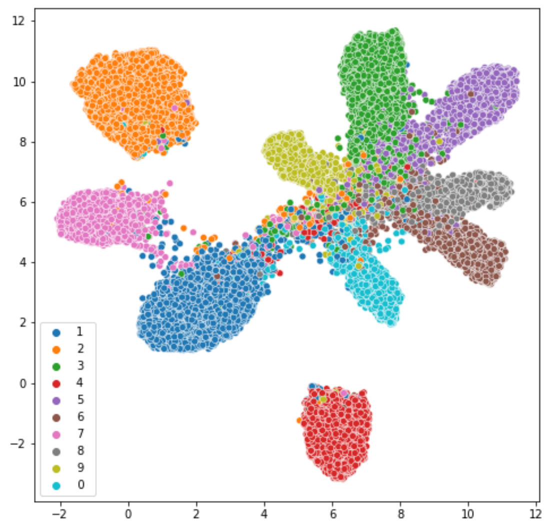
In order to obtain an appropriate task partitioning that meets the above criteria, the following steps are carried out:

1. We train models on the full CIFAR10 and SVHN datasets for all tasks. The architecture of these models is the same as described in Section 4.3 for both datasets.
2. The high dimensional representation vectors obtained on the last hidden layers of the trained models are projected into a two dimensional subspace using the distance preserving UMAP [16] algorithm. The resulting projections can be seen on Figure 4.1.
3. Based on the two-dimensional projections, we identify the task-groups to which the models have assigned similar (spatially close to each other) representations and use these groups to construct the desired partitioning. Table 4.2 shows the resulting partitioning.

On Figure 4.1/a, it can be observed that the representations of the vehicle-related tasks are close to each other (especially car - truck and airplane - ship pairs). The similarity between the representations of horse and deer and between dog and cat is also spectacular. Also for the SVHN dataset, it can be seen on Figure 4.1/b that the representations of the 7-1, 3-5, 6-8 task-pairs are quite close. These results, besides helping to determine the desired task partitioning, suggest that in case of the examined datasets, the spatial distance of the representation vectors is correlated to the cross-similarities between tasks determined by human intuition.



(a) CIFAR10



(b) SVHN

Figure 4.1: Distance preserving 2D projections of representations of the whole dataset learned by one model

Based on the analysis above, for both datasets we can partition the tasks so that M0 and M1 clients learn more similar tasks based on both human interpretation and inner representation similarities, while the M2 client learns more distinct tasks (Table 4.2).

Table 4.2: Task partitioning

	M0	M1	M2
CIFAR10	cat, deer, dog, horse	cat, deer, frog	airplane, automobile, ship, truck
SVHN	0, 5, 6, 9	0, 3, 6, 8	1, 2, 4, 7

4.3 Model architecture and hyperparameter optimization

The local clients are simple CNNs. A convolution block is the sequence of the following layers: convolutional layer, ReLU, convolutional layer, ReLU, Maxpooling, Dropout. The representation trunk of a client consists of b convolution blocks and one linear layer with r neurons, while the task-specific layers are composed of another linear layer and a softmax non-linearity. Another dropout layer is applied right before the output layer, with a different dropout rate (p_l) from the one utilized in the convolution blocks (p). In a given block, both convolution layers contain the same number of 3x3 size filters, while the number of filters in the i-th block is $c{\cdot}2^{i-1}$. For both datasets, the values of b, r, c, dropout rates p, p_l and other hyperparameters (η, number of epochs, batch size, learning rates) of FedLinked procedure were determined using the Optuna automatic hyperparameter optimiser [1].

The following are required for automatic hyperparameter optimisation with Optuna:

- A scalar metric to evaluate the training for given hyperparameter values. In our case, this metric is the average of the validation accuracy scores achieved by each client.
- A bounded interval for all hyperparameters. During the optimization, Optuna tries to find the best values for each parameter from these intervals. Table 4.3 shows the

upper and lower bounds of the intervals we used to optimize the hyperparameters mentioned above.

Table 4.3: Upper and lower bounds of intervals examined during hyperparameter optimization

	CIFAR10		SVHN	
	Min	Max	Min	Max
Nr. blocks (b)	1	4	1	4
Repr. size (r)	64	512	32	256
First channel (c)	8	128	8	64
Dropout (p)	0.0	0.8	0.0	0.8
Last dropout(p_l)	0.0	0.8	0.0	0.8
Etha (η)	0.8	3.0	0.8	3.0
Batch ratio	0.008	0.12	0.008	0.12
Local epochs	1	20	1	20
Init. epochs	1	40	1	40
Dist. epochs	1	60	1	60
Finetune epochs	1	20	1	20
Rounds	3	12	3	12
Learning rate	0.0001	0.01	0.0001	0.01

For the robustness of the optimization, 3 different training were performed for each set of hyperparameter values chosen by Optuna. Of these three training, the one achieving the lowest average accuracy score is used to evaluate the given hyperparameter setting.

In the following experiments, we compare the performance of FedLinked with two other learning schemes: a FedAvg-based method (see Section 5.3) and a fully separated learning (see Section 5.2). Table 4.4 contains the hyperparameter values obtained as a result of the described Optuna optimization algorithm for all three methods and for both datasets.

Note that, in order to reduce runtime, the optimization is performed in two steps: first, for both datasets, we optimize the parameters that define the model architectures (b, r, c, p, p_l), and then for each dataset - method pair, we optimize the training hyperparameters (number of rounds and epochs, learning rate, batch ratio).

