UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

# Intelligence Artificielle à la Périphérie du Réseau Mobile avec Efficacité de Communication

Communication-Efficient Mobile Edge Intelligence

Thèse de doctorat
Specialité: génie électrique

Afaf Taïk

Sherbrooke (Québec) Canada

February 2022

# JURY MEMBERS

Pre. Soumaya Cherkaoui

Supervisor

Pr. Iyad Dayoub

Examiner

Pr. Jean Rouat

Examiner

Pr. Jamal Bentahar

Examiner

# RÉSUMÉ

L'intelligence artificielle (AI) et l'informatique à la périphérie du réseau (EC) ont permis de mettre en place diverses applications intelligentes incluant les maisons intelligentes, la fabrication intelligente, et les villes intelligentes. Ces progrès ont été alimentés principalement par la disponibilité d'un plus grand nombre de données, l'abondance de la puissance de calcul et les progrès de plusieurs techniques de compression. Toutefois, les principales avancées concernent le déploiement de modèles dans les dispositifs connectés. Ces modèles sont préalablement entraînés de manière centralisée. Cette prémisse exige que toutes les données générées par les dispositifs soient envoyées à un serveur centralisé, ce qui pose plusieurs problèmes de confidentialité et crée une surcharge de communication importante. Par conséquent, pour les derniers pas vers l'AI dans EC, il faut également propulser l'apprentissage des modèles ML à la périphérie du réseau. L'apprentissage fédéré (FL) est apparu comme une technique prometteuse pour l'apprentissage collaboratif de modèles ML sur des dispositifs connectés. Les dispositifs entraînent un modèle partagé sur leurs données stockées localement et ne partagent que les paramètres résultants avec une entité centralisée. Cependant, pour permettre l' utilisation de FL dans les réseaux périphériques sans fil, plusieurs défis hérités de l'AI et de EC doivent être relevés. En particulier, les défis liés à l'hétérogénéité statistique des données à travers les dispositifs ainsi que la rareté et l'hétérogénéité des ressources nécessitent une attention particulière. L'objectif de cette thèse est de proposer des moyens de relever ces défis et d'évaluer le potentiel de la FL dans de futures applications de villes intelligentes.

Dans la première partie de cette thèse, l'accent est mis sur l'incorporation des propriétés des données dans la gestion de la participation des dispositifs dans FL et de l'allocation des ressources. Nous commençons par identifier les mesures de diversité des données qui peuvent être utilisées dans différentes applications. Ensuite, nous concevons un indicateur de diversité permettant de donner plus de priorité aux clients ayant des données plus informatives. Un algorithme itératif est ensuite proposé pour sélectionner conjointement les clients et allouer les ressources de communication. Cet algorithme accélère l'apprentissage et réduit le temps et l'énergie nécessaires. De plus, l'indicateur de diversité proposé est renforcé par un système de réputation pour éviter les clients malveillants, ce qui améliore sa robustesse contre les attaques par empoisonnement des données.

Dans une deuxième partie de cette thèse, nous explorons les moyens de relever d'autres défis liés à la mobilité des clients et au changement de concept dans les distributions de données. De tels défis nécessitent de nouvelles mesures pour être traités. En conséquence, nous concevons un processus basé sur les clusters pour le FL dans les réseaux véhiculaires. Le processus proposé est basé sur la formation minutieuse de clusters pour contourner la congestion de la communication et est capable de traiter différents modèles en parallèle.

Dans la dernière partie de cette thèse, nous démontrons le potentiel de FL dans un cas d'utilisation réel impliquant la prévision à court terme de la puissance électrique dans un réseau intelligent. Nous proposons une architecture permettant l'utilisation de FL pour encourager la collaboration entre les membres de la communauté et nous montrons son importance pour l'entraînement des modèles et la réduction du coût de communication à travers des résultats numériques.

**Mots-clés :** apprentissage fédéré, apprentissage machine, communication sans fil, intelligence artificielle, informatique en périphérie, optimisation.

# ABSTRACT

Artificial intelligence (AI) and Edge computing (EC) have enabled various applications ranging from smart home, to intelligent manufacturing, and smart cities. This progress was fueled mainly by the availability of more data, abundance of computing power, and the progress of several compression techniques. However, the main advances are in relation to deploying cloud-trained machine learning (ML) models on edge devices. This premise requires that all data generated by end devices be sent to a centralized server, thus raising several privacy concerns and creating significant communication overhead. Accordingly, paving the last mile of AI on EC requires pushing the training of ML models to the edge of the network. Federated learning (FL) has emerged as a promising technique for the collaborative training of ML models on edge devices. The devices train a globally shared model on their locally stored data and only share the resulting parameters with a centralized entity. However, to enable FL in wireless edge networks, several challenges inherited from both AI and EC need to be addressed. In particular, challenges related to the statistical heterogeneity of the data across the devices alongside the scarcity and the heterogeneity of the resources require particular attention. The goal of this thesis is to propose ways to address these challenges and to evaluate the potential of FL in future applications.

In the first part of this thesis, the focus is on incorporating the data properties of FL in handling the participation and resource allocation of devices in FL. We start by identifying data diversity measures allowing us to evaluate the richness of local datasets in different applications. Then, we design a diversity indicator allowing us to give more priority to clients with more informative data. An iterative algorithm is then proposed to jointly select clients and allocate communication resources. This algorithm accelerates the training and reduces the overall needed time and energy. Furthermore, the proposed diversity indicator is reinforced with a reputation system to avoid malicious clients, thus enhancing its robustness against poisoning attacks.

In the second part of this thesis, we explore ways to tackle other challenges related to the mobility of the clients and concept-shift in data distributions. Such challenges require new measures to be handled. Accordingly, we design a cluster-based process for FL for the particular case of vehicular networks. The proposed process is based on careful cluster-formation to bypass the communication bottleneck and is able to handle different models in parallel.

In the last part of this thesis, we demonstrate the potential of FL in a real use-case involving short-term forecasting of electrical power in smart grid. We propose an architecture empowered with FL to encourage the collaboration among community members and show its importance for both training and judicious use of communication resources through numerical results.

**Keywords:** artificial intelligence, edge computing, federated learning, machine learning, optimization, wireless communication.

To my family.

# ACKNOWLEDGEMENTS

Throughout working on this thesis, I have received a great deal of support, assistance, and encouragement.

I would first like to thank my supervisor, Professor Soumaya Cherkaoui, whose expertise was invaluable to my growth as a researcher. Your insightful feedback pushed me to sharpen my thinking and your encouragement was key toward enforcing my confidence.

I would also like to thank my committee members for their insightful comments and suggestions.

I would also like to acknowledge my colleagues at INTERLAB for their wonderful collaboration and rich discussions. I would also like to thank them for their friendship and great memories inside and outside the lab.

I could not have completed this dissertation without the support of my friends who provided stimulating discussions as well as happy distractions to rest my mind and cheer me up outside of my research.

Lasly, but most importantly, I would like to thank my family for their encouragement and constant support. I would like to dedicate this thesis to them for their wise counsel and sympathetic ear. Specifically, I dedicate this thesis to my brother Abdellah and my sister Kawtar, for always being there for me and for telling me that I am awesome even when I didn't feel that way. To my father Abdelkabir, for always believing in me and always being there for me, and to my mother Latifa, my best friend and the coolest mom ever, for everything.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ACRONYMS

| **Acronyme** | **Définition** |
|---|---|
| 5G | Fifth Generation Wireless Network |
| AD | Autonomous Driving |
| AI | Artificial intelligence |
| ANN | Artificial Neural Network |
| CNN | Convolutional Neural Network |
| CVFL | Clustered Vehicular Federated Learning |
| DAS | Data-Aware Scheduling |
| DL | Deep Learning |
| DNN | Deep Neural Network |
| DQS | Data Quality based Scheduling |
| EC | Edge Computing |
| EI | Edge Intelligence |
| FL | Federated Learning |
| FEEL | Federated Edge Learning |
| HAN | Home Are Network |
| IoT | Internet of things |
| LSTM | Long short-term memory |
| MAPE | Mean absolute percentage error |
| MEC | Multi-access edge computing |
| ML | Machine learning |
| MLP | Multi-Layer Perceptron |
| OFDMA | Orthogonal frequency division multiple access |
| PCG | Prosumer Community Group |
| QoS | Quality of service |
| RB | Radio Resource Block |
| RL | Reinforcement Learning |
| RMSE | Root mean square error |
| RNN | Recurrent neural network |
| SGD | Stochastic Gradient Descent |
| STLF | Short-Term Load Forecasting |
| UE | User Equipment |

# CHAPTER 1

# INTRODUCTION

Following the exponential growth of Internet of Things (IoT) applications, the volume of data generated by connected objects requires phenomenal processing power. It is estimated [1] that by 2025 the number of IoT devices will reach 41.6 billion with data generation up to 79.4 ZettaBytes. Existing cloud computing infrastructure will not be able to cope with the generated data of these massively distributed devices. Furtheremore, sending data to the cloud over untrusted networks raises privacy concerns from the data owners and users whose behaviours are captured in the data. Users are wary of uploading their sensitive and personal data to the cloud (*e.g.,* faces, speech, location) and of how the application will use these data.

Therefore, Edge Computing (EC) [2] emerges as an attractive solution to host computation tasks as close as possible to the data sources and end users. Rather than being transmitted to a remote data center, in EC data is processed directly by the device that generates it (*e.g.,* connected object, smartphone, etc.) or by a multi-access edge computing (MEC) server. Thereby EC complements and extends the cloud functionalities.

Artificial Intelligence (AI), on the other hand, is what nurtures the growth of smart applications and analysis of data generated by IoT devices. Pushing AI to the edge of the network is seen as the best way to meet current and upcoming network service requirements, such as low latency, high bandwidth and real-time data processing. This leads to the emergence of a new field of study termed Edge Intelligence (EI) [3]. We use EI to refer to the mechanisms and approaches used to enable AI in EC, and extend this definition to include the collaboration with the cloud.

Several aspects of EI, such as using pre-trained models on edge devices (*i.e.,* on edge ML inference) [4], and using AI to enhance edge functions [5, 6], are already at production stage, owing to the improved hardware and enhanced software libraries. Training models on the other hand remains an arduous task however enticing. Federated learning (FL) [7] was recently proposed to train ML models on edge devices with the coordination of a centralized server (*e.g.,* cloud server, MEC server). Nonetheless, FL is still in its infancy, and it is facing challenges inherited from both EC and AI [4], such as the limitations of the computation and communication resources versus the resource-hungry ML algorithms.

## 1.1 Problem Statement

As described previously, EI enables the full potential of EC and allows applications to meet the quality of service requirements in terms of latency and privacy. In particular, leveraging compression and careful design techniques, several ML models are successfully deployed on edge and end devices for tasks such as facial recognition [8], keyboard prediction [9], and autonomous driving (AD) [10]. However, these models are often trained in a centralized manner either on publicly available data or through collecting the user generated data. Such training paradigm raises concerns related to privacy and communication. Instead, to unlock the full potential of EI, it is necessary to find ways to enable model training at the edge of the network. FL [7] was proposed as a promising solution for privacy preserving and communication efficient ML. FL is a ML setting that utilizes EC to tackle these concerns. In contrast to centralized ML, FL consists of training the model on the user equipment (UE)- referred to as client, device, or user, interchangeably throughout the thesis- under the orchestration of a central entity, where only the resultant parameters are sent to the MEC or cloud servers to be aggregated.

In this setting, a global model is usually initialized by a centralized entity, then an iterative training process begins. Each iteration, called communication round, starts by the selection of a subset of participating clients that receive the current version of the global model. Ideally, all the available devices should participate, but the communication bandwidth often limits the number of participants. Next, each client trains the data on its local dataset and uploads a model update to the server. The collected updates are then aggregated, typically by averaging, to create a new global model, and a new communication round starts. This process is repeated until the model converges or a maximum number of iterations is attained.

Similarly to edge inference, there are many challenges facing the efficient deployment of FL in wireless edge environment, inherited from both AI and EC.

1) The heterogeneity, scarcity, and uncertainty of computation, storage and communication capabilities across different devices sparks new system challenges. For example, significant delays can be caused by stragglers. In other cases, devices might lose their connection or run out of battery. Furthermore, the bandwidth's scarcity limits the scalability of FL for large models or large numbers of clients. Communication overhead becomes the bottleneck of FL. Additionally, mobility, especially in vehicular networks, poses new limitations to FL in applications such as autonomous vehicles and infotainment.

2) As the data collected by the clients depends on their local environment and usage pattern, both the size and the distribution of the local datasets will typically vary between different clients. This non-identically and independent distribution (non-i.i.d) nature of data across the network imposes significant statistical challenges. Furthermore, local datasets can be subject to data poisoning (*e.g.,* label-flipping attack) and concept-shift (*e.g.,* varying preferences), which requires additional measures to be integrated to the FL process.

Consequently, it is important to design resource-efficient FL algorithms that are able to handle data challenges. Accordingly, this thesis raises the following research question:

> *How to enable federated learning in wireless edge networks taking into consideration data and resource constraints?*

## 1.2 Objective

Given the multi-disciplinary nature of FL, optimizing the iterative process proves to be a challenging task. In fact, it is hard to capture both the resources problems and data challenges in a combined goal, as there is no direct relation between the learning goal (*i.e.,* model's loss function) and the resource optimization (*e.g.,* reducing required energy or time). Furthermore, the wireless edge environment imposes several constraints related to the available bandwidth and energy budget. As a result, each step of FL's iterative process can be redesigned to take these goals and constraints into account.

The main objective of this thesis is therefore to enable FL in wireless edge environments under data and resource constraints. To achieve this objective, the aim is to find ways to jointly handle these challenges. We divide the main objective into the following intermediate objectives.

- Propose a data-aware client selection and resource allocation
  The first intermediate objective is to judiciously select the subset of participating clients while taking into consideration the richness of their data. The client selection in wireless edge networks cannot be considered independently from the resource allocation. Accordingly, we raise the following research questions:
  - How to determine whether a dataset is more informative?
  - How to include the data properties as a key-consideration in the FL process?
  - How to assess the data quality under data poisoning attacks?
  - How to jointly select clients and allocate the bandwidth ?

- Extend FL to account for mobility and concept-shift

  The second intermediate objective is to enhance the FL process and extend it to handle extreme case such as concept-shift and mobility in vehicular networks. A tractable solution found to handle cluster-shift is through clustering. Yet, clustering is also used in vehicular networks through vehicle-to-vehicle communication to bypass the communication bottleneck. We aim to define possible ways to exploit clustering to handle both these extreme cases. The corresponding research questions for this intermediate objective are as follows:

  - How to form FL clusters under mobility constraints?
  - How to maintain FL clusters under concept-shift ?

- Integrate FL in a EI empowered architecture and evaluate it in a realistic scenario.

  The last intermediate objective is to demonstrate the potential of FL in a use-case and determine an appropriate mechanism to enhance its potential. We chose the electrical consumption in residential context as an interesting use case as the short-term forecasts are privacy-sensitive, but important for the electrical production and other applications in the smart grid. However, to train FL models in the smart grid context, and how to use the resulting models is an unexplored question. To the best of our knowledge, our work is the first to explore this use-case in a FL setting. The research questions for this intermediate objective are as follows:

  - How to determine which subset of clients should collaborate ?
  - What is the motivation / incentive for clients to participate in the training ?
  - How to improve the accuracy of the models given the different lifestyles of the clients ?
  - How to integrate FL with other mechanisms to build a pro-active decision mechanism ?

## 1.3 Contributions and Originality

The originality of this work rests in bridging the gap between ML related goals and EC related challenges through the direct integration of the data properties in the different steps of FL. In fact, data properties were at the heart of FL since its inception, but they have been largely overlooked in the optimization of FL process in edge networks.

### 1.3.1 Contributions

The first contribution of this thesis draws its inspiration from active learning. The premise of active learning is that models can be trained using few labelled data samples if the highly diverse data is selectively added to the training set. In FL, due to the communication constraints, only a subset of the clients can participate. However, in FL, the selection is at the dataset level. Furthermore, due to privacy constraints, the server cannot have access to the local datasets. Hence, we design a dataset diversity indicator that evaluates the richness of a local dataset without revealing sensitive details. With the purpose of enabling FL in wireless edge networks, this thesis proposes a general framework for FL that takes into consideration the statistical heterogeneity of the local datasets. This framework is then formalized to include energy and time optimization goals, and an algorithm that encloses learning and resource goals.

The second contribution of this thesis is proposing an optimized framework for FL in vehicular networks and concept-shift cases. This is achieved through the use of a carefully tailored clustering algorithm. The proposed process is based on careful cluster-head selection and matching algorithms, which utilize vehicle-to-vehicle communication to bypass the communication bottleneck and is able to handle different models in parallel. Such algorithm will enable FL for smart transportation systems by providing a more secure alternative than communication intensive data-sharing and protecting privacy-sensitive localization data.

The last part of this thesis consists on defining an enabling architecture and process in a smart city level application, namely smart grid. More specifically, to the best of our knowledge, our work is the first to explore and demonstrate the potential of FL in electrical load forecasting, which is a key operation for energy production and trading.

In the following, we summarize the main contributions of this thesis.

- We propose a general framework for incorporating data properties in FEEL, by providing a guideline for a thorough algorithm design, and criteria for the choice of diversity measures in both datasets and models.
- we present several possible measures and techniques to evaluate data and model diversity, which can be applied in different scenarios (e.g., classification, time series forecasting), in an effort to assist fellow researchers to further address FEEL challenges.
- we design a suitable diversity indicator, which serves as a priority criterion for the selection of devices;
- we formulate a joint device selection and bandwidth allocation problem taking into account data diversity, and we propose a data-aware scheduling algorithm based on an iterative decomposition technique to solve it;

- we evaluate the diversity indicator and the data-aware algorithm through extensive simulations;
- we design a reputation system to handle outliers and combine it with the diversity indicator;
- we evaluate the importance of each component in client selection under different data-poisoning attack scenarios.

