CISUC
CENTRO DE INFORMÁTICA E SISTEMAS DA UNIVERSIDADE DE COIMBRA

Federated Learning Optimization

- **Miguel Fernandes**· mfernandes@student.dei.uc.pt
- **Joel P. Arrais**· jpa@dei.uc.pt
- **Catarina Silva**· catarina@dei.uc.pt
- **Alberto Cardoso**· alberto@dei.uc.pt
- **Bernardete Ribeiro**· bribeiro@dei.uc.pt

University of Coimbra CISUC - Centro de Informática e Sistemas
FCTUC-DEI - Departamento de Engenharia Informática Coimbra, Portugal

In a recent approach defined as **Federated Learning (FL)**, a single model is shared between a server and the clients instead of the data itself, reducing the amount of data transferred. In a **FL** setting, the clients receive a shared model from the server and train it with data which is only accessible to it. Afterwards, each client sends the updated models to the server. In the server, the uploaded models are aggregated in order to form a new model.

This work proposes two new methods which outperform the state of the art of Federated Learning**: Federated Congruent Directional Learning (FedCong)** and **Federated Momentum (FedMom)**. While the first is based on the directions of the models' updates, the second algorithm is based on the momentum of the global model.

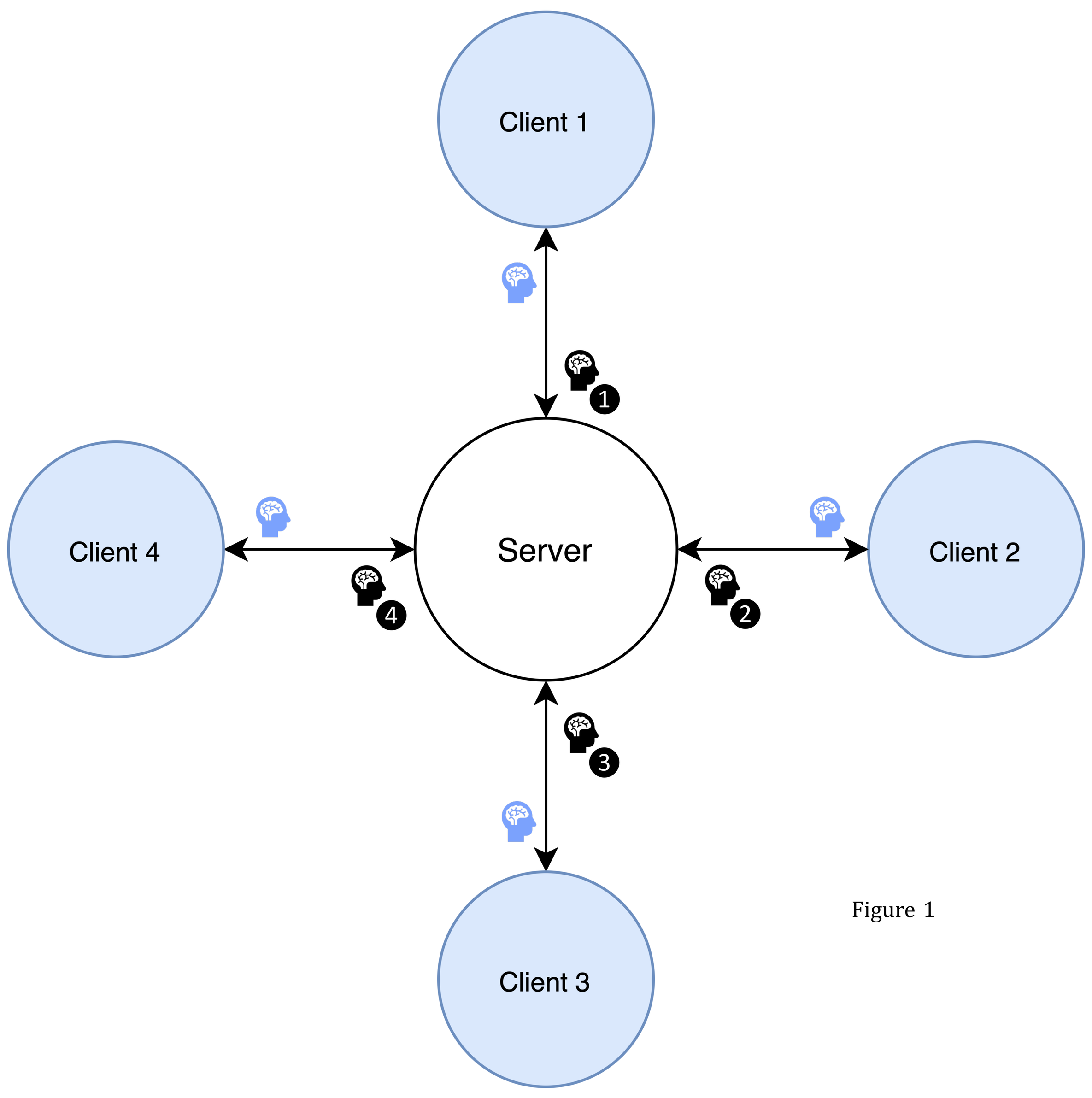

Figure 1

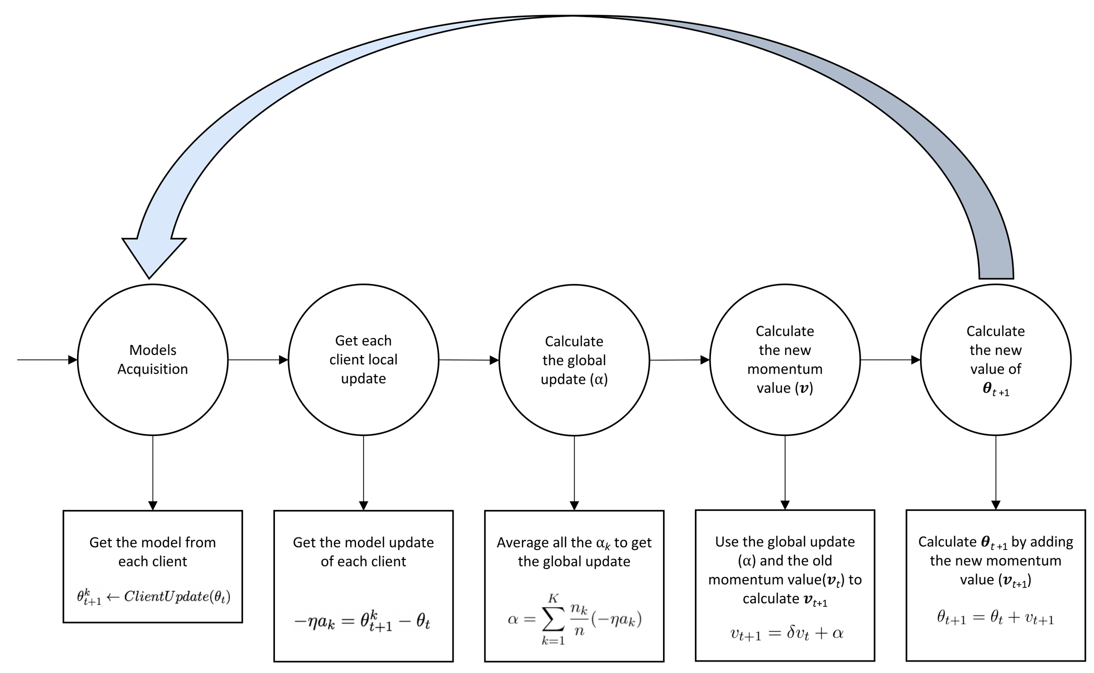

Figure 2

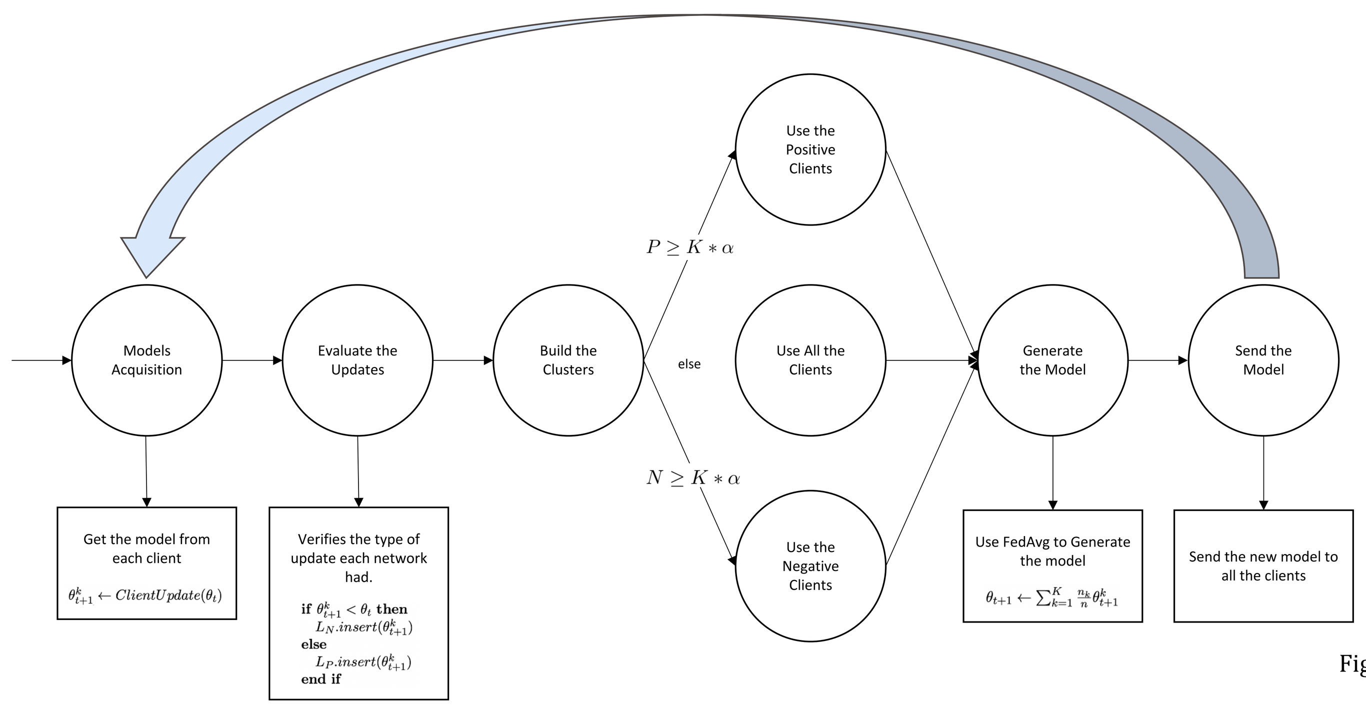

Figure 3

Objectives

The objectives of this work are:

1. To develop new **FL** algorithms which increase the convergence speed while maintaining the same performance;
2. To propose a **FL** algorithm which is based on the models' updates' directions;
3. To propose a **FL** algorithm which is based on the momentum of the global model;
4. To compare the new proposed algorithms with the state of the art of **FL**.

Contribution 1: Federated Congruent Directional Learning (FedCong)

This algorithm tries to mitigate a problem in the state of the art algorithms in **FL**, where the bigger the number of updates, the more fitted the model is to the local optimization problem, potentially causing divergence. This divergence can lead to a decay in the model's convergence speed.

The **FedCong** algorithm was developed taking these facts into consideration. In this algorithm, at the server, for each local model received, each weight update for the local problem is analysed. For each weight, **FedCong** calculates the number of positive and negative updates of all the local models.

Afterwards, the number of positive updates is compared with the number of negative updates in order to generate the new global model.

The experimental results showed that **FedCong** outperformed the state of the art of **FL** by **23%** while maintaining the same performance.

Contribution 2: Federated Momentum Learning (FedMom)

This algorithm was inspired by the momentum optimizer, having the objective of maximizing the training speed of the **FL** state of the art.

Momentum is known to help the gradient vector pointing to the right direction, damping oscillations and taking more straight forward paths to the local minimum.

The key difference in this algorithm to the state of the art is that when generating a new global model, a fraction of the update of the previous global model is used to generate it.

The experimental results showed that **FedMom** outperformed the state of the art of **FL** by **60%** while maintaining the same performance.

Figure Legend

- **Figure 1.** *Diagram of* ***FL****, where four clients participate in the model generation. The black models represent the updated models trained with the client's private data being sent to the server. The blue models represent the model generated in the server being sent to the clients.*
- **Figure 2.** *Diagram which represents the pipeline of* ***FedMom*** *where θ_t represents the global model, θ_{t+1}^k represents the k client uploaded model, n_k is the amount of data used to train θ_{t+1}^k, δ(0,1) is the momentum term and n is the sum of all the n_k.*
- **Figure 3.** *Diagram which represents the pipeline of* ***FedCong*** *where θ_t represents the global model, θ_{t+1}^k represents the k client uploaded model, n_k is the amount of data used to train θ_{t+1}^k, α(0,1) is a control parameter and n is the sum of all the n_k.*

UNIVERSIDADE DE COIMBRA · FCT Fundação para a Ciência e a Tecnologia · Cofinanciado por: COMPETE 2020 · PORTUGAL 2020 · UNIÃO EUROPEIA