Table 4.4: Optimal values of the hyperparameters

	CIFAR10			SVHN		
	FedLinked	FedAvg	Sep.	FedLinked	FedAvg	Sep.
Nr. blocks (b)	3	3	3	3	3	3
Repr. size (r)	256	256	256	128	128	128
First channel (c)	64	64	64	16	16	16
Dropout (p)	0.36	0.36	0.36	0.24	0.24	0.24
Last dropout(p_l)	0.79	0.79	0.79	0.35	0.35	0.35
Etha (η)	2.97	-	-	2.22	-	-
Batch ratio	0.02	0.03	0.03	0.11	0.01	0.01
Local epochs	17	13	20	18	20	14
Init. epochs	19	16	24	30	16	15
Dist. epochs	18	-	-	48	-	-
Finetune epochs	13	8	-	15	20	-
Rounds	7	9	10	11	10	10
Learning rate	2e-4	4e-4	2e-4	3e-3	4e-4	4e-4

Parallel plots on Figures 4.2 and 4.3 show the performance of the different hyperparameter settings tested during the optimization of the training hyperparameters. The higher the average accuracy of a parameter setting, the darker the corresponding line on the parallel plot. For both datasets, the hyperparameter setting with the highest accuracy is highlighted in red. However, it can also be observed in both plots that several different hyperparameter settings can result in high accuracy values. It should be also noted that the parameters of the best setting are never in the upper or the lower limit, therefore this validates the chose of the examined possible intervals of the value of these parameters.

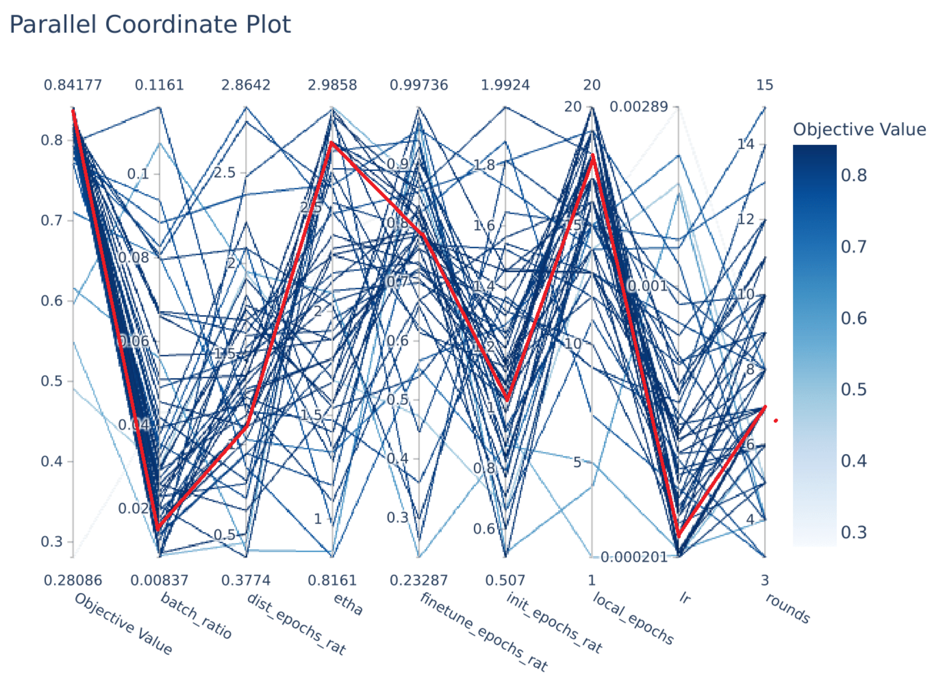


Figure 4.2: Performances with different hyper-parameter settings on CIFAR10 dataset: from left to right the columns show the accuracy on the validation dataset, the batch ratio, the number of the distillation epochs in ratio of local epochs, the value of etha, the number of fine tuning epochs in the ratio of the local epochs, the number of initialization epochs in the ratio of the local epochs, the number of local epochs, the learning rate, and the number of rounds. The red curve corresponds to the optimal values.