In order to extend the work to vehicular networks and concept-shift cases, we leverage the vehicle-to-vehicle (V2V) communication to bypass the communication bottleneck. Hence, our contributions are as follows:

- we design an architecture and corresponding FL process for clustered FL in vehicular environments;
- we formulate a joint cluster-head selection and resource block allocation problem taking into account mobility and data diversity properties;
- we formulate a matching problem for cluster formation taking into account mobility and model preferences;
- we evaluate the proposed scheme through extensive simulations.

Finally, since most of the work in FL is evaluated using simulated distributions of datasets such as CIFAR-10 and MNIST which does not give realistic expectations for real deployments, we evaluated FL on a real-dataset of electrical consumption and production and proposed a process to integrate FL in a collaborative process. In fact, residential electrical consumption data reveals sensitive details about the habits of the residents. At the same time, forecasting the consumption and production of electricity is important for saving energy and other applications in the smart grid such as demand-response. Accordingly, we propose a EI framework leveraging a centralized aggregator to coordinate the energy trading in prosumer (*i.e.,* producers and consumers) community groups (PCGs). The proposed framework demonstrates the potential of EI in smart grid through the use of EC and FL. The contributions in this part of the thesis are as follows:

- we discuss key elements for 5G empowered energy markets and highlight challenges related to their design,
- we design a multi-stage energy forecasting framework and a decision process for PCGs empowered with FL using edge equipment,
- we use personalization to enhance the quality of the resulting models, and
- using real datasets, we evaluate the accuracy of load forecasts and the potential network load gain through simulations.

### 1.3.2 List of Papers

The research conducted during this doctoral project resulted in high-quality papers, which are published and submitted to renowned and highly respected conferences and journals. The list of the papers is as follows:

1 **A. Taïk** and S. Cherkaoui, "Federated Edge Learning: Design Issues and Challenges", in IEEE Network, vol. 35, no. 2, pp. 252-258, March/April 2021, doi: 10.1109/MNET.011.2000478. **(Chapter 3)**

2 **A. Taïk**, Z.Mlika and S. Cherkaoui, "Data-Aware Device Scheduling for Federated Edge Learning" IEEE Transactions on Cognitive Communications and Networking, vol. 8, no. 1, pp. 408-421, March 2022, doi: 10.1109/TCCN.2021.3100574. **(Chapter 4)** [1]

5 **A. Taïk**, H.Moudoud, and S. Cherkaoui "Data-quality based device Scheduling for Federated Edge Learning", IEEE Local Computer Networks Conference (LCN) 2021, pp. 17-23, doi: 10.1109/LCN52139.2021.9524974. **(Chapter 5)** [2]

4 **A. Taïk**, Z.Mlika, and S. Cherkaoui "Clustered Vehicular Federated Learning: Process and Optimization", in IEEE Transactions on Intelligent Transportation Systems, [Early access], doi: 10.1109/TITS.2022.3149860.**(Chapter 6)** [3]

5 **A. Taïk** and S. Cherkaoui,"Electrical Load Forecasting Using Edge Computing and Federated Learning", in 2020 IEEE International Conference on Communications (ICC), Jun. 2020, pp. 1–6, doi: 10.1109/ICC40277.2020.9148937 [4]

6 **A. Taïk**, B.Nour and S. Cherkaoui,"Empowering Prosumer Communities in Smart Grid with Wireless Communications and Federated Edge Learning", IEEE Wireless Communications 2021, vol. 28, no. 6, pp. 26-33, December 2021,
doi: 10.1109/MWC.017.2100187. **(Chapter 7)** [5]

## 1.4 Thesis Plan

This thesis is based on manuscripts (Article-based). Chapter 2 conducts a literature review describing recent advances related to our research project. Chapter 3, chapter 4, and chapter 5 present our contributions related to joint client selection and resource allocation with a focus on data aspects. In chapter 3, we propose a general design framework for

1. https://github.com/afaf-taik/Data-aware-FL
2. https://github.com/afaf-taik/ReputationFL
3. https://github.com/afaf-taik/vehicularFL
4. https://github.com/afaf-taik/Smart-Grid-FL
5. https://github.com/afaf-taik/Smart-Grid-FL

data-aware FL, listing several possible measures to evaluate the data diversity. In chapter 4, we formulate an optimization problem to select the clients with potentially more informative datasets while optimizing the overall time and energy required for the training, and we develop an iterative algorithm to solve the optimization problem. In chapter 5, we extend the client selection indicator to take into account outliers and malicious devices by implementing a reputation system. In chapter 6, we broaden the horizon of tackled challenges to consider mobility in vehicular networks, and concept-shift by proposing an optimized clustered vehicular FL process. In chapter 7, we integrate FL in a collaborative EI empowered energy trading decision process, and we evaluate the efficiency of FL in the use-case of smart-grid using real time series data.

# CHAPTER 2

# State of the Art

EI is an emerging paradigm meeting the challenging requirements of future smart services scenarios and applications where latency and privacy are of upmost importance. EI mainly consists of training and inference of different AI models at the edge of the network. While on-device and edge inference have reached production-level, training models at the edge remains challenging. Federated learning (FL) is the arch end of EI offering the possibility to train several ML models on user equipment (UE), thus aligning with stringent privacy requirements of several AI-based applications. However, FL is still in its infancy and faces several challenges related to both AI and EC. In this chapter, we present a literature review of recent advances related to this research project. We start by providing, in section 2.1, an overview on AI on edge, and some key definitions. In section 2.2, we introduce FL and the challenges surrounding its adoption in wireless edge networks. More specifically, we identify two categories of challenges : resources and data. In section 2.3, we review recent approaches proposed to address the challenges related to data in FL and discuss their advantages and shortcomings. In section 2.4, we discuss the recent work focusing on challenges related to resources in FL. Finally, in section 2.5, we conclude the state of the art and introduce the contributions of this thesis.

## 2.1 EC and AI : Preliminaries and definitions

The recent progress in deep learning (DL) have paved the way for large scale adoption of AI applications and services, spanning from home assistants to caching and recommendation systems. More recently, with the proliferation of mobile computing and the Internet of Things (IoT), data generated by widely distributed mobile and IoT devices require phenomenal processing power. Existing cloud computing infrastructure will be soon unable to keep up with these massively distributed devices and analyze their data. Inevitably, there will be a pressing need to push the AI algorithms to the network's edge so as to deliver the full potential of AI applications in IoT. To resolve this challenge, EC was proposed as an emerging paradigm consisting of pushing computing tasks and services from the network core to a vicinity near the devices that generated the data. The new inter-discipline AI on edge, is beginning to receive a tremendous amount of attention. In the following, we define its various components, starting with an overview on EC, then presenting preliminary definitions of AI algorithms.

### 2.1.1 Edge computing/ Multi-Access Edge Computing

The exponential growth of IoT deployment, and the emergence of new applications with stringent latency and privacy requirements, have paved the way to a wide adoption of EC as a popular and innovative technology. EC refers to placing computing, storage and network capabilities at the network's edge, close to the end users. The physical proximity to the data generators is the most emphasized characteristic of EC. Such proximity insures low latency response and data privacy, making EC an important complement to the cloud, and even a replacement for it in some scenarios [2, 11, 12]. Indeed, being in the vicinity of end-devices can benefit many use cases where latency and privacy are of foremost importance, such as industrial IoT [13] autonomous driving (AD) [14], and e-health [15].

Various new technologies are designed to work at the edge of the network, such as Micro Data Centers (MDCs) [16], Fog Computing [17] and Multi-Access Edge Computing (MEC) [18]. Several technologies are identified as enabling technologies for EC realization, such as Network Function Virtualization (NFV) [19] and Software Defined Networking (SDN) [20]. NFV aims to abstract and virtualize the network functions addressing flexibility and agility requirements of EC. On the other hand, SDN provides 5G networks with programmability, automation and centralization of control functions for the traffic flows. NFV/SDN can enhance the interoperability of the EC infrastructure. For example, Edge Analytics services can be hosted on a NFV framework as virtual network functions [21]. SDN/NFV can also allow edge nodes to be efficiently orchestrated and integrated with cloud data centers [22].

The EC layer is the intermediate layer between the users (end devices) and the Cloud data centers. Fig 2.1 illustrates the different components of an EC enabled architecture. The EC network comprises both Edge and end nodes. End nodes are often heterogeneous devices, from different manufacturers, with limited capacities. The end nodes reside at the bottom of the architecture. They are essentially smart devices that generate data and/or consume smart services. The upper layer is the cloud computing layer, it is dedicated to big data analytics and permanent storage facilities.
The edge layer is formed by various networking components like edge routers, Base Stations (BSs), and switches, alongside with MEC servers and MDCs. These components act as a single processing, storage or networking unit. They provide real-time data processing and serve as a caching facility. As a result, EC is one of the 5G enabling pillars by virtue of its advantages of reducing data transmission, improving service latency and easing cloud computing pressure.

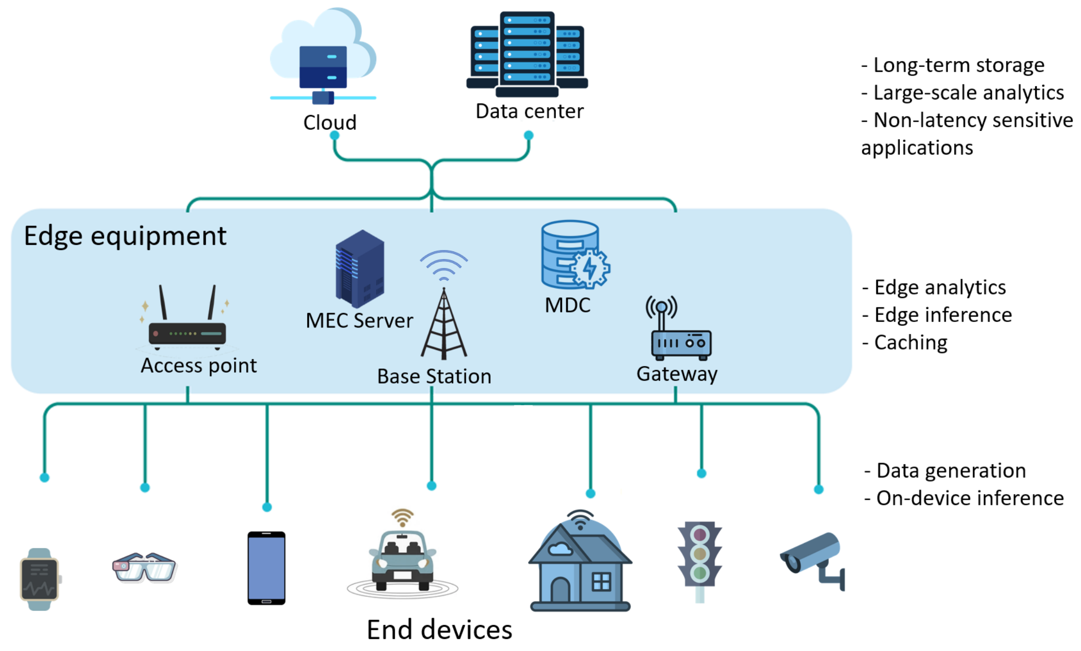


Figure 2.1 EC enabled architecture

### 2.1.2 Artificial Intelligence

AI refers to the set of techniques and approaches aimed at building intelligent machines capable of performing tasks requiring human-level intelligence such as planning, learning, reasoning, problem solving, and even some social intelligence and creativity. AI is essential for enabling quick analysis of huge volumes of data and extracting insights, which leads to a strong demand to integrate EC and AI. This integration leads to the emergence of Edge Intelligence (EI) [23]. EI refers to pushing AI to the edge of the network. This definition is not restricted to running AI models on edge servers/devices, but expanded to include the collaboration of edge and cloud [3], and using AI for EC. By adapting models to resource-restricted devices, EI will unleash the full potential of EC.

An AI system combines machine learning (ML) algorithms alongside other data analysis methods to achieve intelligent capabilities. ML refers to a set of techniques that enable a system to learn from data or past experiences without explicit programming. ML encompasses a wide range of algorithms including supervised learning (*e.g.,* linear regression, decision trees), unsupervised learning (*e.g.,* k-means), and reinforcement learning (RL). Briefly, supervised learning requires training with labelled data comprising inputs and desired outputs. In contrast with the supervised learning, unsupervised learning does not require labeled training data and the environment only provides inputs without desired targets. RL enables learning from feedback received through interactions with an external environment.

In ML, a model refers to the mathematical expression of a real-world process learned through a learning algorithm that finds patterns in the training data, by aiming to minimize the error function often termed loss function. A model is first trained then used to inference useful information from raw/test data. In the following, we present several definitions related to ML that are used throughout this thesis.

**a) Loss function:** ML models include a set of parameters which are learned based on training data. A training data sample $j$ usually consists of two parts for supervised learning models. The first part is the input denoted $x_j$ (*e.g.,* the pixels of an image, previous steps in time series) and the other denoted $y_j$ is the desired output of the model (*e.g.,* the label of the image, the value of the next step in a time series). To enable the learning, each model has a loss function defined on its parameter vector $w$ for each data sample $j$. The loss function captures the error of the model on the training data, and the model learning process aims to minimize the loss function on a set of training data samples. For each data sample $j$, we define the loss function as $f(w, x_j, y_j)$. For some unsupervised models such as K-means, the training data is composed of only $x_j$, and accordingly, the loss function value only depends on $x_j$. Table. 2.1 summarizes some examples of ML models and their corresponding loss functions, with $\parallel . \parallel$ denoting the $L_2$ norm, and $x^T$ the transpose of $x$.

**b) Stochastic Gradient Descent** Stochastic gradient descent (SGD) is an iterative method for optimizing an objective function with smoothing properties (*e.g.,* differentiable) [24]. SGD is used in high-dimensional optimization problems, as it reduces the computational load compared to gradient descent. This is achieved by replacing the actual gradient, which is computed from the entire data set in gradient descent, by an estimate of it computed from a randomly chosen subset of the data. After initializing the model and choosing a learning rate, a model $w$ in step $t+1$ is computed either on a single

| Model | Loss Function $f(w, x_j, y_j)$ |
|---|---|
| Linear Regression | $\frac{1}{2}\left\| y_j - w^T x_j \right\|^2$ |
| K-means | $\frac{1}{2} min_l \left\| x_{(l)} - w_{(l)} \right\|^2$ where $w \overset{\text{def}}{=} \{w^T_{(1)}, ...\}$ |
| Squared-SVM | $\lambda \left\| w \right\|^2 + \frac{1}{2} max0; 1 - y_j w^T x_j{}^2$ ($\lambda$ *is a constant*) |

Table 2.1 Loss functions for some ML models

sample, or on a mini-batch as follows:

$$w(t+1) = w(t) - \eta \nabla f(w(t)) \tag{2.1}$$

Other variations and improvements of the SGD were proposed, such as adding momentum, Adagrad [25] and Adam [26]. SGD is widely used for training several models such as SVMs and logistic regression. Moreover, combined with the back-propagation algorithm, it is is also used for training artificial neural networks (ANNs).

**c) Artificial neural networks:** The latest rise in AI was in the past decade and was partially due to breakthroughs achieved in deep learning (DL), a subset of ML algorithms, where the new advances achieved human-level accuracy in some areas and even surpassed humans in some applications [27], including computer vision [28] and natural language processing [29, 30]. DL models consist of various types of Deep Neural Networks (DNNs). In short, a DNN is an ANN that has multiple hidden layers. Many of the work realized in the thesis and its related work use or are related to ANN models in general and DNNs for supervised learning in particular, therefore, we provide a brief background on these models.

As depicted in Fig 2.2, each layer of a DNN is composed of neurons that generate linear or non-linear outputs. Each neuron has a vector of weights associated with the input data size of the layer.

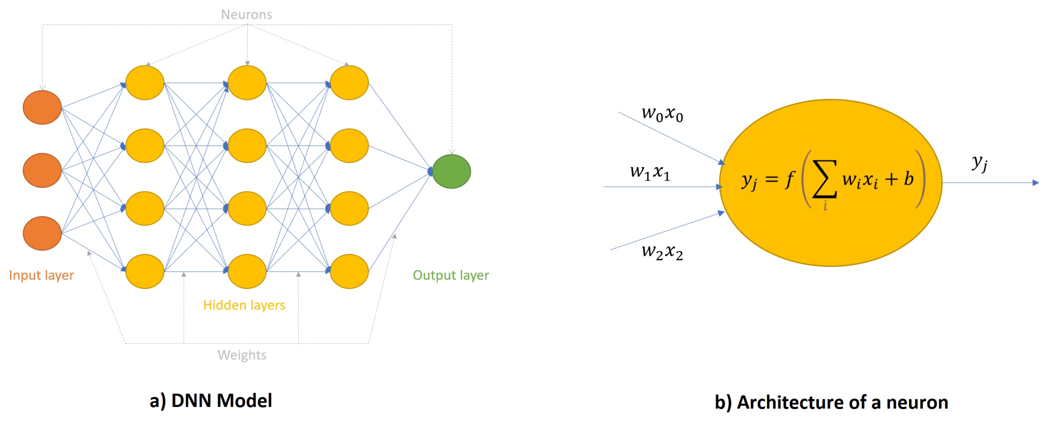


Figure 2.2 DNN and neuron structures

In inference, the input data is propagated through the layers sequentially, where each layer performs matrix multiplications on it. The output of a layer is -usually- the input to the next layer. Once the data has been processed by the final layer, the output is provided at the output layer.This process is called feed-forward pass. Depending on the problem, an output can take different forms such as a numerical value, a vector, or a matrix.

Convolutional neural networks (CNNs) are a special case of DNNs where the matrix multiplications include convolutional filter and pooling operations, which are essential to capturing the high-level representation of the input data. They are mostly used in computer vision tasks [31, 32]. For example, provided a picture of an animal, the CNN will be used to analyze the pixels to classify the animal. Recurrent neural networks (RNNs) are another type of ANNs. They are used to process sequential input data, such as time series and text. RNNs are widely used in forecasting tasks and natural language processing [29, 30]. Some of the most used RNN neurons are Long-short term memory (LSTM) and gated-recurrent units (GRU).

In supervised ANN training, the initial values of weights and biases in the model are often randomly generated. As it is the case for other ML models, The goal of ANN training is to optimize these weights and biases. We feed samples of data to the network, and after the feed-forward pass, the output of the last layer is compared to the ground truth via a loss function (*e.g.,* root mean squared error (RMSE), mean average percentage error (MAPE) ). To adjust the weights of each neuron in the model, SGD or a variant, is used and the gradient of the loss function is calculated. Using the Back Propagation mechanism [33], the error is propagated back across the layers of the ANN. The weights are then updated based on the gradient and the learning rate. By feeding a large set of training samples and repeating this process for a few epochs (*i.e.,* passages through the entire training dataset) until the training loss is below a predefined threshold, or the accuracy reaches the desired level, the final model is obtained. The DL training procedure consists of the feed-forward process and the back-propagation process while the inference involves the feed-forward process only. This makes the training more resource-consuming, especially that it is repeated for multiple iterations. As a result, edge inference is in production stages for several applications, while edge training is still at its infancy.

## 2.2 Federated Learning : Overview and Challenges

Earlier work in EI focused on deploying pre-trained models at the edge for inference. Nevertheless, privacy issues and communication overhead generated a pressing need for pushing the training task to the edge. FL emerged as an attractive solution for training models at the edge. FL was first proposed by Google and deployed on android phones for keyboard prediction [9]. The premise of FL is to keep the data at each device and train a shared global model across the federation of distributed connected devices by only sharing model updates with a cloud server for aggregation. By keeping data at the edge, FL prevents data leakage, reduces bandwidth usage, and benefits from rapid access to ephemeral data. The application of FL in wireless edge networks forms the so-called concept of FEderated Edge Learning (FEEL). In FEEL, edge devices collaboratively train models and send updates to a MEC server for aggregation. The MEC server is equipped with a parameter server (PS), and can be a next generation nodeB (gNB), or a base station (BS). In this section, we introduce a general template for FEEL and we present the different challenges related to it in terms of data and resources, key performance indicators, alongside current and future applications.

### 2.2.1 Learning Problem

Let us consider a wireless edge cellular network with $K$ edge devices. Each device $k$ has a local dataset $D_k$ with a data size of $|D_k|$. For each device $k$ the loss function on the dataset $D_k$ is

$$f_k(w) = \frac{1}{|D_k|} \sum_{j \in D_k} f_j(w) \tag{2.2}$$

Assuming that $D_k \cap D_{k'} = \oslash$ for $k \neq k'$ , the goal is to find the optimal global model parameters $\boldsymbol{w} \in \mathbb{R}^l$ that minimizes the average prediction loss $f(\boldsymbol{w})$:

$$\min_{\boldsymbol{w} \in \mathbb{R}^l} f(\boldsymbol{w}) = \frac{1}{D} \sum_{j \in \bigcup_j D_j} f_j(\boldsymbol{w}) = \frac{1}{D} \sum_{k=1}^{K} |D_k| \, f_k(\boldsymbol{w}), \tag{2.3}$$

where $\boldsymbol{w}$ is the model parameter vector to be optimized with dimension $l$, $f_k(\boldsymbol{w})$ is the loss value function computed by device $k$ based on its local training data, and $D$ is the total number of data points across all devices (i.e., $D = \sum_{k=1}^{N} |D_k|$). Due to the inherent complexity of most ML models, it is usually impossible to find a closed-form solution to 2.3. Thus it is often

### 2.2.2 FEEL template

Ideally, all the devices train their local models using their local training data and upload their gradient or model updates to the server for aggregation. Then, the server sends the new global model to the edge devices starting a new iteration termed communication round. Nonetheless, the constrained edge resources and limited communication bandwidth in wireless (edge) networks result in significant challenges and imposes new considerations for FEEL in contrast to centralized and distributed learning paradigms. For instance, only a subset of devices are available and suitable for training at once. Moreover, the aggregation step needs to take into consideration the statistical heterogeneity of the local datasets. A template for FEEL training can be considered as follows [34]:

1. *Client Selection:* The server selects a subset of devices meeting training requirements. For instance, only devices that have enough battery, good reception and enough data samples could be selected for training.
2. *Global Model Broadcast:* The selected clients receive the current model weights alongside other hyperparameters (*e.g.,* number of local epochs, optimizer) from the MEC server.
3. *Local Training:* Each selected device locally trains the model.
4. *Models Upload:* The MEC server collects the device updates.
5. *Model Update:* The server updates the global model using the collected updates typically by averaging.

Each of these steps should be carefully optimized for a seamless deployment in wireless edge networks. Having defined a general framework for FEEL, we next discuss the different challenges that need to be considered.

### 2.2.3 Challenges

Compared to similar problems such as distributed learning in data centers, the two challenging aspects of FEEL settings are resources and data.

**Resources:** The first set of challenges that should be considered are related to computation, storage and communication resources. These resources are usually heterogeneous, scarce, and subject to uncertainties.

– *Resources heterogeneity:* In most use-cases of FEEL, the computation, storage and communication capabilities vary from a device to another. For instance, devices, even from the same manufacturer, will often be equipped with different hardware (*e.g.,* CPU, GPU, AI chips), network connectivity (*e.g.,* 4G/5G, Wi-Fi), and may differ in available power (*e.g.,* plugged-in, unplugged) [35]. For synchronous updates aggregation, which is the most adopted aggregation method [36], the gaps in computational and communication resources introduce challenges such as significant delays caused by stragglers. FEEL algorithms should therefore aim to be adaptive to the heterogeneous hardware during client selection and updates' collection.

– *Limited Resources:* Unlike cloud servers, the computing and storage resources of the devices are very limited. Therefore the models that can be trained on device are relatively simpler and smaller than the models trained on the cloud [37, 38, 39]. Furthermore, devices are frequently offline or unavailable either due to low battery levels, or because their resources are fully or partially used by other applications. As for the communication resources, the available bandwidth is limited. It is therefore important to develop communication-efficient methods that allow to send compressed [40] or partial model updates[41].

- *Mobility:* FL is considered a key enabler toward Intelligent transportation systems [42]. It was investigated and evaluated for scenarios involving vehicle management in AD, Infotainment and route planning. In these scenarios, the communication challenges are exacerbated because of the high mobility in vehicular ad-hoc networks (VANETs). Particularly, the communication channel states in mobile and vehicular environments are subject to rapid changes, which leads to uploading partial participation and frequent disconnections.

**Data:** Unlike distributed learning in data centers where the data distributions can be controlled and identically and independently distributed (i.i.d), data distributions in FEEL contexts are wedged by the users' environments and behaviours [43, 44]. For this reason, the local datasets for most use-cases are massively distributed, statistically heterogeneous (i.e., non-i.i.d, unbalanced, subject to concept-shift), and highly redundant. Additionally, the raw generated data is often privacy-sensitive as it can reveal personal and confidential information. Furthermore, local data quality cannot be verified, meaning local datasets can be subject to poisoning attacks.

- *Privacy:* When FL was first coined, privacy was among the first aspect considered in the design. FEEL aims to protect the raw data generated on each device by only sharing model updates with the server. However, recent work has shown that model updates can be reverse-engineered to reveal sensitive information, either by a third-party or a malicious central server [45]. FEEL design therefore requires additional mechanisms to guarantee data privacy [46].
- *Small and widely distributed datasets:* In FEEL scenarios, the users are usually dispersed over large geographic areas and the number of the local samples is limited by the devices' memory. A large number of devices participate in training with a small average number of data samples per client. Training models on small numbers of samples makes them prone to overfitting. Additionally, depending on the MEC defined requirements, many devices may not own enough data for training.
- *Unbalance:* In several FEEL scenarios, the size of the generated data depends on the user's behaviour and device usage. Moreover, the number of training samples the device's storage capacity. The large gaps in the training data sizes imposes new considerations regarding the training deadline and the number of local epochs that can be executed on each device.

- *Non-i.i.d data:* The training data on a given device is typically based on the usage of the device by a particular user, and hence any particular user's local dataset will not be representative of the population distribution. This data-generation paradigm fails differs from the i.i.d assumptions usually found in distributed learning, hence adding complexity to the FEEL performance evaluation and convergence analysis.
- *Redundancy:* The unbalance of the data can be also observed within the local training samples of a single device. In fact, IoT data is often highly redundant. For instance, in sequential data such as video surveillance and sensors data, only a subset of the data contains new information or events.
- *Concept-shift and data poisoning:* In use-cases where the user preferences are in-play, for example dating application and content streaming platforms, resulting in concept-shift. Concept-shift refers to when data with same feature have different labels between clients, or when different features share the same label. However, this same kind of shift can be viewed differently in other cases. Particularly, in some security applications where some behaviours should be labelled as malicious, a malicious client may label these behaviours as normal thus launching a label-flipping attack. Taking into consideration the particularities of the labels in different scenarios is essential to guarantee the training convergence.
- *Artificial datasets vs real use case data:* There exists a large gap between artificial datasets that are popular and accessible for benchmarking, and datasets that realistically capture the characteristics of a federated scenario. In fact, most existing work use artificial distributions of artificial datasets to simulate non-i.i.d, unbalanced, and concept-shift scenarios. While this allows for reproducibility and accessibility of data, they do not reflect challenges that are often found in real use-cases such as missing and erroneous data [47].

These challenges require re-designing and rethinking the different steps of FEEL. Taking into account possible issues with data and resources constraints, proposed techniques often will aim to accelerate the training and reduce its cost. However, these two goals are often in conflict. Depending on the use-case, the performance of a FEEL method or algorithm should be evaluated based using predetermined indicators.

### 2.2.4 Key Performance Indicators

In order to adequately assess the performance of a FEEL method or technique, we identify key performance indicators that should be evaluated in FEEL environments. More specifically, we pinpoint three general axis of evaluation : Time, energy, and learning, and for each of these axis, we detail key performance indicators.

**Time:** Time in FEEL can be evaluated through different angles: communication round duration, total time or required number of communication rounds until model convergence. For synchronous updates aggregation, the total duration of a round is determined by the slowest device among all the selected devices [48]. Bandwidth and power allocation in the model upload step are crucial when the round's deadline is not pre-determined. For instance, more bandwidth could be allocated to stragglers and less for faster devices. Another aspect than can be evaluated is the time/number of rounds until convergence. The optimization techniques centered on this aspect mainly focus either on the selective upload of updates [49], or on maximizing the participating devices in each round [50].

**Energy efficiency:** Optimizing the energy consumption across the network is necessary to reduce the rate of drop-out devices because of battery drainage. In fact, training and transmission of large-scale models are energy consuming, while most edge and end devices have limited battery lives. Additionally, using the maximum capacity of the devices would make the users less-likely willing to participate in the training. A design goal of a scheduling algorithm (*i.e.,* joint selection and resource allocation) would be to allocate bandwidth based on the devices' channel states and battery levels. As a result, more bandwidth should be allocated to devices with weaker channels or poorer power states, to maximize the collected updates [50].

**Learning performance:** The goal of several training processes is to find the parameters that minimize a loss function across the population of the devices. However, the loss function is not enough to evaluate the performance of the model. Instead, the model is usually tested on a test-set, which can either be stored at the server through publicly available data, or distributed across the devices. In order to evaluate the learning performance, several metrics can be used throughout the training. For instance, in classification models, we can use accuracy, which refers to the ratio of the number of the input samples that get the correct predictions from inference to the total number of input samples. In regression models, the performance can be reflected by other metrics such as RMSE and MAPE, where the RMSE is often in the same unit as the predicted value, and the MAPE reflects the percentage that the error represents compared to the real value. In contrast to centralized learning, optimizing the learning in the FEEL setting cannot be seen independently from time and energy optimization.

### 2.2.5 FEEL applications

FEEL will enable a wide range of IoT services and applications.

#### a) IoT services

FEEL was proposed as a key-enabler for several IoT services, such as IoT data sharing, caching, and mobile crowdsensing. Data sharing consists of sending data over a shared network to serve end users in a specific application. Instead of sharing the raw IoT data, FEEL offers an alternative of sharing learning results to enable intelligent IoT networks with low latency and privacy preservation [51]. Using FEEL as an alternative was proposed for industrial IoT [52] and vehicular network scenarios [53, 54]. Moreover, work in [55, 56] proposed to build proactive data caching schemes without directly accessing user data to predict the most popular files for caching.

#### b) IoT Applications

Leveraging several IoT services, FEEL has the potential to be the backbone for several IoT applications such as smart health-care, smart transportation, smart cities, and smart industry.

Compared to other domains, healthcare date are highly sensitive and often subject to regulations such as United States Health Insurance Portability and Accountability Act (HIPPA) [57]. Anonymization techniques such as the removal the metadata are insufficient to preserve privacy of patients. FL in general and FEEL in particular were used for medical imaging processing [58, 59] and arrhythmia detection [60]. Smart health applications focus on reinforcing FL with privacy techniques such as differential privacy [61].

Smart transportation applications such as AD [62] and traffic prediction tasks [63]. AD in particular relies heavily on data-sharing, thus creating several concerns regarding privacy and communication overhead where FL becomes an attractive alternative. Nonetheless, challenges related to mobility are understudied in the case of vehicular networks.

Smart city applications such as smart surveillance [64] and smart grid use several privacy sensitive data in their classification and prediction models. More specifically, smart surveillance can make individuals subject to stalking and harassment. Additionally, smart-grid data reveal the habits of house residents, household occupancy, and even traces of some appliances. Such information can be maliciously used for burglary and targeted advertisement. Nonetheless, the prediction models are still required to enable the potential of these applications, which makes FEEL a promising alternative. Nonetheless, experimental studies in this area are still lacking.

## 2.3 Tackling data related challenges

To tackle challenges related to data, several adjustments, techniques, and algorithms, were proposed. These efforts focused on one or several steps of the FEEL algorithms, spanning from client selection to updates averaging. In this section, we first focus on data distribution skew (non-i.i.d and unbalanced aspects), then we discuss efforts toward handling concept-shift and poisoning attacks.

### 2.3.1 Non-i.i.d and unbalanced data

In FEEL, the client represents the data owner. Each client has full autonomy for the local data and it often reflects the end user's behaviour. The non-i.i.d data is among the most common features of data in FEEL and distinguishes it from similar settings such as distributed training. In classification problems, the non-i.i.d aspect is often reflected in cases where the input features or data labels are not evenly distributed between clients. Some works [65, 66] proposed to distribute publicly available data or data shared by some clients to overcome this challenge. However, it is unrealistic to deploy this solution as data might not always be publicly available and clients might not be willing to share sensitive data. Instead, optimizing the different steps in FEEL's iterative process remain more tractable solutions.

**Client Selection:** In FEEL, the number of available clients could be sufficiently large, but the bandwidth available for model broadcast and updates' upload is rather limited, making it more practical to only involve a subset of the devices in the training. Under data distribution skew, the client selection policy becomes critical in terms of training efficiency, and highly affects the final model's quality. Early work focusing on client selection often aimed to maximize the number of selected clients [50, 67]. In [67], the client selection algorithm gives priority to the end devices with good communication and computation capabilities. Authors in [50] propose an energy-efficient algorithm that ensures the training speed by collecting the maximum amount of updates possible. These algorithms do not perform well in skewed data distributions, as they are biased toward powerful devices with better channel states. Such bias will likely lead to models that cannot generalize to a wide population of devices, as it does not guarantee the diversity of updates sources. Consequently, fairness measures [68, 69, 70] were adopted to ensure gradient diversity. In [69, 70], an age-based scheduling algorithm was proposed, where higher priority is given to devices that were not selected in several previous rounds. However, relying on fairness only might affect the number of collected updates within a round. In [68], authors focus on long-term fairness, where the client selection is defined as an online Lyapunov optimization problem and a long-term guarantee of client participating rate is quantified using dynamic queues.

**Updates collection:** *"Not all updates are significant"* is the intuition on which several works are based. For instance, Gaia [71] determines which local updates should not be transferred based on their magnitude compared to the global model. Updates where $\|\frac{Update}{Model}\| < Threshold$ are considered insignificant and thus are not uploaded. However, this method cannot be applied for non-i.i.d distributions.Furthermore, the value of this significant measure decreases over the iterations which makes finding a global threshold difficult. Accordingly, Communication-Mitigated Federated Learning (CMFL) [49] proposed a significance measure based on the percentage of same-sign parameters in the two updates. This approach identifies whether the client-side updates follow the collaborative direction and excludes outliers. Evaluating the updates' significance might help mitigate the communication overhead and improve the learning efficiency by excluding devices with redundant data and outliers. However, this method is not suitable in energy and resource-constrained environments, mainly leading to wasting devices computing resources and energy.