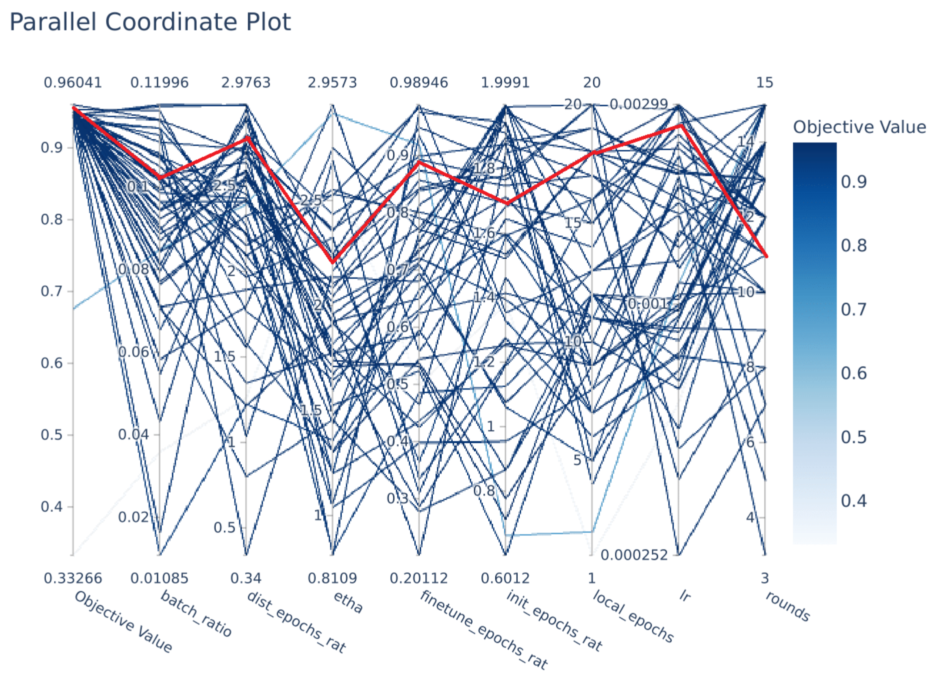


Figure 4.3: Performances with different hyper-parameter settings on SVHN dataset

Chapter 5

Experiments and analysis

In this chapter we present the experiments used to evaluate and examine the proposed FedLinked procedure and we also provide a detailed analysis of the results.

5.1 Utility coefficients

In this experiment, the FedLinked method is used to collaboratively train three participating clients, and at the end of the collaboration (after 7 and 11 learning rounds respectively for CIFAR10 and SVHN), the resulting utility coefficients are examined for each client pair.

The results of this experiment are shown in Tables 5.1 and 5.2 for both CIFAR10 and SVHN datasets. In case of both datasets, it can be observed that M2 proved to be less useful for the other models, while M0 and M1 assigned mutually high utility values to each other. Furthermore, for M2, M1 proved to be slightly more useful than M0. As mentioned in the previous section, the tasks were split among clients in a way that the tasks learned by M0 and M1 were similar, including even common tasks. In this experiment, this similarity is also reflected in the value of the utility coefficients, especially in case of SVHN dataset. Nevertheless, it is important to note that the human intuition based ranks of the task similarities can highly differ from the utility coefficients based ranks depending on the given dataset and models.

Table 5.1: Utility coefficients - CIFAR10

	M0	M1	M2
M0	-	2.4278	1.7167
M1	2.3064	-	1.8767
M2	1.9505	2.2444	-

Table 5.2: Utility coefficients - SVHN

	M0	M1	M2
M0	-	1.8632	1.2092
M1	1.8075	-	1.2910
M2	1.4937	1.6439	-

5.2 Federated versus separated learning

The goal of this experiment is to compare the performance of FedLinked with fully separated learning, where clients do not use the unlabeled, public $X^{(0)}$ dataset and do not collaborate at all. Since the optimal number of learning rounds is different for federated and separated learning, we consider the local test accuracy of each client after the first, middle (chronologically) and last learning rounds in both cases.

The results of this experiment are shown in Table 5.3. Each field contains the mean and standard deviation of client-wise test accuracy values calculated from multiple runs. It can be seen that, at all stages of training, models trained with FedLinked outperform the ones trained separately. The performance of models learning similar tasks (M0 and M1) clearly increases due to collaboration, but even M2 - learning highly different tasks from those of other models - benefits from FedLinked method on the CIFAR10 dataset.

Table 5.3: Federated learning compared to separated learning

		First round		Middle round		Last round	
		Federated	Separated	Federated	Separated	Federated	Separated
CIFAR10	M0	0.74 ± 0.017	0.71 ± 0.012	0.76 ± 0.011	0.71 ± 0.015	0.76 ± 0.013	0.73 ± 0.013
	M1	0.83 ± 0.015	0.82 ± 0.021	0.85 ± 0.021	0.83 ± 0.019	0.86 ± 0.017	0.839 ± 0.021
	M2	0.88 ± 0.013	0.86 ± 0.011	0.89 ± 0.007	0.86 ± 0.024	0.89 ± 0.005	0.87 ± 0.013
SVHN	M0	0.94 ± 0.006	0.94 ± 0.003	0.95 ± 0.003	0.94 ± 0.003	0.95 ± 0.002	0.94 ± 0.004
	M1	0.93 ± 0.010	0.92 ± 0.012	0.94 ± 0.008	0.93 ± 0.009	0.95 ± 0.009	0.93 ± 0.007
	M2	0.95 ± 0.004	0.95 ± 0.005	0.96 ± 0.008	0.96 ± 0.003	0.96 ± 0.007	0.96 ± 0.003

5.3 FedLinked versus FedAvg

This experiment investigates how efficient the federating scheme introduced in the FedLinked algorithm is compared to the simple FedAvg-based collaboration. In order to implement the latter procedure in a multitasking environment (in which the structures of the task-specific layers may vary), we apply FedAvg algorithm only on the parameters of the representation trunk. So the steps of one learning round in the FedAvg-based method are: local supervised learning, federated averaging on representation trunks, fine-tuning of the task-specific layers. It is worth noting that, unlike FedLinked, this method assumes that the representation trunk of all clients are structurally identical, which can be an undesirable restriction in federated multi-task environments. During this experiment, 10 consecutive training are performed using FedLinked, gradually increasing the size of available public dataset in each iteration, while keeping the clients' private datasets fixed. In this way, we analyze that, depending on the size of the available public dataset, how FedLinked performs compared to the FedAvg-based method in a multitasking environment in the following two scenarios:

- **FedAvg - without public data:** The private datasets of clients are the same for both FedLinked and FedAvg.
- **FedAvg - with all public data:** In case of FedAvg training, the private dataset of each client is expanded by the whole labeled public dataset.