**Model Update:** To mitigate issues related to data heterogeneity, previous work proposed different ways to update the global model using the collected updates. For instance, authors in [72] propose q-Fair FL (q-FFL), that encourages fairer and more uniform accuracy distributions across devices in FL. q-FFL minimizes an aggregate reweighted loss function, parameterized by $q > 0$ such that the devices with higher loss are given higher relative weight. Authors in [73] proposed weighting mechanism based on how well the local model performs on the model performs on the device's validation data. Yet, the federated averaging (FedAvg) [74] is perhaps the most widely adopted and studied FL algorithm. In FedAvg, the updates are aggregated using weights parameterized using the local datasets' sizes. Nonetheless, measuring the contribution using the dataset size only does not take into consideration the statistical heterogeneity and redundancy of the data.

### 2.3.2 Concept-shift and Data poisoning

While simple data distribution skewness cases might be handled with tweaks and adjustments to the FEEL process, concept-shift and data-poisoning require novel and additional measures, especially since work on these two aspects is still in its early days [75]. an early attempt to focus on data poisoning is found in [76], where authors propose to send a model to be executed locally to assess whether data is poisoned. However, a malicious client can fake the results of the model and a model might not be trained to predict all kinds of possible poisoning attacks. Another approach was proposed, handling both malicious clients and concept-shift is updates clustering. In the following, we discuss how clustering and personalization techniques are used to handle outliers and concept-shift.

**Updates clustering:** Due to training small models in FEEL, the standard process in FEEL might not be suitable in the presence of adversaries (*e.g.,* label flipping attack), and varying preferences. In order to tackle these challenges, updates clustering was proposed to detect malicious clients and overcome limited model complexity. By calculating the updates' similarity, using cosine similarity for instance, the updates can be clustered during a pre-set communication round [77] or when the model has converged to a stationary point [78]. After clustering the updates, new models are created for each cluster in the case of varying preferences for each cluster, or outliers (and possibly malicious devices) can be excluded from the training.

**Personalization:** The standard formulation of FEEL produces one shared model for all clients. However, in some cases, the local models trained solely on the private data perform better than the global shared model due to statistical heterogeneity of data [79]. As a result, several techniques were proposed to personalize global models to work better for individual clients. For instance, work in [80] suggest clustering clients with similar properties (*e.g.,* geographical) and training a separate model for each group. Personalization can also be achieved by fine-tuning a global model using standard optimization methods on data stored locally on a single device[81]. While it is expected that personalization could be beneficial for most clients, it is necessary to ensure that personalized models do not overfit.

### 2.3.3 Benchmark data

Several efforts were put toward evaluating and demonstrating the potential of FL algorithms. Existing and realistic FL scenarios use proprietary federated datasets *e.g.,* crowd-sourced and proprietary data by Huawei in [82] and by Google in [9]. Some realistic-like federated datasets were derived from publicly available data, but which are not straightforward to reproduce or to simulate FEEL scenarios, examples include Shakespeare's works based dataset for text generation, where each of the characters represents a client [83]. As a result, the datasets used in FEEL are usually based on artificial partitions of MNIST[84], MNIST-fashion [85] or CIFAR-10 [86].

### 2.3.4 Conclusion

Several approaches were proposed to tackle the non-i.i.d and unbalanced aspect of FL. For instance, a hybrid form of FL consisting of using public data and data from clients willing to share some sample to balance the distributions. Nonetheless, this method is not always feasible. Other approaches focused on optimizing the different steps in the FL iteration. In the client selection step for instance, the most common goal is the diversify the data sources through maximizing the number of collected updates or through implementing fairness as a selection criterion. The updates' collection step was also modified to prioritize more significant updates through the evaluation of the resulting models and their direction. Lastly, several methods for the model aggregation were proposed to assign weights to the collected updates with different considerations such as fairness and the local data size. However, the local data properties are often overlooked by these general methods in the client selection and updates collection. Likewise, more challenging problems like concept-shift and data poisoning require additional attention. Updates clustering in particular was proposed to mitigate these challenges. Nonetheless, it has is based on some unrealistic expectations like the participation of all the clients in the training round where the clustering is executed. Moreover, personalization techniques, through local retraining, can allow outliers with different usage patterns to be better captured locally without overfitting. Lastly, the empirical evaluation of FEEL algorithms on non-i.i.d data is usually performed on artificial partitions of datasets such as MNIST and CIFAR-10, which do not provide a realistic model of a federated scenario. It is thereby necessary to evaluate its potential to encourage and facilitate future adoption.

## 2.4 Tackling Resources related challenges

Recently, the resource-constrained FEEL systems have surged in popularity, and have been widely studied in the literature. In addition to several compression [87] and partial participation [7] techniques, several mechanisms were proposed to adapt the federated training to the constrained resources. This is often achieved bu optimizing time and energy throughout client selection, local training and model upload steps. Tackling mobility related challenges, especially in vehicular networks, imposes other considerations. In this section, we review and discuss the research efforts in FEEL from resource point of view.

### 2.4.1 Client Selection and Resource Allocation

As communication is FEEL's bottleneck, client selection has been studied jointly with resource allocation in the existing literature. More specifically, algorithms where clients are selected and allocated an amount of time [67], fraction of the bandwidth [50], number of resource blocks [88], and/or transmission power [89].

The work considering resource allocation is two-fold: work aiming to optimize energy, and work focusing on time in terms of round duration.

In work aiming to reduce the **total required energy** for training and upload, the client selection will exclude devices that do not have enough energy to carry-out these tasks [90, 91]. Through resource allocation, adequate transmission power and CPU-frequency will be chosen to maximize the number of participants with an optimal energy consumption across devices [92]. Authors in [93] propose scheduling clients with a long-term perspective, where more participants are included in later rounds compared to earlier rounds.

Work aiming to reduce the **duration time of a round** need to handle the straggler problem [94]. Indeed, the heterogeneity of computation and communication resources will lead to some devices being faster than others. In this case, more bandwidth is allocated for transmission by slower devices and less for faster devices. This to some extent can equalize their total update time (training plus upload time) [95]. Furthermore, to avoid squandering bandwidth on extremely slow devices, client selection should exclude slowest devices by applying thresholds on their expected completion time, which can be inferred using their computing and communication capacities, alongside their channel states [67]. Nonetheless, in the case of highly unbalanced datasets, such method may exclude clients owning larger and richer datasets.

While these works focused either on time or energy, there exists a clear trade-off between these two goals in resource allocation. As a result, several proposed algorithms considered both time and energy during resource allocation [96]. For instance, authors in [97] formulate resource allocation as a multi-objective optimization problem, aiming to minimize both time and energy. But in general, defining the optimization goal depends on the use-case and its constraints. For example, in the case of battery-powered devices, the optimization problem will often have energy constraints, while for devices that are always plugged in, the focus might be on time for instance. For example, authors in [98] aim to minimize communication round's time within an energy budget, while authors in [99] aim to minimize the total energy within a fixed deadline.

### 2.4.2 Model Training and Upload

In order to enable the heavy tasks of training and uploading large models, several compression techniques such as quantization and pruning have been used in FEEL. Quantization [40] consists of changing the model's parameters from floating-point numbers to low-bit width numbers, thus avoiding costly floating-point multiplications. Pruning [100] is also a widely adopted technique of model compression. Pruning a neural network involves removing the least important parameters (*e.g.,* weights below a threshold), namely neurons or connections. Nonetheless, removing neurons might damage the accuracy of the DNN, leading to an important trade-off between the network size and accuracy. Moreover, dynamic choice of training parameters such as number of local epochs and batch size significantly improve the number of devices that can participate in FEEL. Additionally, FL is premised on replacing frequent gradient uploads with more local computation. Such trade-off can be carefully modelled to reduce the overall communication round's cost.

**Partial Updates:** Authors in [7] define two strategies to improve resource usage in model update: structured updates and sketched updates.

- *Structured Updates:* The main advantages gained from this strategy are energy efficiency and low latency in both training and update upload. When it was first proposed in [7], the update is restricted to be a sparse matrix, following a predefined random sparsity pattern (*i.e.,* a random mask). The pattern is generated afresh by a random seed in each round and for each client independently. After training, each client then sends the update and the random seed. Later work in [101] takes resources heterogeneity into account, as it proposes soft-training where an algorithm is designed to choose subsets of neurons of an ANN to be masked in each round for each client based on energy and time profiling.

- *Sketched updates:* Sketched updates only optimize the upload phase in contrast to structured updates which optimize both training and upload. They can be based on edge intelligence compression techniques such as quantization [102] and pruning [100]. Furthermore, similarly to structured updates, sketched updates can be achieved through sending a subset of parameters using a random mask. Variations of sketching were used to enhance communication in a loss-less manner [103], and to enforce privacy in FL [104].

**Communication and computation parameters:** While FL's main idea is trading frequent communication with more local computation, optimizing training (*e.g.,* number of local iterations, batch size) and uploading parameters is necessary to better benefit from this trade-off. For instance, authors in [35] propose an adaptive mechanism where the numbers of global and local iterations are chosen depending on the available communication and computation resources. Authors in [105] design a compression control scheme to balance the energy consumption of local training and wireless communication from the long-term learning perspective. In particular, the compression parameters are elaborately chosen for FL participants adapting to their computing and communication environments. Another idea is found in [106], where authors propose to bypass the synchronization barrier through adaptive batch size, enforced by appropriate learning rates on different devices.

### 2.4.3 Model Aggregation

Resources heterogeneity and scarcity often lead to straggler's problem. A popular direction is asynchronous aggregation with adaptive weighting. Work in [107] proposes FedAsync, which consists of collecting the updates in an asynchronous manner and assigning weights depending on the staleness of the update, allowing a smooth adaptation to heterogeneous resources through a flexible updates collection. Similarly, in [108], the weight assigned to the updated gradients decreases as the staleness value increases. Other works proposed semi-asynchronous approaches. For instance, in [109] local models on clients that were not selected will be cached for several iterations before uploading. However, due to stale updates' effect on learning, and due to the convergence being guaranteed only on a subset of problems [107], synchronous updates collection remains the preferred method [36].

### 2.4.4 Mobility

Vehicles have become equipped with a wide range of sensors and equipment serving diverse applications such as autonomous driving and infotainment. To enhance the overall ML models' performance, the deployed ML/AI models in vehicles need to be updated and improved periodically by original equipment manufacturers (OEM). Traditionally, this would be achieved by uploading the collected data to the OEMs, which creates significant communication overhead and violates data privacy. FL was recently introduced to vehicular applications as an efficient tool for reducing this transmission overhead while also achieving privacy. However, mobility challenges are a roadblock facing an efficient deployment for FL in vehicular networks.

A naive approach in [42] proposes to simply train models when vehicles stay in a fixed location, for example, parking lots, ensuring that the connection is stable. Nonetheless, this is not always suitable since vehicles from different OEMs may not be available in the same area. A joint vehicle selection and updates aggregation based on contract-theory was proposed by authors in [110]. The vehicle selection focuses on image quality. In fact, the mobility has on the quality of captured images by on-board cameras, as they generally suffer from motion blur, noise, and distortion. In particular, the motion blur level varies with instantaneous velocity of each vehicular client. However, this work does not take the impact of the mobility on the training and upload time and the possible disconnections. Authors in [111] focus on the same problem related to the quality of images, but propose a greedy selection algorithm taking into consideration the mobility and heterogeneity of the vehicles. They formulate a joint selection and resource allocation problem aiming to minimize time and energy, and solved using a greedy algorithm. Nonetheless, the communication with the MEC server in vehicular scenarios is subject to several uncertainties and disconnections. Vehicle-to-vehicle (V2V) provides an efficient alternative to evade the communication issues. Authors in [112] introduce a decentralized alternative where vehicles collaboratively train a model without any centralized coordination by utilizing the consensus mechanism of the blockchain. Nonetheless, fully decentralized models are harder to control and reuse by the OEMs.

### 2.4.5 Conclusion

Communication and computation efficiency of FEEL have been largely studied in the literature. Several compression techniques can be leveraged for FEEL, such as sketched and structured updates. Furthermore, optimization algorithms were heavily investigated for client selection, resource allocation, and to select optimal training parameters (*e.g.,* number of local epochs, batch size, number of clients to select). The optimization often targeted time, energy, or both, and usually considered cost budgets (limited energy, limited bandwidth). Nonetheless, data properties were often overlooked, thus making the proposed algorithms inefficient, especially in the case of non-i.i.d and unbalanced data distributions. Furthermore, mobility related challenges, especially in vehicular networks are under-explored. While FEEL can potentially enable several privacy-sensitive applications in AD and infotainement for instance, the iterative process needs several adjustments before its adoption in vehicular networks. Few works were done in this direction, mainly considering the effect of velocity on the quality of captured images, and not on the overall process. Moreover, V2V offers a new opportunity to overcome the communication resources' scarcity, but requires further investigation.

## 2.5 Conclusions and contributions

Through this review of the state of the art, we have briefly presented AI and EC as the main components of EI, and FL as the last piece of the puzzle. Then, we discussed the challenges facing the adoption of FL in wireless edge networks in terms of data and resources. We then discussed recent efforts toward addressing these challenges and identified some of their shortcomings. In particular, bridging the gap between resource-oriented work and ML-focused work is vital for FEEL.

To begin with, data properties were highly overlooked in client selection and update collection steps. In this thesis, we consider such properties at the heart of FEEL. Therefore, we identify several metrics to evaluate the quality of a dataset and its richness. Then, we formulate the client selection as an optimization problem taking into account resources and data. We then extend the proposed algorithm by adding to its robustness against malicious clients launching data poisoning attacks through the evaluation of the collected updates.

Additionally, clustering was used in vehicular networks to enable several services, while a different notion of clustering was proposed in FL to mitigate concept-shift and malicious clients. In this thesis, we propose a combined approach using clustering in a vehicular sense and clustering in FL to enable several applications in vehicular networks.

Lastly, we bring a more realistic sense for FEEL and evaluate its potential in smart city applications. We propose an EI-powered architecture and decision process for energy trading in the smart grid. To the best of our knowledge, we were the first to investigate

FEEL as a solution for building energy forecasting in the smart grid. Additionally, we show through evaluations its potential gain in terms of communication and improved prediction models.

# Chapitre 3: Avant-propos

**Auteurs et affiliation:**

Afaf Taïk: étudiante au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Soumaya Cherkaoui: Professeure, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

**Date d'acceptation:** juin 2021.

**État de l'acceptation:** version finale publiée.

**Revue:** IEEE Network.

**Titre français:** Apprentissage fédéré à la périphérie du réseau : problèmes et défis de conception

**Résumé français:**

L'apprentissage fédéré (en anglais Federated Learning FL) est une technique d'apprentissage automatique distribué, où chaque dispositif contribue au modèle d'apprentissage en calculant indépendamment le gradient sur ses données d'entraînement locales. Cette technique est récemment devenue un sujet de recherche attrayant, car elle promet plusieurs avantages liés à la confidentialité des données et à la scalabilité. Cependant, la mise en œuvre de FL à la périphérie du réseau est difficile en raison de l'hétérogénéité des systèmes et des données et des contraintes de ressources. Dans cet article, nous examinons les défis et les compromis existants dans le contexte de FL à la périphérie du réseau (en anglais Federated Edge Learning FEEL). La conception d'algorithmes FEEL pour un apprentissage efficace en termes de ressources présente plusieurs défis essentiellement liés à la nature multi-

disciplinaire du problème. Comme les données sont la composante clé de l'apprentissage, cet article préconise un nouvel ensemble de considérations pour les caractéristiques des données dans les algorithmes d'ordonnancement sans fil dans FEEL. Par conséquent, nous proposons un cadre général pour l'ordonnancement tenant compte des données comme ligne directrice pour les futures directions de recherche. Nous discutons également des principaux axes et exigences de l'évaluation des données et de certaines techniques et mesures exploitables.

# Chapitre 3: Foreword

**Authors and affiliation:**

Afaf Taïk: Ph.D. Student, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Soumaya Cherkaoui: Professor, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

**Date of acceptance:** June 2020.

**Acceptance status:** final version published.

**Venue:** IEEE Network.

**Title:** Federated Edge Learning : Design Issues and Challenges.

CHAPTER 3

# Federated Edge Learning : Design Issues and Challenges.

## 3.1 Abstract

Federated Learning (FL) is a distributed machine learning technique, where each device contributes to the learning model by independently computing the gradient based on its local training data. It has recently become a hot research topic, as it promises several benefits related to data privacy and scalability. However, implementing FL at the network edge is challenging due to system and data heterogeneity and resources constraints. In this article, we examine the existing challenges and trade-offs in Federated Edge Learning (FEEL). The design of FEEL algorithms for resources-efficient learning raises several challenges. These challenges are essentially related to the multidisciplinary nature of the problem. As the data is the key component of the learning, this article advocates a new set of considerations for data characteristics in wireless scheduling algorithms in FEEL. Hence, we propose a general framework for the data-aware scheduling as a guideline for future research directions. We also discuss the main axes and requirements for data evaluation and some exploitable techniques and metrics.

## 3.2 Introduction

The growing interest in intelligent services motivates the integration of artificial intelligence (AI) in Internet of Things (IoT) applications. The collection of large volumes from the different devices and sensors is necessary for training AI models. However, uploading massive data generated by connected devices to the cloud is usually impractical, mainly due to issues including privacy, network congestion, and latency. Federated Edge Learning (FEEL) [113] is a Machine Learning (ML) setting that utilizes edge computing [12, 114] to tackle these concerns. In contrast to centralized ML, Federated Learning (FL) [7] consists of training the model on the devices, with the orchestration of a central entity, where only the resultant model parameters are sent to the edge servers to be aggregated. FEEL refers to the use of FL at the edge of the network, which makes it a promising solution for privacy preserving ML.

An important design decision for a FEEL algorithm is whether to choose either asynchronous or synchronous aggregation. Recent works tend to promote synchronous training, where, for instance, synchronization among participating devices is required for updates averaging [7] and privacy-preservation [115]. However, there are many challenges upon using synchronous FL in edge environments.

To begin with, the heterogeneity of resources across different devices sparks new system challenges. For instance, significant delays can be caused by stragglers. Moreover, communication loads across devices limit the scalability of FL for large models. Participating devices communicate full model updates during every training iteration, which are of the same size as the trained model. For large models, such as deep neural networks, the model size can be in the range of gigabytes. As a result, if communication bandwidth is limited or communication is costly, FEEL can be deemed impractical or unfeasible, as communication overhead becomes a bottleneck for FEEL.

Furthermore, end devices have limited battery lives and varying available energy levels. As training ML models is a computation-heavy task, only devices that have enough energy can be solicited to participate. Furthermore, energy and computational constraints limit both the size of the models that can be trained on-device, and the number of local training iterations.

Additionally, as the data collected by the clients depends on their local environment and usage pattern, both the size and the distribution of the local datasets will typically vary between different clients. This non-Independently and Identically Distributed (non-IID) and unbalanced nature of data across the network imposes significant challenges linked to models' convergence.

Consequently, designing an efficient FEEL algorithm should take into account the limited and heterogeneous nature of the resources, alongside the non-IID and unbalanced aspect of the data distributions. In general, proposed FEEL algorithms target efficient selection of participant devices, optimization of the resource allocation and usage, or adequate updates' aggregation. However, it is hard to capture both the resources problems and the learning goal, as there is no direct relation between the model's loss function and the resource optimization. A manageable approach found in current works is to focus on resource optimization with certain learning guarantees, such as maximizing the number of collected updates and maintaining the level of local accuracy [50]. Nonetheless, these guarantees are not sufficient, as a significant drop in accuracy is observed when data is non-IID and unbalanced. Therefore, we propose to lighten the effects of design trade-offs through the direct integration of the data properties in the device selection and resource optimization algorithms. In fact, data properties were at the heart of FL since its inception, but they have been largely overlooked in the design of FEEL algorithms. Moreover, data diversity has long been premised on in active learning, where models can be trained using few labelled data samples if the highly diverse data is selectively added to the training set. Thus, data diversity should be considered in the design of FEEL algorithms, as we advocate in this article.

The main contributions of this article can be summarized as follows:

– We discuss the FEEL challenges imposed by the nature of the edge environment, from an algorithms design perspective. We review the challenges related to computational and communication capacities, as well as data properties, as they are at the core of the trade-offs in learning and resource optimization algorithms.

- We propose a general framework for incorporating data properties in FEEL, by providing a guideline for a thorough algorithm design, and criteria for the choice of diversity measures in both datasets and models.
- We present several possible measures and techniques to evaluate data and model diversity, which can be applied in different scenarios (e.g., classification, time series forecasting), in an effort to assist fellow researchers to further address FEEL challenges.

The remainder of this article is as follows. In Section II, we review the challenges found in designing FEEL algorithms, and we derive the main trade-offs. Then, we shed the light on a new data-aware design direction for FEEL algorithms in section III. Some possible techniques and methods to evaluate diversity are detailed in this section. At last, a conclusion and final remarks are presented in Section IV.

## 3.3 Design challenges : Overview

FEEL has several constraints related to the nature of the edge environment. In fact, FEEL involves the participation of heterogeneous devices that have different computation and communication capabilities, energy states, and dataset characteristics. Under device and data heterogeneity, in addition to resources constraints, participants selection [67] and resource allocation have to be optimized for an efficient FEEL solution.

### 3.3.1 Design Challenges

The core challenges associated with solving the distributed optimization problem are twofold: Resources and Data. These challenges increase the FEEL setting complexity compared to similar problems, such as distributed learning in data centers.

**Resources:** The challenges related to the resources, namely computation, storage and communication, are mainly in terms of their heterogeneity and scarcity.

*Heterogeneity of the resources:* The computation, storage and communication capabilities vary from a device to another. Devices may be equipped with different hardware (CPU and memory), network connectivity (e.g., 4G/5G, Wi-Fi), and may differ in available power (battery level). The gap in computational resources creates challenges such as delays caused by stragglers. FEEL algorithms must therefore be adaptive to the heterogeneous hardware and be tolerant toward device drop-out and low or partial participation. A potential solution to the straggler problem is asynchronous learning. However, the reliability of asynchronous FL and the model convergence in this setting are not always guaranteed. Thus, synchronous FL remains the preferred approach.

*Limited Resources:* In a contrast to the cloud, the computing and storage resources of the devices are very limited. Therefore the models that can be trained on device are relatively simpler and smaller than the models trained on the cloud. Furthermore, devices are frequently offline or unavailable either due to low battery levels, or because their resources are fully or partially used by other applications.

As for the communication resources, the available bandwidth is limited. It is therefore important to develop communication-efficient methods that allow to send compressed or partial model updates. To further reduce communication cost in FEEL settings, two potential directions are generally considered 1) reducing the total number of communication rounds until convergence [49], and 2) reducing the size of the transmitted updates through compression and partial updates [7].

**Data:** In most cases, data distributions depend on the users' behaviour. As a result, the local datasets are massively distributed, statistically heterogeneous (i.e., non-IID and unbalanced), and highly redundant. Additionally, the raw generated data is often privacy-sensitive as it can reveal personal and confidential information.

*Small and widely distributed datasets:* In FEEL scenarios, a large number of devices participate in the FL training with a small average number of data samples per client. Learning from small datasets makes local models prone to overfitting.

*Non-IID:* The training data on a given device is typically based on the usage of the device by a particular user, and hence any particular user's local dataset will not be representative of the population distribution. This data-generation paradigm fails to comply with the independent and identically distributed (IID) assumptions in distributed optimization, and thus adds complexity to the problem formulation and convergence analysis. The empirical evaluation of FEEL algorithms on non-IID data is usually performed on artificial partitions of MNIST or CIFAR-10, which do not provide a realistic model of a federated scenario.

*Unbalance:* Similarly to the nature of the distributions, the size of the generated data depends on the user. Depending on users' use of the device, these may have varying amounts of local training data.

*Redundancy:* The unbalance of the data is also observed within the local datasets at a single device. In fact, IoT data is highly redundant. In sequential data (e.g., video surveillance, sensors data) for instance, only a subset of the data is informative or useful for the training.

*Privacy:* The privacy-preserving aspect is an essential requirement in FL applications. The raw data generated on each device is protected by sharing only model updates instead of the raw data. However, communicating model updates throughout the training process can still be reverse-engineered to reveal sensitive information, either by a third-party or a malicious central server.

### 3.3.2 Design Trade-offs

Several efforts were made to tackle the aforementioned challenges. However, FEEL is a multi-dimensional problem that brings about several trade-offs. As a result, algorithms designed to address one issue at a time are deemed unpractical. Perhaps a tractable solution may be to combine several techniques when developing and deploying FEEL algorithms.

In general, an end-to-end FEEL solution should cover devices selection, resource allocation, and updates aggregation. In the following, we discuss major trade-offs that should be considered when designing solutions in the FEEL setting.

*1) General FEEL solution*
Given the wide range of applications that can benefit from FEEL, there is no one-size-fits-all solution. However, in general, a FEEL solution needs to act on the following aspects:

**Device selection:** Participant selection refers to the selection of devices to receive and train the model in each training round. Ideally, a set of participants is randomly selected by the server to participate. Then, the server has to aggregate parameter updates from all participants in the round before taking a weighted average of the models. However, due to the communication bottlenecks and the desire to tame the training latency, the device selection should be optimized in terms of resources [67] and data criteria.

**Resource allocation:** Device selection should not be considered independently from resource allocation, especially computation and bandwidth. We refer to the joint selection and resource allocation as a scheduling algorithm. Indeed, the number of scheduled devices is limited by the available bandwidth that can be allocated. Additionally, for an optimal learning round duration and energy consumption, both bandwidth and computation resources should be adapted based on the number of local iterations at each device, and the number of global iterations (i.e., learning rounds) [35]. Due to the fast-changing aspect of the FEEL environment, the computational complexity of scheduling algorithms should be especially low. Therefore, the use of meta-heuristics and heuristics should be encouraged.

**Updates aggregation:** This aspect of the solution design refers to how the updates are aggregated and how frequently they are aggregated. For instance, the frequency of the communication and aggregation can be reduced with more local computation [35], or reduced through selective communication of gradients [49]. For instance, FedAvg [65] is one of the most used methods in aggregation which uses weighted average Stochastic Gradient Descent updates, where the corresponding weights are decided by the volume of the training dataset. While FedAvg uses synchronous aggregation, in FedAsync [107] algorithm, newly received local updates are weighted according to their staleness, where stale updates received from stragglers are weighted less based on how many rounds elapsed. It should also be noted that proposing new aggregation methods requires theoretical and empirical convergence analysis to guarantee that the learning loss function will converge to a global optimum. Updates aggregation should also be communication-efficient [7, 49] and secure by the means of techniques such as differential privacy [115].

*2) Optimization axes*

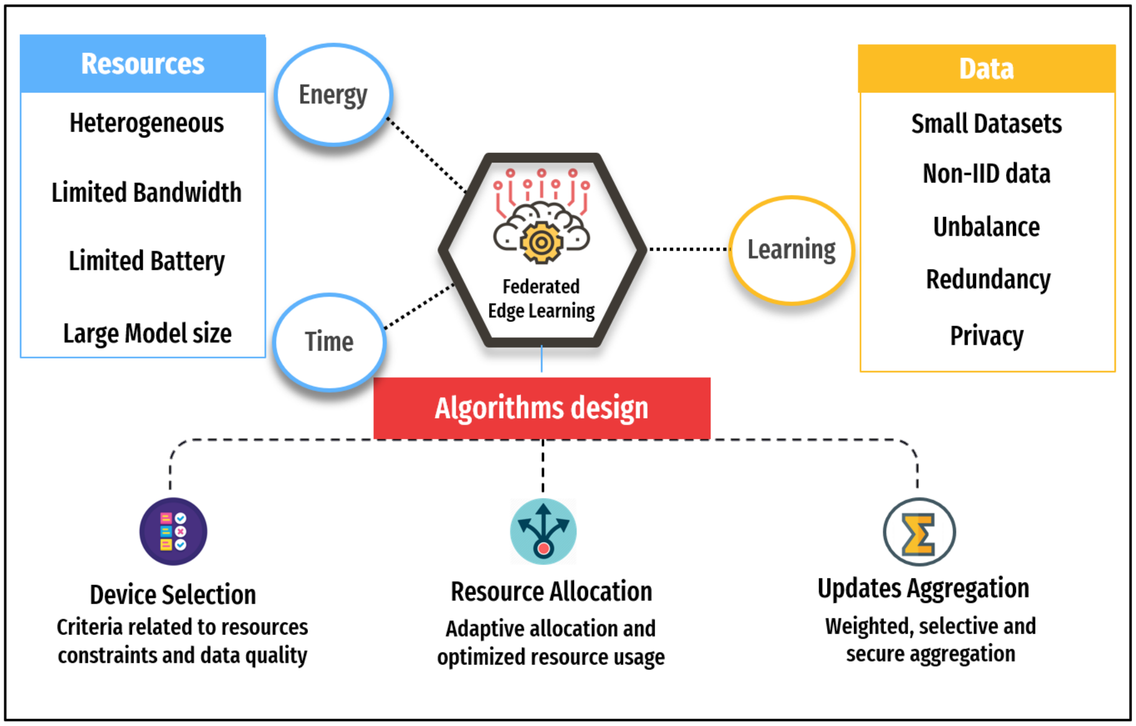


Figure 3.1 FEEL algorithms, challenges and optimization axes

In addressing FEEL challenges, three optimization axes are often considered: Time, Energy and Learning. In many cases, the FEEL algorithm can be viewed as a Pareto optimal problem [116]. The relation between the three axes and the challenges is illustrated in Figure 3.1.

**Time optimization:** Accelerating the learning time can be evaluated with different lenses: learning round duration and time until learning convergence. Due to the synchronous model aggregation of FEEL, the total duration of a round is determined by the slowest device among all the scheduled devices [48]. For this reason, more bandwidth should be allocated for transmission by stragglers and less for faster devices. This to some extent can equalize their total update time (computing plus communication time). Furthermore, to avoid squandering bandwidth on extremely slow devices, scheduling (i.e., joint selection and resource allocation) should exclude slowest devices by applying thresholds on their expected completion time, which can be inferred using their computing capacities and channel states. From a learning perspective, the learning latency is determined by the number of rounds until convergence. The optimization techniques centered on this aspect mainly focus either on the selective upload of updates, or on maximizing the participating devices in each round.

**Energy optimization:** Optimizing the energy consumption across the network is necessary to reduce the rate of drop-out devices because of battery drainage. In fact, training and transmission of large-scale models are energy consuming, while most edge and end devices have limited battery lives. Additionally, using the maximum capacity of the devices would make the users less-likely willing to participate in the training. A design goal of a scheduling algorithm (i.e., joint selection and resource allocation) would be to allocate bandwidth based on the devices' channel states and battery levels. As a result, more bandwidth should be allocated to devices with weaker channels or poorer power states, to maximize the collected updates [50].

**Learning optimization:** In contrast to centralized learning, optimizing the learning in the FEEL setting cannot be seen independently from time and energy optimization. However, capturing the optimization of time, energy and the learning goal in the same optimization problem is hard, because there is no direct relation between the objective function of the learning (i.e., the loss function) and the time and energy minimization goal. A manageable approach used is to minimize time and energy under a certain convergence speed guarantee. For instance, some works argue that the number of collected updates in each round is inversely proportional to the convergence speed, and therefore is used as a guarantee [50]. Indeed, multi-user diversity (i.e., collecting a maximum of updates) can yield a high convergence speed, especially in IID environments, however there is a significant chance of choosing the same sets of devices repeatedly. To avoid this issue, a goal of the FEEL algorithm can be to maximize the fairness in terms of the number of collected updates among devices [69]. The fairness measure maximizes the chance of more diverse data sources, thus achieving gradient diversity. Nonetheless, the number of collected updates in this setting might be low. The fairness is also considered in the aggregation by q-Fair FL (q-FFL) [72], which reweighs the objective function in FedAvg to assign higher weights in the loss function to devices with higher loss. Another approach is to use data size priority, which maximizes the size of data used in the training, by using a probability of selection inversely proportional to the available dataset's size. In the background, these scheduling algorithms all share the same idea : if the size of the training data is large then the training would converge faster. However, IoT data is highly redundant and inherently unbalanced. Thus, many of the proposed algorithms witness a drop in performance in non-IID and unbalanced experiments. Therefore, the data properties should be considered throughout the FEEL algorithm.

## 3.4 Data-aware FEEL design: Future Direction

Even if FL was first proposed with data as a central aspect, it has been overlooked in the design of proposed FEEL scheduling algorithms. With the significant drop of accuracy of models trained with resource-aware FEEL algorithms in non-IID and unbalanced settings, it becomes clear that the data aspect should be considered. Henceforth, we propose a new possible data-aware end-to-end FEEL solution based on the diversity properties of the different datasets. In general, diversity consists of two aspects, namely, richness and uncertainty. Richness quantifies the size of the data, while the uncertainty quantifies the information contained in the data. In fact, it has been long proven in Active Learning that by choosing highly uncertain data samples, a model can be trained using fewer labelled data samples. This fact suggests that data uncertainty should be incorporated into the design of FL scheduling algorithms. Nonetheless, the uncertainty measures used in Ac-

tive Learning targets individual samples from unlabeled data in a centralized setting, thus, these measures cannot be directly integrated in FEEL. In the FEEL setting, the updates' scheduling can be either before the training or after it, therefore the diversity measures should be selected depending on the time of scheduling. If the scheduling before the training is preferred, then the datasets' diversity is to be considered. Otherwise, if the scheduling is set after the training is over, the diversity to be considered is model diversity, as the diversity of the dataset can be reflected by the resulting model. In both cases, in addition to maximizing the diversity through careful selection of participating devices, the scheduling algorithm can focus on minimizing the consumed resources in terms of completion time of FL and transmission energy of participating devices. For the pre-training scheduling, local computation energy can also be optimized. Furthermore, the scheduling problems' constraints are to be derived from the environment's properties concerning resources and data.
In this section, and to better illustrate the data-aware solutions, we consider the architecture illustrated in Figure 3.2. The architecture is a cellular network composed of one base station (BS) equipped with a parameter server, and $N$ devices that collaboratively train a shared model. In the following, we discuss different constraints related to the scheduling algorithms in this setting. Then, we present pre-training and post-training algorithms guidelines, where we detail the key criteria for the design of data-aware FEEL solutions, and we present some potential measures and methods to enable a variety of data-aware FEEL applications, which are summarized in Figure 3.3.