Figures 5.1 and 5.2 show the results of the comparison between FedLinked and FedAvg based methods on both datasets. Note that the test accuracy values shown on Figures are all averages of all clients' individual test accuracy values from three trainings. For the CIFAR10 dataset, it can be seen that starting from 20% availability of the entire public dataset, FedLinked outperforms FedAvg procedure, even when FedAvg clients' local datasets are expanded by the entire labeled public dataset. This result clearly shows the effectiveness of the introduced federalization scheme (representation regularisation and transformation) compared to FedAvg based representation averaging in multi-task learning scenarios.

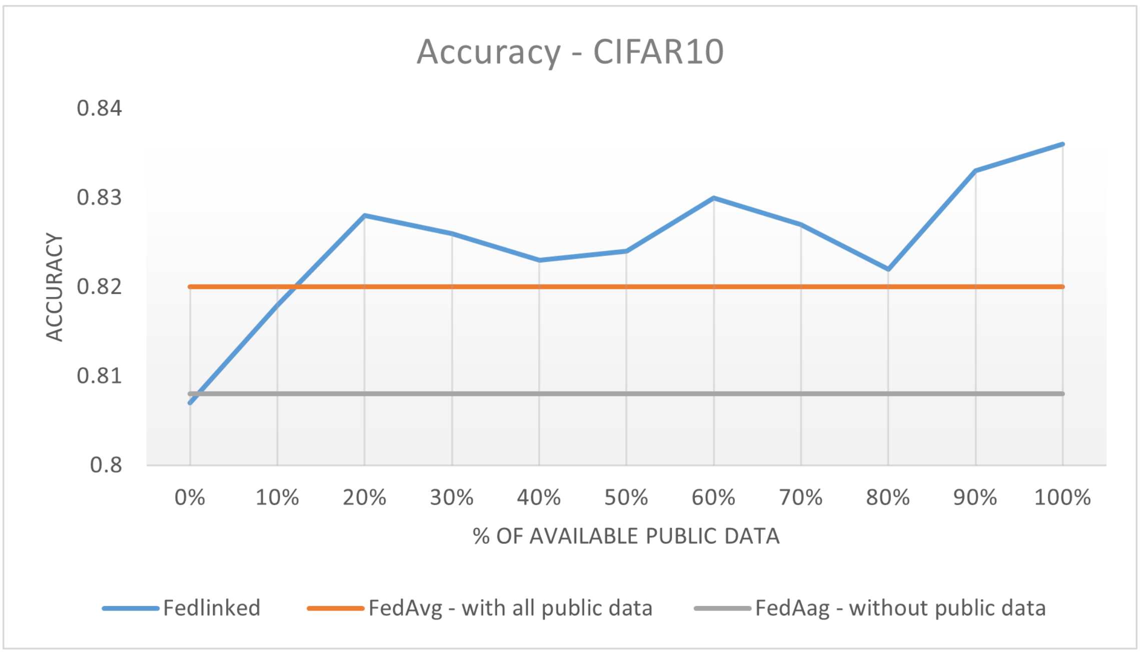


Figure 5.1: Accuracy of FedLinked compared to FedAvg-based method in the function of the number of the public, unlabeled training samples - CIFAR10 dataset

In case of SVHN dataset (Figure 5.2), the performance gap between the two methods is not so striking due to the already high test accuracy values produced during local training, but it can be observed that for the same available local dataset, FedLinked clearly outperforms the FedAvg-based method.

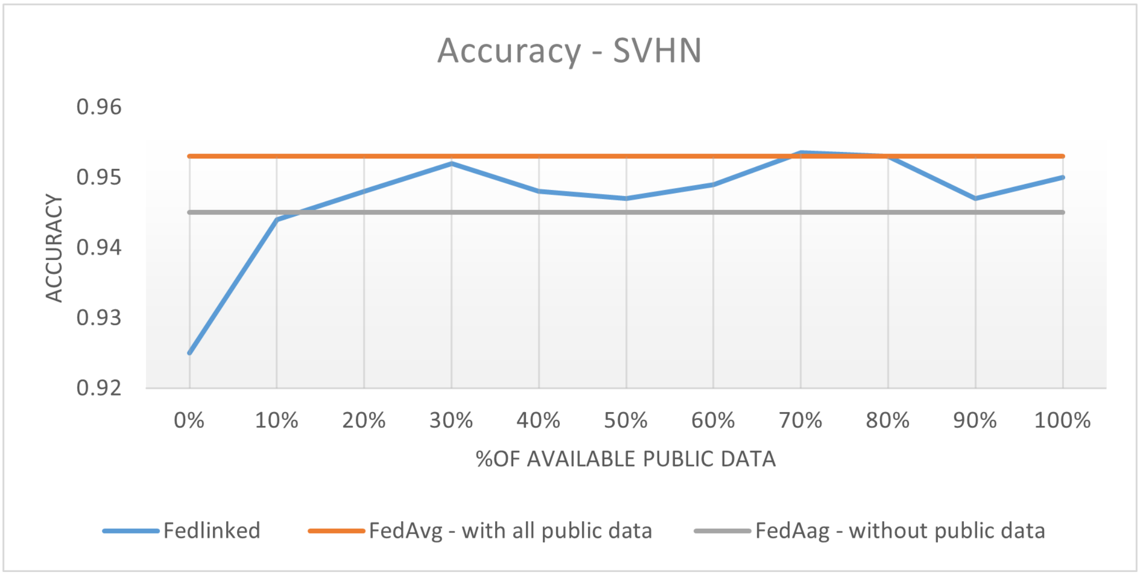


Figure 5.2: Accuracy of FedLinked compared to FedAvg-based method in the function of the number of the public, unlabeled training samples- SVHN dataset

5.4 Visualizing representations

In this experiment, we analyze the representations formed by each client's representation stub, more specifically, the impact of the collaborative regularization on the distribution of the representations. To this end, the representations[1] created by each client on the public dataset are projected and displayed in 2D after both federated and separated learning, using the UMAP ([16]) algorithm.