### 3.4.1 Scheduling Constraints

The scheduling algorithms' must consider the following constraints that arise from the FEEL environment's properties: *Energy consumption:* Due to the limited energy level and the high computational requirements of training algorithms, it is necessary to evaluate a device's battery level before scheduling

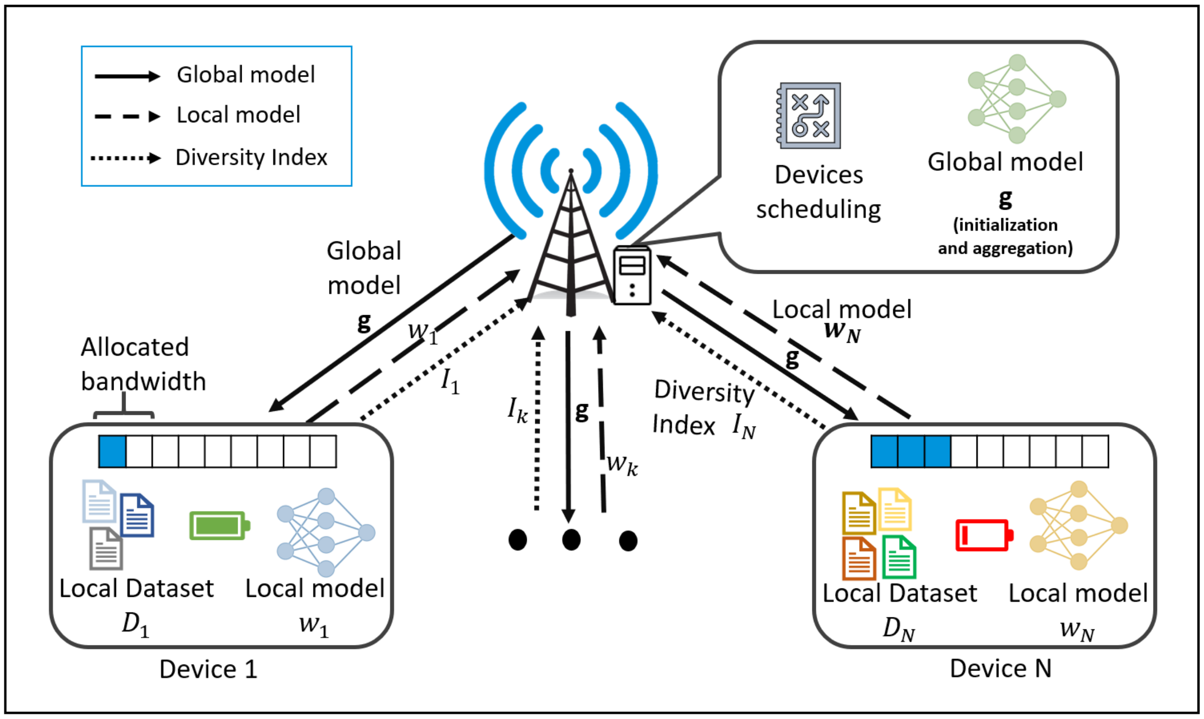


Figure 3.2 The proposed FEEL system model

it for a training round. When first FL was proposed, the selected devices were limited to the ones plugged for charging. However, this criterion limits the number of devices that can be selected, leading to a slow convergence of the learning.

*Radio Channel State:* It is important to consider the radio channel state changes in the scheduling. The quality of the communication is critical for both the device selection and resource allocation.

*Expected completion time:* The available computation resources, alongside data size, can be used to estimate the completion time of the device. Potential stragglers can be discarded even before the training process.

*Number of participants:* A communication round cannot be considered valid unless a minimum number of updates is obtained. Therefore, a training round can be dropped if there are not enough devices to schedule.

*Data size:* The available data in the device is smaller in size than a required minimum, it can be immediately discarded from the selection process. For instance, if the number of samples is less than the selected mini-batch size, the device should be excluded.

### 3.4.2 Pre-training scheduling: Dataset Diversity

The pre-training scheduling that we propose uses dataset diversity to choose devices that will conduct the training and send the updates. Scheduling the devices before the training allows to eliminate potential stragglers, and adapt the number of epochs based on the battery levels available at the participating devices.

*1) Scheduling algorithm:*

In this algorithm, the global model is initialized by the BS. Afterwards, the following steps are repeated until the model converges or a maximum of rounds is attained:

– **Step 1:** At the beginning of each training round, the devices send their diversity indicators and battery levels to the server.

– **Step 2:** Based on the received information, alongside with the evaluated channel state indicator, the server schedules a subset of devices and sends them the current global model.

– **Step 3:** Each device in the subset uses its local data to train the model.

– **Step 4:** The updated models are sent to the server to be aggregated.

– **Step 5:** The PS aggregates the updates and created the new model.

*2) Datasets Diversity Measures:*

In the pre-training scheduling, dataset diversity will serve essentially as a lead for device selection, where it should prioritize devices that have potentially informative datasets with less redundancy, to speed up the learning process. While the richness of datasets can be easily quantified through the total number of samples, the uncertainty of the dataset depends strongly on the application. For supervised learning, the uncertainty can be evaluated through the evenness of the dataset (i.e., the degree of balance between the classes in classification problems), which can be calculated through entropy measures. For sequence data, the uncertainty is reflected by the regularity of the series. Moreover, for unsupervised learning, local dissimilarity between pseudo-classes or randomly sampled data points can be considered. Furthermore, it is essential to consider the privacy as a component of the used index. Sending the number of samples from each class for instance is a violation of the privacy principle of FEEL. In the following, we introduce some potential methods to evaluate datasets diversity.

**Diversity measures for classification:** The measures of diversity have long been used in Active learning. In fact, uncertainty is used to choose the samples that should be labeled as this task is costly. However, in FL, the client selection does not concern independent samples, instead the diversity should be evaluated at the level of the entire dataset. Moreover, in the premise of supervised FL, the labels are already known, which gives the possibility to use more informed measures. For instance, Shannon Entropy or Gini-Simpson index are suitable measures for datasets' uncertainty in classification problems. Shannon Entropy and Gini-Simpson index both favor IID partitions, where the maximum for both indexes is obtained for balanced distributions and the datasets with a single class has the minimum possible value. The Shannon entropy quantifies the uncertainty (entropy or degree of surprise) of a prediction. It was first proposed to quantify the information content in strings of text. The underlying idea is that when a text contains more different letters, with almost equal proportional abundances, it will be more difficult to correctly predict which letter will be the next one in the string. However, Shannon Entropy is not defined for the case of classes with no representative samples. Therefore, it may not practical in scenarios with high unbalance. Another possible measure is the Gini-Simpson index. The Simpson index $\lambda$ measures the probability that two samples taken at random from the dataset of interest are from the same class. The Gini–Simpson index is its transformation $1 - \lambda$, which represents the probability that the two samples belong to different classes. Nonetheless, if the number of classes is large, the distinction using this index will be hard.

**Diversity measures for time-series forecasting:** In time series problems, other methods can be used, such as Approximate Entropy (ApEn) and Sample Entropy (SampEn). In sequential data, statistical measures such as the mean and the variance are not enough to illustrate the regularity, as they are influenced by system noise. ApEn is proposed to quantify the amount of regularity and the unpredictability of time-series data. It is based on the comparison between values of data in successive vectors, by quantifying how many data points vary more than a defined threshold. SampEn was proposed as a modification of ApEn. It is used for assessing the complexity of time-series data, with the advantage of being independent from the length of the vectors.

**Diversity measures for clustering tasks:** For clustering tasks, a similarity measure between data points from a randomly sampled subset should be considered. The measure can be distance based (e.g., Euclidean distance, Heat Kernel) or angular based (e.g., cosine similarity). A higher value is obtained if most of the data points in the sample are dissimilar, and thus the dataset should be considered as more diverse. It should be noted that angular based measures are invariant to scale, translation, rotation, and orientation, which makes them suitable for a wide range of applications, particularly multivariate datasets.

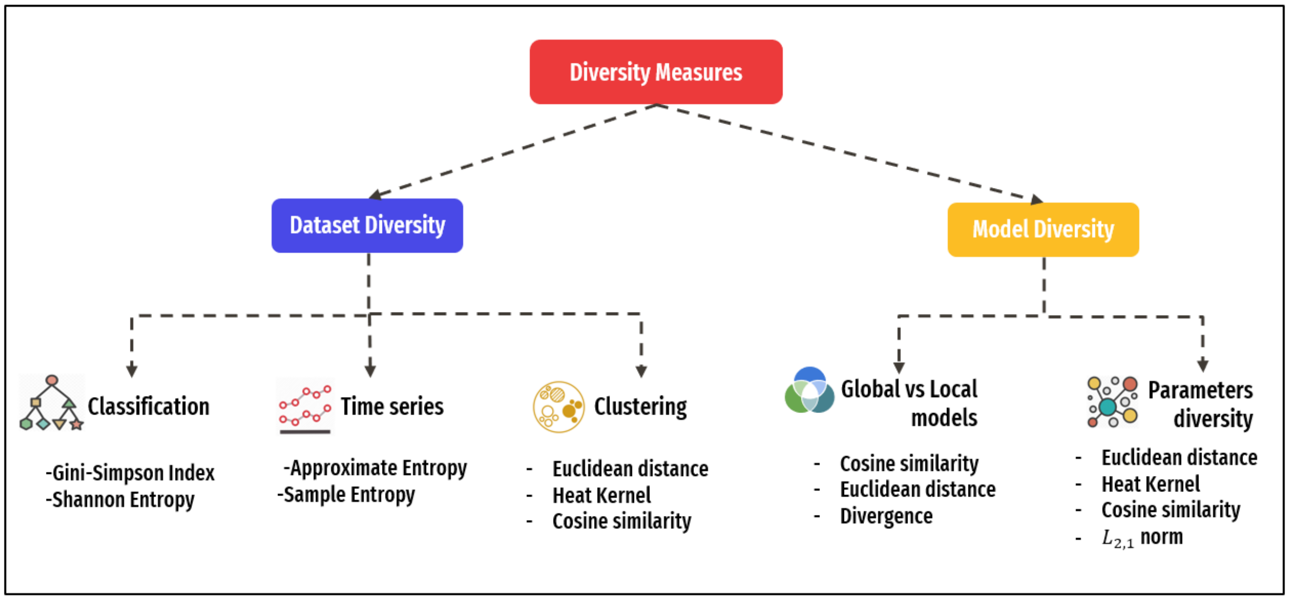


Figure 3.3 Diversity measures that can be used in pre-training and post-training scheduling

### 3.4.3 Post-training scheduling: Model Diversity

The post-training setting uses model diversity to choose devices that will send the updates. The model diversity is evaluated on two different aspects: 1) by comparing the dissimilarity between the local model's parameters and the previous global model's parameters. 2) by comparing the diversity within the model's parameters. In fact, choosing the local models that are divergent from the previous global model will possibly improve the representational ability of the global model directly, by aggregating updates that have potentially new information. Furthermore, if a dataset is highly unbalanced and limited in size, the model's parameters would be very similar. The redundancy within parameters negatively affects the model's representational ability. It is therefore necessary to prioritize updates with high diversity. In the following, we detail the post-training scheduling algorithm, then we present some possible measures for model diversity.

*1) Scheduling algorithm:*

Similarly to pre-training scheduling, the global model is initialized by the BS. Afterwards, the following steps are repeated until the model converges or a maximum of rounds is attained:

- **Step 1:** At the beginning of each training round, devices receive the current model.
- **Step 2:** Each device in the subset uses its local data to train the model.
- **Step 3:** The server sends an update request to the devices, to which each device responds by sending its model diversity index.
- **Step 4:** Based on the received information, alongside with the evaluated channel state indicator, the server schedules a subset of devices to upload their models. Then, the updated models are sent to the server to be aggregated.
- **Step 5:** The PS aggregates the updates and created the new global model.

*2) Model Diversity Measures:*

While the richness aspect of the diversity is irrelevant in models diversity due to fixed model size among devices, the information contained in the models can be quantified through how the local model's vary compared to the global model, and how the parameters within the same model repulse from each other. Some possible measures are as follows:

**Local and global models' dissimilarity:** Choosing the local models that are divergent from the previous global model will possible improve the representational ability of the global model directly [49]. Pairwise similarity measures such as cosine similarity and Euclidean distance can be used to evaluate the similarity of the new local parameters and the global parameters. Moreover, Divergence, a Bayesian method used to measure the difference between different data distributions, can be used to evaluate diversity of the learned model compared to the global model. Nonetheless, relying on model's dissimilarity might lead to collecting updates from outliers. It is thereby necessary to regulate these diversity measures through the use of thresholds in particular.

**Parameters Dissimilarity:** To evaluate the redundancy within the model's parameters, the same similarity measures used for clustering can also be applied to the parameters. Additionally, the $L_{2,1}$ norm can be used to obtain a group-wise sparse representation of the dissimilarity [117]. The internal $L_1$ norm encourages different parameters to be sparse, while the external $L_2$ norm is used to control the complexity of entire model.

## 3.5 Conclusion

Federated Learning is a promising machine learning technique by virtue of its privacy-preserving aspect and ability to handle unbalanced and non-IID data. However, deploying federated learning based solutions at the edge of the network is subject to several challenges. In fact, FEEL is a multi-disciplinary problem that requires optimization over both the resources and the data. Nonetheless, the data properties are overlooked in many parts of the proposed algorithms, despite being the essence of federated learning. Several FEEL design challenges and issues are introduced and discussed in terms of trade-offs. Furthermore, a new research direction is presented in an effort to incorporate the datasets' diversity properties into the design of FEEL algorithms. Our proposed method supposes that the data quality and veracity are guaranteed, which requires leveraging other techniques such as the blockchain as a trusted third-party for data verification.

# Chapitre 4: Avant-propos

**Auteurs et affiliation:**

Afaf Taïk: étudiante au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Zoubeir Mlika: Stagiaire Postdoctoral, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Soumaya Cherkaoui: Professeure, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

**Date d'acceptation:** juin 2021.

**État de l'acceptation:** version finale publiée.

**Revue:** IEEE Transactions on Cognitive Communication and Networking (IEEE TCCN).

**Titre français:** Programmation des clients participants à l'apprentissage fédéré à la périphérie du réseau en fonction des propriétés des données.

**Résumé français:**

L'apprentissage fédéré à la périphérie du réseau désigne l'apprentissage collaboratif de modèles d'apprentissage automatique par des dispositifs connectés sous l'orchestration d'un serveur dans un réseau sans fil à la périphérie du réseau. En raison des fréquentes mises à jour des modèles, l'apprentissage fédéré doit s'adapter aux limitations de la périphérie du réseau telles que la bande passante de communication limitée, la faible énergie des dispositifs et à l'hétérogénéité statistique des distributions de données locales. Il

est donc nécessaire de plrogrammer soigneusement un sous-ensemble de dispositifs pour l'entraînement et le téléchargement de modèles. Contrairement aux travaux antérieurs concernant l'apprentissage fédéré à la périphérie du réseau où les aspects liés aux données sont peu explorés, nous considérons les propriétés des données au cœur de l'algorithme d'ordonnancement proposé. Ainsi, nous proposons un nouveau schéma d'ordonnancement tenant en compte la nature des distributions de données locale des dispositifs.

Compte tenu du fait que les données sont l'élément clé de l'apprentissage, nous proposons un nouvel ensemble de considérations pour les caractéristiques des données dans les algorithmes d'ordonnancement dans les réseaux sans fil. En fait, les données collectées par les appareils dépendent de l'environnement local et du modèle d'utilisation. Ainsi, les ensembles de données varient en taille et en distribution entre les appareils. En effet, les ensembles de données sont souvent non indépendantes et non distribuées de manière identique et non équilibrées. Dans l'algorithme proposé, nous tenons compte à la fois des perspectives des données et des ressources. En plus de minimiser le temps d'exécution de l'algorithme d'apprentissage fédéré ainsi que l'énergie de transmission des dispositifs participants, l'algorithme donne la priorité aux dispositifs possédant des ensembles de données riches et diversifiés. Nous définissons d'abord un cadre général pour l'ordonnancement axé sur la nature des données et les principales exigences pour l'évaluation de la diversité. Ensuite, nous discutons des aspects de la diversité et de certaines techniques et mesures exploitables selon les cas d'usage. Ensuite, nous formulons le problème et présentons notre algorithme d'ordonnancement. Des évaluations dans différents scénarios montrent que notre algorithme peut aider à atteindre une haute précision en quelques itérations avec un coût réduit.

# Chapitre 4: Foreword

**Authors and affiliation:**

Afaf Taïk: Ph.D. Student, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Zoubeir Mlika: PostDoctoral Researcher, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Soumaya Cherkaoui: Professor, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

**Date of acceptance:** June 2021.

**Acceptance status:** final version published.

**Venue:** IEEE Transactions on Cognitive Communication and Networking (IEEE TCCN)

**Title:** Data-Aware Device Scheduling for Federated Edge Learning[1]

1. https://github.com/afaf-taik/Data-aware-FL

# CHAPTER 4

# Data-Aware Device Scheduling for Federated Edge Learning

## 4.1 Abstract

Federated Edge Learning (FEEL) involves the collaborative training of machine learning models among edge devices, with the orchestration of a server in a wireless edge network. Due to frequent model updates, FEEL needs to be adapted to the limited communication bandwidth, scarce energy of edge devices, and the statistical heterogeneity of edge devices' data distributions. Therefore, a careful scheduling of a subset of devices for training and uploading models is necessary. In contrast to previous work in FEEL where the data aspects are under-explored, we consider data properties at the heart of the proposed scheduling algorithm. To this end, we propose a new scheduling scheme for non-independent and-identically-distributed (non-IID) and unbalanced datasets in FEEL. As the data is the key component of the learning, we propose a new set of considerations for data characteristics in wireless scheduling algorithms in FEEL. In fact, the data collected by the devices depends on the local environment and usage pattern. Thus, the datasets vary in size and distributions among the devices. In the proposed algorithm, we consider both data and resource perspectives. In addition to minimizing the completion time of FEEL as well as the transmission energy of the participating devices, the algorithm prioritizes devices with rich and diverse datasets. We first define a general framework for the data-aware scheduling and the main axes and requirements for diversity evaluation. Then, we discuss diversity aspects and some exploitable techniques and metrics. Next, we formulate the problem and present our data-aware scheduling (DAS) algorithm for FEEL.