Figures 5.3 and 5.4 show the projected representations of all three clients on the public dataset at the end of separated (left subfigure) and FedLinked training (right subfigure), colored by task labels.

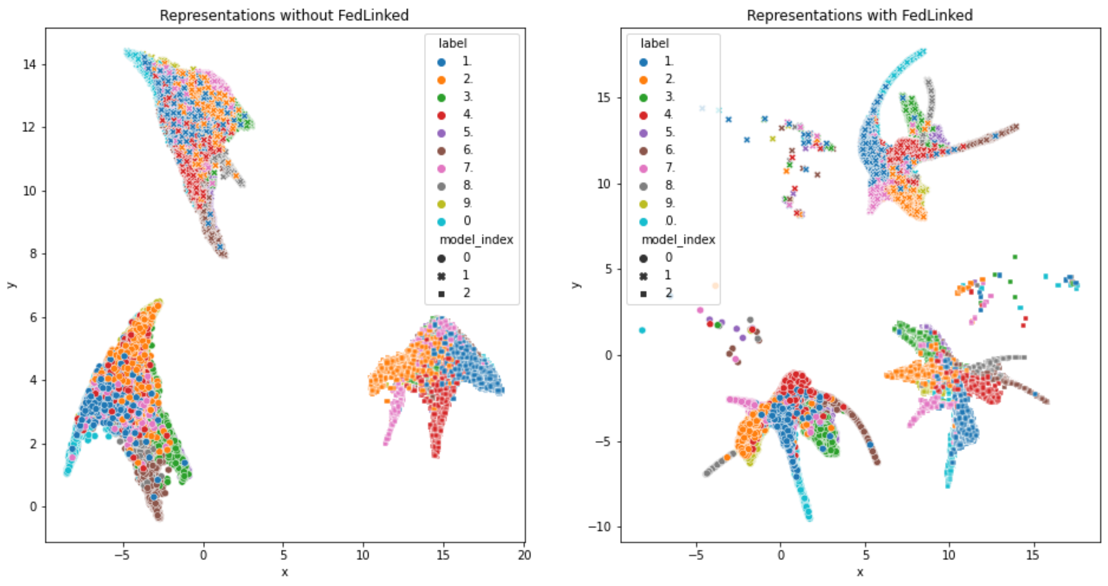


Figure 5.3: Distance preserving projected representations of public data points of SVHN dataset

For the SVHN dataset (Figure 5.3), the effect of collaboration on the formation of the representations is quite impressive: it can be observed that by utilizing FedLinked, the representations of data points belonging to each task are much more separated into homogeneous groups than in case of regular, separated training. Furthermore, at the end of collaboration, each client was able to well cluster data points of tasks that they had not seen with labels in their private, labelled datasets.

[1]Representations are vectors of size 256 (CIFAR10) and 128 (SVHN)

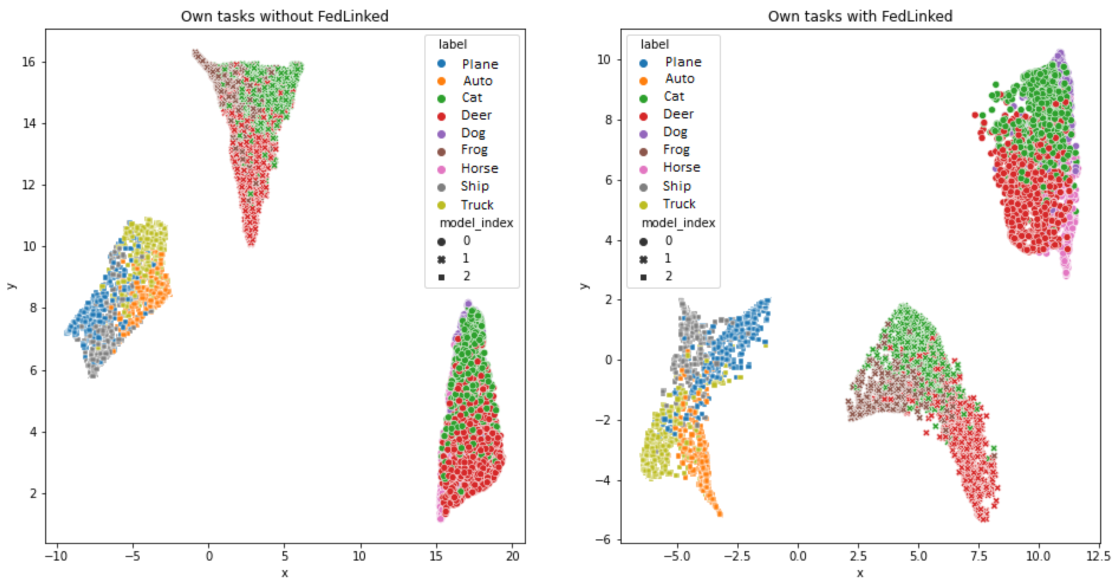


Figure 5.4: Distance preserving projected representations of public data points belonging to tasks originally learned by the given clients on CIFAR10 dataset

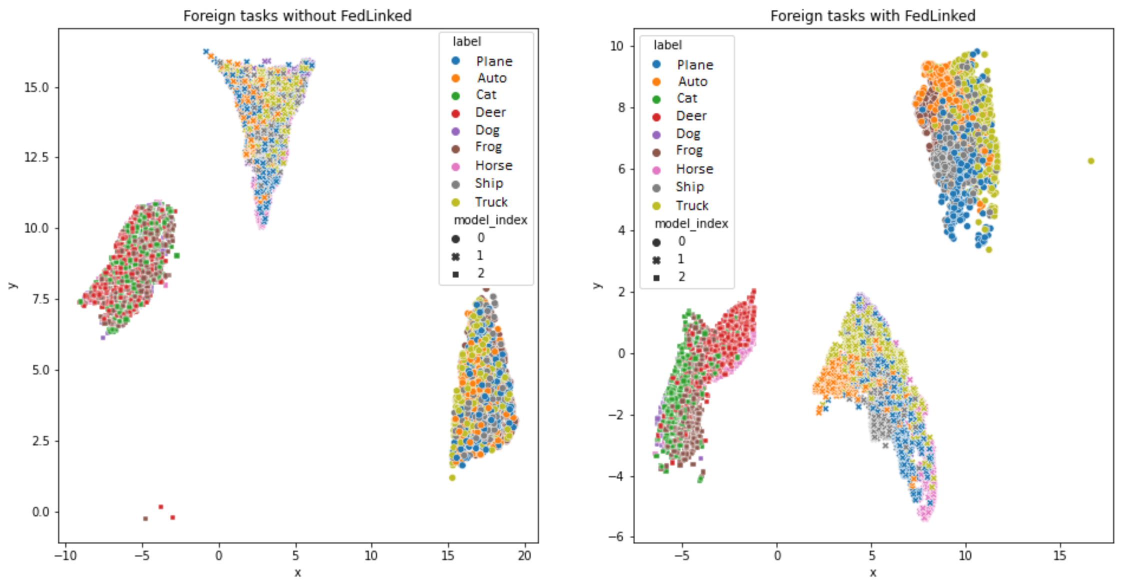


Figure 5.5: Projected representations of public data points belonging to tasks originally not learned by the given clients on CIFAR10 dataset.

As shown in the previous results, classification on the CIFAR10 dataset is a more difficult problem to learn than classification on the SVHN dataset. Therefore it is more difficult for models to separate representations belonging to different tasks into distinct, homogeneous groups. For this reason and for the sake of clarity, the projected public representations of CIFAR10 dataset are presented on two figures:

- On **Figure 5.4** the representations of those public data points are shown, which belong to tasks learned originally by the given client from its local dataset.
- On **Figure 5.5** the representations of those public data points are shown, which do not belong to tasks originally learned by the given client from its local dataset.

On Figure 5.4 it can be seen that, thanks to the use of FedLinked, the models projected the data points belonging to same tasks into well-separated, homogeneous domains, while in case of separated training, greater overlapping between representations belonging to different tasks can be observed.

Figure 5.5 shows that, in contrast to the separated learning, models trained with FedLinked have managed to group even representations belonging to tasks originally not learned from their local dataset in a well separable way.

These results show that the FedLinked-based collaboration can also help to increase the generalisation capability of the participating models.

5.5 Visualization of representation transformation

Finally, we investigate the efficiency of representation transformation, in a similar way to the previous experiment: Let $\alpha(i, X^{(0)})$ and $\alpha(j, X^{(0)})$ be the representation of the public dataset determined by the i^{th} and j^{th} client respectively. In this experiment we visualize the 2D projected form of $\alpha(i, X^{(0)})$, $\alpha(j, X^{(0)})$ and $B_{i,j} \cdot \alpha(i, X^{(0)})$ computed in a given learning round.

The visualized representations before (left side of the figure) and after (right side of the figure) the clientpair-wise transformations between model M0 and M1 on the CIFAR10 dataset are shown in Figure 5.6. As expected, the representation vectors belonging to model M0 are clearly transformed into the domain where the representation vectors of model M1 are located. Furthermore, it can be observed that even after the transformation, the representation vectors belonging to same tasks remain in the same subspace, as shown by the colored homogeneous clusters becoming dense.