Evaluations in different scenarios show that DAS algorithm can help achieve high accuracy in few rounds with a reduced cost.

## 4.2 Introduction

Machine learning (ML) models require large and rich sets of data for training. Nonetheless, the collection of large volumes of data generated by connected devices over wireless networks raises concerns related to data privacy and network congestion [118, 119]. Federated edge learning (FEEL) [74, 120] was proposed to tackle these concerns, by implementing distributed ML at the edge of the network. In addition to preserving privacy by keeping the data locally, FEEL benefits from rapid access to the data generated by end devices and leveraging their computational resources. In FEEL, the model training is performed on edge devices with the orchestration of a multi-access edge computing (MEC) server. Each device trains the model using its local data, and only the resultant model parameters or stochastic gradients are sent to the MEC server for aggregation.

The scarce resources, especially the communication bandwidth, limit the efficacy of FEEL operations, particularly for the transmission of large size models. Consequently, most of the existing works in FEEL focus on designing scheduling algorithms with optimal resource usage. Several proposed works aim, for instance, to minimize the completion time of FEEL [67], local computation energy [121], or transmission energy of participating devices [50]. As a result, the number of scheduled devices is often restricted as a means to meet latency and energy constraints. This restriction often slows down the convergence of training [122, 50]. Therefore, scheduling algorithms aim to maximize the number of collected updates in each round, but this scheduling goal can be biased towards powerful devices with smaller datasets. Thus, the collected updates might not be representative, as they are not trained on richer data. To avoid this issue, scheduling algorithms should also aim to diversify the participating devices through the use of fairness measures [69, 70]. But, the amount of connected devices grows faster than the network capacities, which will make scaling these algorithms harder in practice. Moreover, Internet of things data is highly redundant and inherently unbalanced, given that the data collected by the devices depend on the local environment and the device's usage pattern. Therefore, the size and the statistical properties of local datasets distributions vary among devices [47]. Thus, a careful selection of participating devices imposes the consideration of their data properties, which motivates this work.

The main idea we advocate in this paper finds its roots in active learning [123, 124], where models are trained using fewer data points provided that the chosen samples are required to be diverse and informative. While in active learning, the selection concerns single unlabelled data points, the selection in FEEL concerns complete datasets with already labelled data points, and therefore requires a different evaluation of diversity. Additionally, the incorporation of the diversity measures in FEEL requires different considerations in regards of privacy and the properties of the FEEL setting [125]. In this paper, we consider diversity as the baseline criterion for choosing participating devices in FEEL. The diversity evaluation is applied on datasets, where the priority is given to devices with potentially more informative datasets to speed up the training process. To this end, we propose a method for incorporating datasets' diversity properties in FEEL scheduling, by identifying a set of dataset diversity measures, and designing a data-aware scheduling (DAS) algorithm.

The contributions of this paper can be summarized as follows:

1) we design a suitable diversity indicator, which serves as a priority criterion for the selection of devices;

2) we formulate a joint device selection and bandwidth allocation problem taking into account data diversity;
3) we prove that the formulated problem is NP-hard and we propose a data-aware scheduling algorithm based on an iterative decomposition technique to solve it; and
4) we evaluate the proposed diversity indicator and the DAS algorithm through extensive simulations.

The remainder of this paper is organized as follows. In Section II, we present the background for FEEL and related work. In Section III, we present the design of the proposed diversity measure, starting with the used uncertainty measures and their integration in FEEL. In Section IV, we integrate the proposed measure in the design of the joint selection and bandwidth allocation algorithm. Simulation results are presented in Section V. At last, conclusions and final remarks are presented in Section VI.

## 4.3 Background and related work

In this section, we start by briefly introducing the main concepts of FEEL. Then, we describe the existing challenges in deploying FL in wireless edge networks. Next, we discuss the related work, illustrate the existing research gaps and motivate the need for a new scheduling scheme for FEEL.

### 4.3.1 Federated Edge Learning

In contrast to centralized training, FL keeps the training data at each device and learns a shared global model through the federation of distributed connected devices. Keeping data locally yields many benefits, namely preserved privacy [126], reduced bandwidth use, and rapid access to data. Applying FL to wireless edge networks forms the so-called concept of federated edge learning or simply FEEL. FEEL involves a multi-access edge computing (MEC) [12] server that performs aggregation and edge devices that perform collaborative learning. The MEC server that is equipped with a parameter server

(PS) can be a next generation nodeB (gNB), or simply a base station (BS), in a wireless edge cellular network in which there are $N$ edge devices that collaboratively train a shared model.

Each device $k$ has a local dataset $D_k$ with a data size of $|D_k|$. The goal is to find the optimal global model parameters $\boldsymbol{w} \in \mathbb{R}^l$ that minimizes the average prediction loss $f(\boldsymbol{w})$:

$$\min_{\boldsymbol{w} \in \mathbb{R}^l} f(\boldsymbol{w}) = \frac{1}{D} \sum_{k=1}^{N} f_k(\boldsymbol{w}), \tag{4.1}$$

where $\boldsymbol{w}$ is the model parameter vector to be optimized with dimension $l$, $f_k(\boldsymbol{w})$ is the loss value function computed by device $k$ based on its local training data, and $D$ is the total number of data points across all devices (i.e., $D = \sum_{k=1}^{N} |D_k|$). Several models' loss functions can be trained using FEEL, such as linear and logistic regression, support vector machines, and artificial neural networks.

Ideally, all the devices independently train their local models using their local training data. Then, each one uploads its gradient updates to the server for aggregation. The server aggregates the received local updates, typically by averaging, to obtain a global model. Afterwards, the server sends the global model to the edge devices, and a new iteration begins where each device computes the gradient updates and uploads it to the server. Nonetheless, the constrained edge resources and limited communication bandwidth in wireless (edge) networks result in significant delays for FEEL. The federated averaging (FedAvg) [74] algorithm was therefore proposed to perform FEEL in a communication-efficient way. FedAvg is perhaps the most adopted communication-efficient FEEL algorithm. The main idea behind FedAvg is to select a small subset of devices and to run local epochs, in parallel, using stochastic gradient descent (SGD) on the local datasets of the selected devices. Next, all devices' resulting model updates are averaged to obtain the global model. In contrast to the naive application of SGD, which requires sending updates very often , FedAvg performs more local computation and less frequent communication updates. Since FedAvg assumes synchronous updates collection, this may result in large communication delays. In fact, the computation, storage, and communication capabilities among participating devices might be very different. Further, devices are frequently offline or unavailable either due to low battery levels, or because their resources are fully or partially used by other applications. Thus, due to the synchronous nature of FedAvg, straggler devices, i.e., devices with low performances, will cause large delays to the whole learning process.

Despite the promising theoretical results attained using FedAvg, deploying it or other FEEL algorithms in wireless edge networks is still not clear and challenging due to the fast changing nature of the network, limited resources and statistical heterogeneity. Statistical heterogeneity is a very important aspect in FEEL. In fact, most use cases of FEEL suppose that the system does not have control over participating devices and their training data. Furthermore, data distributions in user equipment depend heavily on the users' behaviour. As a result, any particular user's local dataset will not be representative of the population distribution. Additionally, the datasets are massively distributed, statistically heterogeneous, i.e., non-independent and identically distributed (non-IID) and unbalanced, and highly redundant. Moreover, the raw generated data is often privacy-sensitive as it can reveal personal and confidential information.

The wireless edge environment is composed of heterogeneous and limited capabilities of the devices [127], as well as of heterogeneous data distributions. As a result, many new considerations related to communication resources and edge devices' data should be reflected in the design and deployment of

FEEL algorithms. This motivates several works in FEEL to adapt to the communication-restrained edge environment.

### 4.3.2 Related Work

Several prior works investigated the communication-constrained FEEL systems from different perspectives. In addition to several compression [87] and partial participation [7] techniques, several mechanisms were proposed to adapt the federated training to the constrained resources. For instance, authors in [35] propose an accommodation mechanism where the numbers of global and local iterations are changed depending on the available communication and computation resources. Another suggested approach relays on collecting the updates in an asynchronous manner [107], which allows a smooth adaptation to heterogeneous resources and a flexible updates collection. However, due to stale updates' effect on learning, synchronous updates collection remains the preferred method. As a result, a common approach is to selectively schedule a subset of devices to send their updates in each communication round. For example, authors in [67] proposed a client selection algorithm to reduce the latency of the model training, where only the end devices with good communication and computation capabilities are chosen, thus avoiding the straggler's problem. Nevertheless, this method is biased toward powerful devices with better channel states, which discards devices with potentially more informative or important updates, and might lead to models that cannot generalize to a wide range of devices. To diversify the sources of updates, several works adopted scheduling algorithms that aim to maximize the number of participating devices. For instance, authors in [50] proposed an energy-efficient joint bandwidth allocation and scheduling algorithm, which ensures the training speed by collecting the maximum amount of updates possible. Nonetheless, this method does not guarantee the diversity of updates sources. Consequently, fairness measures [69, 70] were adopted in scheduling policies to ensure gradient diversity. For instance, an age-based scheduling (ABS) algorithm was proposed in [69], where higher priority is given to devices that were not selected in several previous rounds. However, relying on strict fairness-based policy may yield a low number of collected updates within a round.

Another approach for selective scheduling relies on evaluating the resulting model in an attempt to reduce the number of collected updates by removing the irrelevant ones [49]. This is achieved through measuring the significance of a local update relative to the current global model, and whether this update aligns with the collaborative convergence trend. Nonetheless, this approach may not be energy efficient, as it is applied post-training. Computing updates is an energy consuming operation, as a result, disregarded updates are synonymous with wasted energy.

Despite the variety of research progresses, the resource-efficient FEEL scheduling algorithm design with highly heterogeneous dataset distributions remains a topic that is not well addressed. This motivates our work, in which we investigate a possible direction to evaluate the potential significance of the updates through local dataset characteristics, namely size and diversity.

## 4.4 Diversity in Federated Learning

Our idea comes from the fact that many prior works in ML have imposed the diversity on the construction of training batches to improve the efficiency of the learning process [128]. Furthermore, active learning [124, 123] is premised on the idea that models can be trained with fewer data points, provided that the selected samples are diverse and more informative. In active learning, the diversity is used as a criterion for choosing informative data points for efficient ML training. However, to the best of our knowledge, this premise has never been used in FEEL prior to this work. Thus, we investigate the possibility of exploiting the different dataset properties to carefully select devices with potentially more informative datasets with less redundancy, by measuring their size and diversity. Therefore, we propose data-aware scheduling (DAS) algorithm for FEEL where these aspects are the heart of the devices selection.

The first question to be asked is what would be a good diversity measure for FEEL? Various measures of diversity are used in active learning to choose the samples that should be labeled. For instance, in an image classification problem, images that are hard to classify with high certainty are considered more informative and are selected to be labelled, and the chosen samples are selected from different classes in order to form a diverse dataset [129]. In FEEL, the device selection does not concern independent samples, and the dataset construction is not possible due to on-device processing. As a result, the diversity must be evaluated at the level of the entire dataset. Moreover, in the premise of FEEL, the labels are already known which gives the possibility to use more informed measures. For instance, we can use Shannon entropy [130] or Gini-Simpson index [131] for classification problems, to see whether a dataset has diverse examples from several classes. If a classification dataset has samples from one class only, it should be discarded during training. Additionally, sequence prediction models are sensitive to data-quality, especially missing data and null values [132, 133, 134]. Consequently, the diversity (i.e., variations of the sequence data) should be evaluated using methods such as approximate entropy (ApEn) and sample entropy (SampEn) [135].

The Gini-simpson index is a modification of the Simpson index. The Simpson index measures the probability that two samples taken at random from the dataset of interest are from the same class, it is calculated as follows:

$$\lambda = \sum_{c=1}^{C} p_c^2, \tag{4.2}$$

where $C$ is the total number of classes, and $p_c$ is the probability of the class $c$. The original Simpson index $\lambda$ represents the probability that two samples

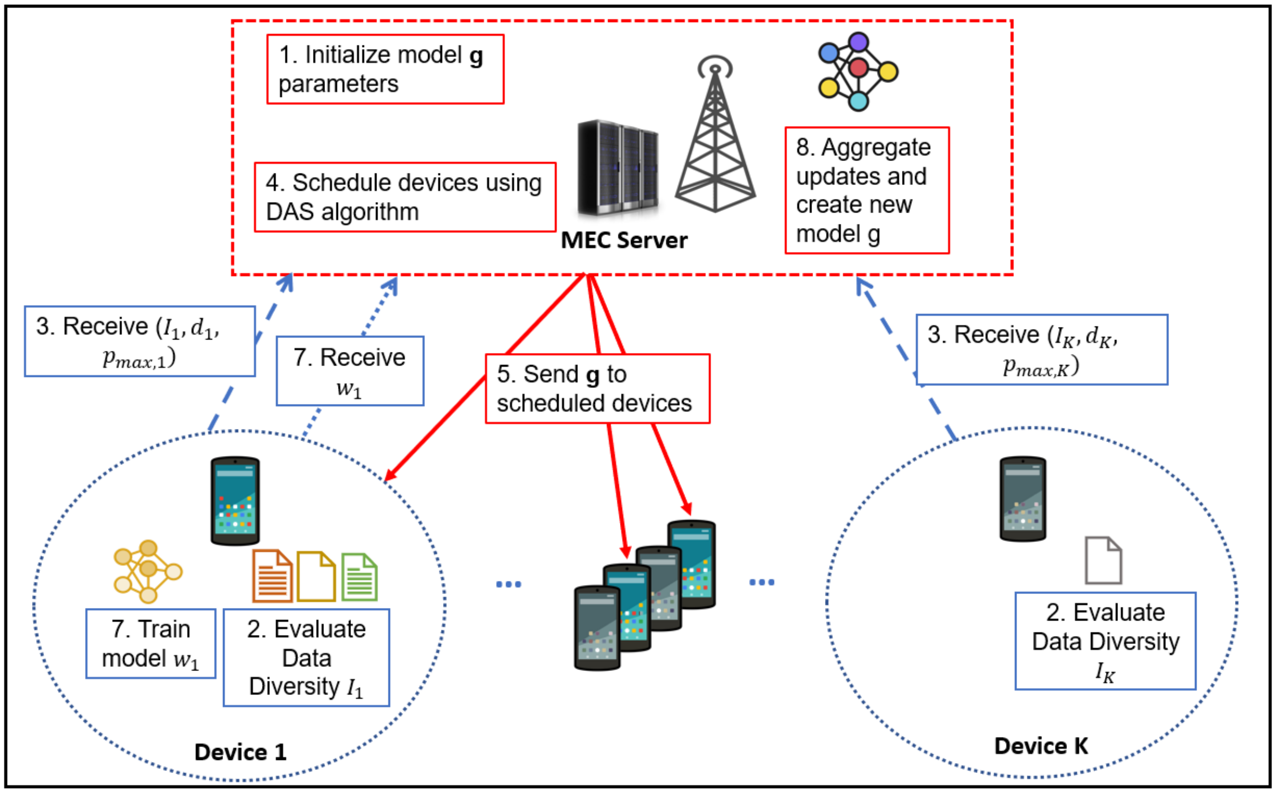

1. Initialize model **g** parameters
4. Schedule devices using DAS algorithm
MEC Server
8. Aggregate updates and create new model g
3. Receive $(I_1, d_1, p_{max,1})$
7. Receive $w_1$
5. Send **g** to scheduled devices
3. Receive $(I_K, d_K, p_{max,K})$
7. Train model $w_1$
2. Evaluate Data Diversity $I_1$
Device 1
2. Evaluate Data Diversity $I_K$
Device K

Figure 4.1 The FEEL system model. Based on the received devices' information, a subset of devices are scheduled for training the model and uploading their updates.

taken at random from the dataset are of the same type (i.e., are within the same class). The Gini–Simpson index is its transformation $1 - \lambda$, which represents the probability that the two samples belong to different classes. The Gini-Simpson index is used in different applications such as financial markets [136] and analyzing ECG signal [137].

The Shannon entropy also quantifies the uncertainty of a prediction, and was used in several applications such as text prediction and image classification [138]. In the context of FEEL, it can be used as follows:

$$H = -\sum_{c=1}^{C} p_c \log_2(p_c), \tag{4.3}$$

where $C$ is the total number of classes, and $p_c$ is the probability of the class $c$. Shannon Entropy is not defined for the extreme case of 0 samples in a class, which can be problematic in some highly unbalanced classification problems.

In sequence data, statistical measures such as the mean and the variance are not enough to illustrate the regularity, as they are influenced by system noise. ApEn was proposed to quantify the amount of regularity and the unpredictability of time-series data [139]. It is based on the comparison between values of data in successive vectors, by quantifying how many successive data points vary more than a defined threshold. A random time series with fewer data points can have a lower ApEn than a more regular time series, whereas, a longer random time series will have a higher ApEn. SampEn [140] was proposed as a modification of ApEn. It can be used for assessing the complexity of time-series data, with the advantage of being independent from the length of the vectors. Both these measures can help eliminate outliers, however, it should be noted that measuring ApEn and SampEn is a computationally heavy task, therefore it should be evaluated on a small sample rather than the entire dataset. To sum up, several diversity measures can be applied on datasets for different applications. Dataset diversity will allow a more informed participant selection in FEEL. Choosing devices with diversified datasets can accelerate the training and avoid overfitting, as the datasets contain more information.