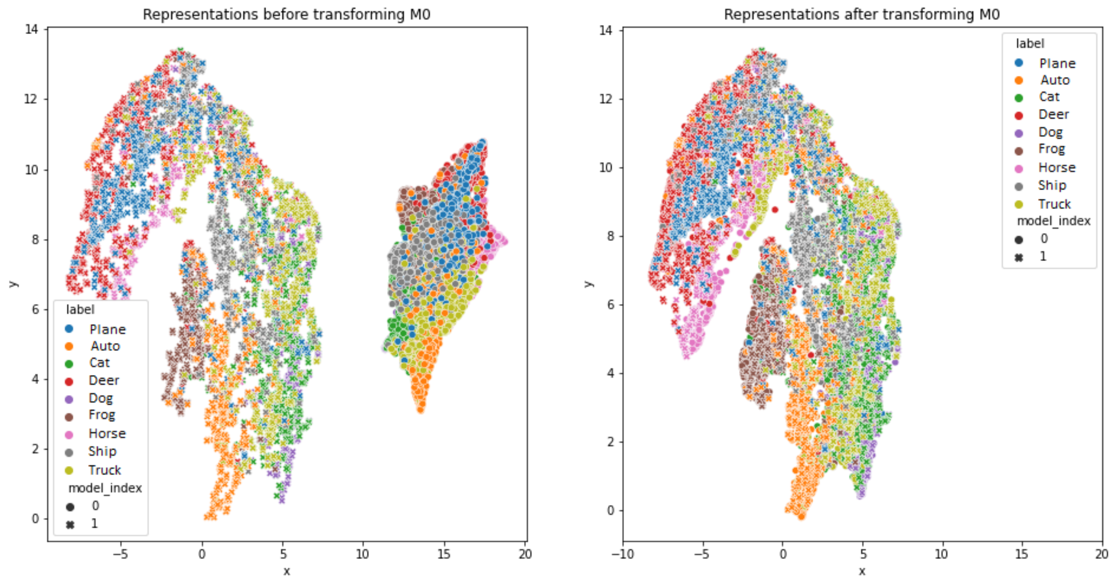


Figure 5.6: Public data representations computed by M0 and M1 before and after transforming representations of M0 on CIFAR10 dataset

Similarly to CIFAR10 dataset, Figure 5.7 shows the visualized representations before and after the clientpair-wise representation transformation on the SVHN dataset. Here again, it can be observed that the representation vectors belonging to the same tasks were correclty mapped into the same subspace, increasing the number of elements in the corresponding homogeneous clusters.

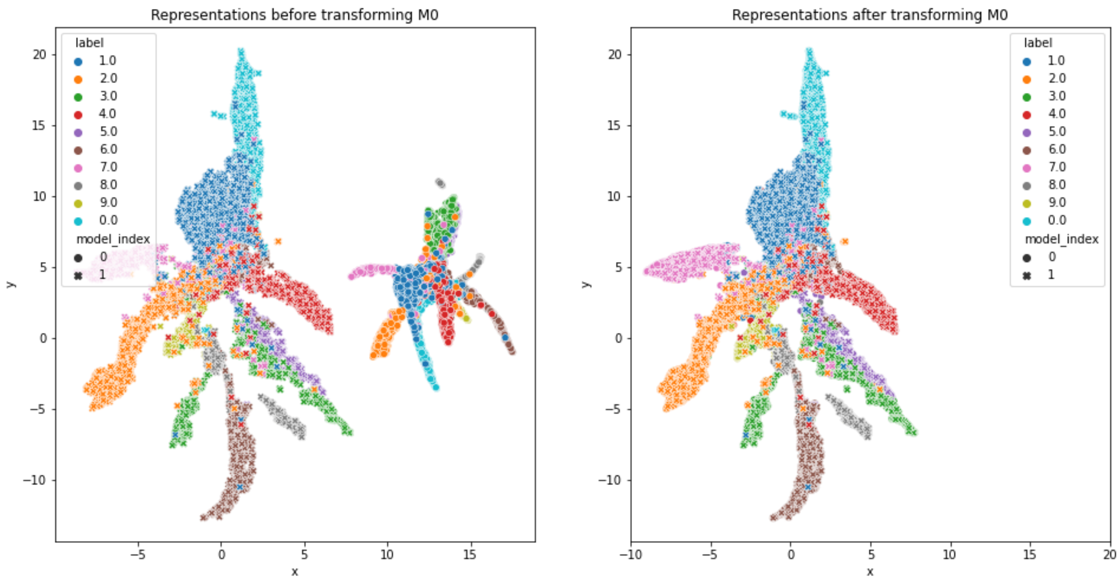


Figure 5.7: Public data representations computed by M0 and M1 before and after transforming representations of M0 on SVHN dataset

Chapter 6

Ablation study

As presented in Section 3.4, a representation transforming component is added to the main representation regularization component of the FedLinked procedure. In this section, we discuss the extent to which the addition of the representation transforming component helps to improve the performance of the participating clients. Other components could not be treated as optional for our task, therefore they can not be discussed during the ablation study.

Table 6.1 shows the achieved accuracy scores (per model and averaged scores) for both datasets, with and without applying the transforming component. Note that before generating the results, the hyperparameters of the procedure not containing the transforming component were also optimized as described in Section 4.3.

Table 6.1: Achieved accuracy scores with and without applying representation transformation

	With transformation		Without transformation	
	CIFAR10	SVHN	CIFAR10	SVHN
M0	0.76 ± 0.013	0.95 ± 0.002	0.73 ± 0.004	0.95 ± 0.004
M1	0.86 ± 0.017	0.95 ± 0.009	0.82 ± 0.008	0.95 ± 0.005
M2	0.89 ± 0.005	0.96 ± 0.007	0.86 ± 0.006	0.96 ± 0.005
AVG	0.84 ± 0.012	0.954 ± 0.006	0.80 ± 0.006	0.954 ± 0.005

The results show that for the SVHN dataset, the application of representation transformation does not improve significantly the performance of the models, which - according to our hypothesis - is due to the fact that SVHN itself is a quite easy dataset to learn in a federated environment, so the addition of a new component does not change the performance of the models in a spectacular way. In contrast, for the more difficult-to-learn CIFAR10 dataset, we observe a significant improvement, with an average accuracy increase of 3-4% per model, confirming the effectiveness of the transforming component and the need for it within the FedLinked solution-block.

Chapter 7

Conclusion

In this thesis, we proposed a new collaborative learning method called FedLinked, which utilizes unsupervised samples in order to achieve representation regularization-based collaboration of clients, which learn different tasks. The proposed solution includes several parts, which is introduced in this thesis: transformation between representations in order to enable the clients to adapt their representations to their tasks, while the server can provide it to other clients in a maximally useful way; estimation of the utility coefficients between representations of clients, which enables the solution to handle the fact that the clients learn more similar tasks can benefit from each other; and a representation regularization method, which enables the collaboration between the clients, while the privacy of their private data, architecture and model parameters are preserved. These components can also be used separately in solving other federated multitask problems too. The evaluation of our proposed solution on CIFAR10 and SVHN datasets showed that collaborative learning performed by FedLinked significantly improves the performance of the participating clients compared to separated learning. Furthermore, in multitask scenarios, FedLinked outperformed even the FedAvg-based collaborative learning method on both datasets. Finally, the projected representation vectors showed that the models trained with FedLinked are able to organize the representations of data points belonging to the same tasks into well-distinguishable, homogeneous groups, thus achieving better performance of the collaborating models. We also noticed from these representation studies that clients, which do not see any labeled sample from a task, learn representations in which these samples also form distinct groups from the others (consistently to their task). Based on our opinion, these results show that the FedLinked method can be an effective solution to the problem of federated multitask learning with the utilization of only unlabeled data for the collaboration, while the privacy of the trained models of the clients and their private data are preserved maximally.

Acknowledgements

This work was supported by the Ministry of Human Capacities under its New National Excellence Program (ÚNKP-21-2-I-BME-149).

Furthermore, I would like to thank my advisor, Dániel Hadházi, who was always willing to help me in my work, and with his unique ideas he contributed a lot to the growth in my knowledge and to the development of this research work.

List of Figures

List of Tables

Bibliography

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[2] Abdullatif Albaseer, Bekir Sait Ciftler, Mohamed Abdallah, and Ala Al-Fuqaha. Exploiting unlabeled data in smart cities using federated edge learning. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 1666–1671, 2020. DOI: `10.1109/IWCMC48107.2020.9148475`.

[3] Edwin V Bonilla, Kian Chai, and Christopher Williams. Multi-task gaussian process prediction. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008. URL `https://proceedings.neurips.cc/paper/2007/file/66368270ffd51418ec58bd793f2d9b1b-Paper.pdf`.

[4] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997. ISSN 1573-0565. DOI: `10.1023/A:1007379606734`. URL `https://doi.org/10.1023/A:1007379606734`.

[5] Michael Crawshaw. Multi-task learning with deep neural networks: A survey. *ArXiv*, abs/2009.09796, 2020.

[6] Peter Kairouz et al. Advances and open problems in federated learning. 2019. URL `https://arxiv.org/abs/1912.04977`.

[7] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction, 2019.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[9] Sohei Itahara, Takayuki Nishio, Yusuke Koda, Masahiro Morikura, and Koji Yamamoto. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *IEEE Transactions on Mobile Computing*, pages 1–1, 2021. DOI: `10.1109/TMC.2021.3070013`.

[10] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[11] Anusha Lalitha. Fully decentralized federated learning. 2018.

[12] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *ArXiv*, abs/1910.03581, 2019.

[13] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention, 2019.

[14] Zewei Long, Liwei Che, Yaqing Wang, Muchao Ye, Junyu Luo, Jinze Wu, Houping Xiao, and Fenglong Ma. Fedsemi: An adaptive federated semi-supervised learning framework, 2020.

[15] Eyal Lotan, Charlotte Tschider, Daniel K. Sodickson, Arthur L. Caplan, Mary Bruno, Ben Zhang, and Yvonne W. Lui. Medical imaging and privacy in the era of artificial intelligence: Myth, fallacy, and the future. *Journal of the American College of Radiology : JACR*, 17(9):1159–1162, Sep 2020. ISSN 1558-349X. DOI: 10.1016/j.jacr.2020.04.007. URL https://pubmed.ncbi.nlm.nih.gov/32360449.

[16] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1273–1282, 2017. URL http://proceedings.mlr.press/v54/mcmahan17a.html.

[18] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3994–4003, 2016. DOI: 10.1109/CVPR.2016.433.

[19] Yuval Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. 2011. URL http://ufldl.stanford.edu/housenumbers.

[20] Christodoulos Pappas, Dimitris Chatzopoulos, Spyros Lalis, and Manolis Vavalis. Ipls : A framework for decentralized federated learning, 2021.

[21] Micah J. Sheller, Brandon Edwards, G. Anthony Reina, Jason Martin, Sarthak Pati, Aikaterini Kotrotsou, Mikhail Milchenko, Weilin Xu, Daniel Marcus, Rivka R. Colen, and Spyridon Bakas. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific Reports*, 10(1):12598, Jul 2020. ISSN 2045-2322. DOI: 10.1038/s41598-020-69250-1. URL https://doi.org/10.1038/s41598-020-69250-1.

[22] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/6211080fa89981f66b1a0c9d55c61d0f-Paper.pdf.

[23] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/68053af2923e00204c3ca7c6a3150cf7-Paper.pdf.

[24] Jesper E. van Engelen and Holger H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, Feb 2020. ISSN 1573-0565. DOI: 10.1007/s10994-019-05855-6. URL https://doi.org/10.1007/s10994-019-05855-6.

[25] Shanfeng Wang, Qixiang Wang, and Maoguo Gong. Multi-task learning based network embedding. *Frontiers in Neuroscience*, 13, 01 2020. DOI: 10.3389/fnins.2019.01387.

[26] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, PP, 07 2017. DOI: 10.1109/TKDE.2021.3070203.