## 4.5 System Model

Having discussed several diversity measures that can be used in various FEEL applications, let us introduce how such diversity measures can be used in the overall design of the FEEL algorithm. Hereinafter, we consider a FEEL system with multi-device and a single MEC server. The system model is illustrated in Fig. 4.1. In this section, we introduce the different elements of the system model, then we formulate a joint device selection and bandwidth allocation problem.

To study the suitability of the proposed selection criteria in the context of FEEL, we consider a wireless edge network composed of a MEC server and $K$ devices collaboratively training a shared model. Each device $k$ is characterized by a local dataset $D_k$ with a data size $|D_k|$.

### 4.5.1 Learning Model

First, the global model's architecture and weights are initialized by the MEC server. At the beginning of each training round $r$, the devices send their information and dataset diversity indicators to the MEC server. Based on the received information, alongside with the evaluated channel state information, the server selects a subset $S_r$ of the devices and allocates the necessary bandwidth to each scheduled device in order to receive the global model $g$. The scheduling of the devices, presented in Algorithm 2, is based on the trade-off between the datasets diversity and the required time and energy, under the constraint of a minimum number of devices that should be scheduled in each round. In fact, given synchronous aggregation, the MEC server requires a minimum number $K$ of updates to be collected to consider a round complete. Then, each device $k$ in the chosen subset $S_r$ uses $|D_k|$ examples from its local dataset. SGD is then used by each device $k$ to compute its local update for some period of $E$ local epochs. The updated models $w_k$ are sent to the MEC server for aggregation. Ideally, all devices transmit their trained local models to the MEC server simultaneously. The FEEL process is repeated over $r_{max}$ communication rounds and we use $D_r = \sum_{k \in S_r} |D_k|$ to denote the total size of the datasets of all selected devices. In order to aggregate the client updates, the MEC server uses the weighted average technique of the *FedAvg* algorithm proposed in [74]. The MEC server aggregates the updates and sends the resulting parameters to a new subset of selected devices.

This process is repeated until the desired prediction accuracy is reached or a maximum number of rounds is attained. In the considered architecture, the considered FEEL procedure is detailed in Algorithm 1.

**Algorithm 1** FEEL Procedure

```
1: while $r < r_{max}$ or $accuracy$ < desired accuracy do
2:     if $r = 0$ then
3:         initialize the model's parameters at the MEC server
4:     end if
5:     Receive devices information (transmit power, available data size, dataset diversity
       index)
6:     Schedule a subset $S_r$ of devices with at least $N$ devices using Algorithm 2
7:     for device $k \in S_r$ do
8:         $k$ receives model $g$
9:         $k$ trains on local data $D_k$ for $E$ epochs
10:        $k$ sends updated model $w_k$ to MEC server
11:    end for
12:    MEC server computes new global model using weighted average: $g \leftarrow \sum_{k \in S_r} \frac{D_k}{D_r} w_k$
13:    start next round $r \leftarrow r + 1$
14: end while
```

### 4.5.2 Dataset Diversity Index Design

Due to the unbalance and non-IID nature of the distributions, and under high bandwidth constraints, the dataset size and diversity need to be considered in the device selection.

Additionally, we consider a second aspect which can be viewed at the system level which is the diversity of sources. This goal can be achieved through maximizing the fairness in terms of the number of updates collected from each device is used to guarantee the diversity of the data sources.

Therefore, the goals of the devices' selection are twofold: 1) select devices with potentially informative datasets, which is achieved through evaluating the size and diversity of the datasets; and 2) guarantee that the selected devices are diversified, which will be attained by adding age-of-update to the designed diversity index.

Since our scheduling problem can consider multiple criteria, namely dataset diversity and fairness of the selection, and each measure is calculated with a function that has an output on different scales, the function should be designed to output a weighted rank value bounded in $[0, \gamma_i]$, where $i \in \{\text{dataset diversity}, \text{dataset size}, \text{age}\}$. The value of this function is given as follows: $v_i \times \gamma_i$, where $\gamma_i$ is the adjustable weight for each metric assigned by the server and $v_i$ is the normalized value of the metric $i$ calculated as follows:

$$v_i = \frac{\text{measured value of metric } i}{\text{maximum for metric } i}.$$

We define the diversity index of dataset $k$ as:

$$I_k = \sum_i v_{i,k} \gamma_{i,k}, \tag{4.4}$$

where $v_{i,k}$ is calculated for specific dataset $k$.

Note that this measure is in line with federated learning principles, as it can be evaluated on-device, and it does not reveal any privacy-sensitive information about the dataset. If fact, by combining several measures into one weighted index, it is hard for an eavesdropper to extract information about the dataset's raw elements if they intercept the index.

We formulate the first goal of the device selection problem as:

$$\max_x \sum_{k=1}^{K} I_k x_k, \tag{4.5}$$

where $x = [x_1, ..., x_K]$ and $x_k$, for $k = 1, 2, \ldots, K$, is a binary variable that indicates whether or not device $k$ is scheduled to send an update, and $I_k$ is the diversity index.

### 4.5.3 Transmission Model

As presented in Algorithm 1, the transmission aspect is also considered during the devices scheduling. Given that the bandwidth is the bottleneck of FEEL, it is essential to estimate the required transmission time and energy during the scheduling, as a means to avoid stragglers and device drop out due to low energy. Henceforth, we consider orthogonal frequency-division multiple access (OFDMA) for local model uploading from the devices to the MEC server, with total available bandwidth of $B$ Hz. We define $\alpha = [\alpha_1, ..., \alpha_K]$, where for each device $k$, $\alpha_k \in [0, 1]$ is the bandwidth allocation ratio. The channel gain between device $k$ and the BS is denoted by $h_k$. Due to limited bandwidth of the system, the bandwidth allocation ration should respect the constraints $\sum_{k=1}^{K} \alpha_k \leq 1$. The achievable rate of device $k$ when transmitting to the BS is given by:

$$r_k = \alpha_k B \log_2(1 + \frac{g_k P_k}{\alpha_k B N_0}), \qquad \forall k \in [1, K], \tag{4.6}$$

where $P_k$ is the transmit power of device $k$, and $N_0$ is the power spectral density of the Gaussian noise. Based on the synchronous aggregation assumption, the duration of a communication round depends on the last scheduled device to finish uploading. The round duration is therefore given by:

$$T = \max((t_k^{train} + t_k^{up})x_k), \qquad \forall k \in [1, K], \tag{4.7}$$

where $t_k^{train}$ and $t_k^{up}$ are, respectively, the training time and transmission time of device $k$. The training time $t_k^{train}$ depends on device $k$'s dataset properties as well as on the model to be trained. It can be estimated using Eq.4.8:

$$t_k^{train} = E\,|D_k|\,\frac{C_k}{f_k}, \tag{4.8}$$

where $C_k$(cycles/bit) is the number of CPU cycles required for computing one sample data at device $k$ and $f_k$ is its computation capacity. To send an update of size $s$ within a transmission time of $t_k^{up}$, we must have:

$$t_k^{up} = \frac{s}{r_k}. \tag{4.9}$$

Finally, the wireless transmit energy of device $k$ is given by:

$$E_k = P_k t_k^{up}. \tag{4.10}$$

### 4.5.4 Device Scheduling Problem Formulation

Considering the collaborative aspect of FEEL, and the communication bottleneck, we define the following goals for the device scheduling algorithm: From the perspective of devices, it is desirable to consume the least amount of energy to carry the training and uploading tasks. Given that the participating devices are responsible for the training, the aspect that can be adjusted is the upload energy. Therefore, the first goal is to minimize the consumed upload energy of the scheduled devices :

$$\min_{x,\alpha} \sum_{k=1}^{K} x_k E_k. \tag{4.11}$$

Due to the heterogeneous device capabilities and the largely varying data sizes, it is hard to estimate a suitable deadline for each round. To avoid stragglers' problem, it is desirable for the MEC server to have short round duration. Thus, a part of the objective is the minimization of the communication round.

$$\min_{x,\alpha} T. \tag{4.12}$$

From the perspective of accelerating learning, we adopt the goal defined in Subsection 4.5.2 in Eq 4.5.

By combining these three goals, the problem is formulated as a multi-objective optimization problem as follows:

$$\underset{x,\alpha}{\text{minimize}} \quad \left\{\sum_{k=1}^{K} x_k E_k, T, -\sum_{k=1}^{K} x_k I_k\right\} \tag{4.13a}$$

subject to

$$(t_k^{train} + t_k^{up})x_k \leq T, \qquad \forall k \in [1, K], \tag{4.13b}$$

$$\sum_{k=1}^{K} \alpha_k \leq 1, \qquad \forall k \in [1, K], \tag{4.13c}$$

$$0 \leq \alpha_k \leq 1, \qquad \forall k \in [1, K], \tag{4.13d}$$

$$\sum_{k=1}^{K} x_k \geq N, \qquad \forall k \in [1, K], \tag{4.13e}$$

$$x_k \in \{0, 1\}, \qquad \forall k \in [1, K]. \tag{4.13f}$$

where the constraints are defined as follows. Constraint (4.13b) guarantees that the scheduled devices finish training and uploading the models before the deadline. Constraints (4.13d) and (4.13c) ensure that the allocated bandwidth fractions are between 0 and 1 and that their sum does not exceed the bandwidth budget. Constraint (4.13e) guarantees that the number of selected devices is at least equal to the minimum required, with constraint (4.13f) setting $x_k$ as a binary variable.

### 4.5.5 NP-hardness

Problem 4.13a is a multi-objective problem that is non-linear. Thus, it is very challenging to solve. Even worse, we show in the following that the problem is NP-hard even for a single objective case. Indeed, a restricted version of problem 4.13a is shown to be equivalent to a knapsack problem and thus it is NP-hard. The main difficulty of the problem comes from maximizing the weighted sum of the devices while allocating the bandwidth. In this regard, when we fix the transmit power and assume that each device is allocated a certain amount of bandwidth, then the multi-objective function in (13a) reduces to, for fixed transmit powers $p_k$ and fixed $\alpha_k$, the problem of maximizing the weighted number of devices, i.e., $\sum_k I_k x_k$ subject to a knapsack capacity given by $\sum_k \alpha_k x_k \leq 1$. Constraint (4.13b) can be verified for each device to filter out the devices that do not respect them since it depends only on the fixed power. Thus, the problem is equivalent to a knapsack problem and since the latter is NP-hard, so is problem 4.13a.

## 4.6 Scheduling algorithm

In this section, we present our data-aware FEEL scheduling algorithm to solve the multi-objective problem 4.13a defined in section 4.5. Besides being NP-hard, problem 4.13a is a mixed integer non-linear multi-objective program that is hard to solve. The proposed FEEL scheduling algorithm to solve 4.13a proceeds in two main steps as follows. First, problem 4.13a is decomposed into two subproblems. Next, the proposed algorithm optimizes iteratively both subproblems.

The first sub-problem (**Sub1**) is a selection problem in which we select the devices in order to optimize a weighted linear combination of the different objectives. The selection sub-problem is formulated as follows:

$$\underset{x}{\text{minimize}} \quad \lambda_E \sum_{k=1}^{K} x_k E_k + \quad \lambda_T T \quad - \lambda_I \sum_{k=1}^{K} x_k I_k \tag{4.14a}$$

subject to

$$x_k \in \{0,1\} \qquad \forall k \in [1,K] \qquad , \tag{4.14b}$$

$$\sum_{k=1}^{K} x_k \geq K \qquad \forall k \in [1,K] \tag{4.14c}$$

where $\lambda_E$, $\lambda_T$, and $\lambda_I$ are positive scaling constants used first to scale the value of the objective function and second to combine the different conflicting objectives into a linear single one.

The second sub-problem (**Sub2**) is a bandwidth allocation problem in which the device selection decision is fixed through solving the previous selection sub-problem. The objective of the bandwidth allocation problem consists of linear combination, using a positive constant $\rho$, of the consumed energy and the round's completion time. This problem is formulated as follows:

$$\begin{aligned}
&\underset{\alpha}{\text{minimize}} \quad && \rho \sum_{k=1}^{K} x_k E_k + \quad (1-\rho)T && \text{(4.15a)}\\
&\text{subject to} && \sum_{k=1}^{K} \alpha_k \leq 1, \qquad \forall k \in [1, K], && \text{(4.15b)}\\
& && 0 \leq \alpha_k \leq 1, \qquad \forall k \in [1, K] && \text{(4.15c)}
\end{aligned}$$

To solve **Sub1**, we use *relaxation and rounding.* Specifically, we relax the integer constraint $x_k \in \{0, 1\}$ as the real-value constraint $0 \leq x_k \leq 1$, and then the integer solution is determined using rounding after solving the relaxed problem, then verifying whether the condition (4.14c) is satisfied. The relaxed problem can be written as:

$$\begin{aligned}
&\underset{x}{\text{minimize}} \quad && \lambda_E \sum_{k=1}^{K} x_k E_k + \quad \lambda_T T \quad - \lambda_I \sum_{k=1}^{K} x_k I_k && \text{(4.16a)}\\
&\text{subject to} && && \\
& && 0 \leq x_k \leq 1 \qquad \forall k \in [1, K] && \text{(4.16b)}
\end{aligned}$$

The continuous value of $x_k$ can be viewed as the selection priority of the device $k$, therefore, if the condition (4.14c) is not satisfied, we set $x_k = 1$ for the $N$ devices with highest priorities. To solve **Sub2**, we applied off-the-shelf solvers in order to obtain the optimal bandwidth allocation ratios.

The proposed FEEL algorithm is an iterative algorithm that solves each sub-problem sequentially as discussed previously and updates the solution in each iteration. The pseudo-code of the data-aware scheduling (DAS) algorithm is given in Algorithm 2. The algorithm iterates until convergence or until a maximum number of iterations is reached, whichever appears first. The convergence happens when the values of $x$ and $\alpha$ does not undergo a large change, e.g., when their values are almost the same as in the previous iteration, the loop is terminated. The number of iterations is set to $iterations_{max}$ and is used to guarantee the algorithm termination.

**Algorithm 2** DAS algorithm for FEEL

```
1: initialize $x_k = 1 \forall k \in [1, K]$;
2: uniformly allocate the bandwidth;
3: $iterations \leftarrow 1$;
4: while $iterations < iterations_{max}$ and not convergence do
5:     Solve Sub1 : Return $x$;
6:     Round $x$;
7:     if condition(4.14c) is satisfied then
8:         continue
9:         select $K$ devices with the highest priorities
10:    end if
11:    Solve Sub2 : Return $\alpha$
12:    $iterations \leftarrow iterations + 1$
13: end while
```

## 4.7 Simulation and results

In this section, we present the performance evaluation of the DAS algorithm. The experiments we conducted consist of two parts: 1) an evaluation for the proposed diversity index; and 2) an evaluation of the performance of the proposed DAS algorithm.

We first present in Section 4.7.1 the simulation environment and parameters including the wireless edge environment and the datasets and trained models. Next, we evaluate our proposed diversity index in Section 4.7.2. We compare the diversity index' impact in the selection of participant devices in comparison to random selection. Then, we evaluate DAS algorithm under different settings by comparing it to other scheduling approaches in Sections 4.7.3 and

4.7.4. As communication is the bottleneck of FEEL, sending large models over wireless networks limits the number of participants. Thus, we mainly evaluate the performance of DAS algorithm by varying the model size in Section 4.7.3. Furthermore, since trading communication rounds with more local computation is one of the key enablers of FEEL, we also study the effect of varying the number of local epochs in Section 4.7.4.

In the experiments, in order to evaluate the possible gain from using our proposed algorithm, we compare DAS to two scheduling strategies: 1) a baseline scheduling where all the devices participate in the training with optimized time and energy following problem **Sub2**. We compare to this baseline scheduling in order to evaluate the scalability of the algorithm in terms of the consumed energy and time. 2) an age-of-update based scheduling (ABS) algorithm [70, 69], specifically we used the age-based priority function proposed in [70] with $\alpha = 1$, $f(k) = \log(1 + T(k))$ with $T(k)$ the number of rounds since last selection of device $k$. This algorithm considers both the variety of participants. In fact, using ABS, the devices that did not participate in the past few rounds have more priority. Therefore, by comparing DAS to ABS, we are able to measure the importance of the dataset diversity in the algorithm, in contrast to ABS approach that only considers the diversity of participants.

### 4.7.1 Simulation Environment and Parameters

The simulations were conducted on a desktop computer with a 2,6 GHz Intel i7 processor and 16 GB of memory and NVIDIA GeForce RTX 2070 Super graphic card. We used Pytorch [141] for the machine learning library, and Scipy Optimize [142] for the optimization modeling and solver. In the numerical results, each presented value is the average of 50 independent runs.

*1) Wireless Edge Environment:*

We consider a cellular network modelled as a square of side 500 meters and composed of one BS located in the center of the square. The $K$ edge devices are randomly deployed inside the square following uniform distribution. Unless specified otherwise, the simulation parameters are as follows. We consider $K = 100$ edge devices, and $N = 1$ the minimum number of devices to be scheduled. The OFDMA bandwidth is $B = 1$ MHz. The channel gains $g_k$ between edge device $k$ and the BS includes large-scale pathloss and small-scale fading following Rayleigh distribution, i.e., $|g_k|^2 = d_k^{-\alpha}|h_k|^2$ where $h_k$ is a Rayleigh random variable and $\alpha$ is the pathloss exponent and $d_k$ is the distance between edge device $k$ and the BS. We set the parameters of problem **Sub1** as $\lambda_E = \lambda_T = \frac{1}{4}, \lambda_I = \frac{1}{2}$ and the parameters of **Sub2** as $\rho = \frac{1}{2}$. The remainder of the used parameters are summarized in Table I.

Table 4.1 Generated Values

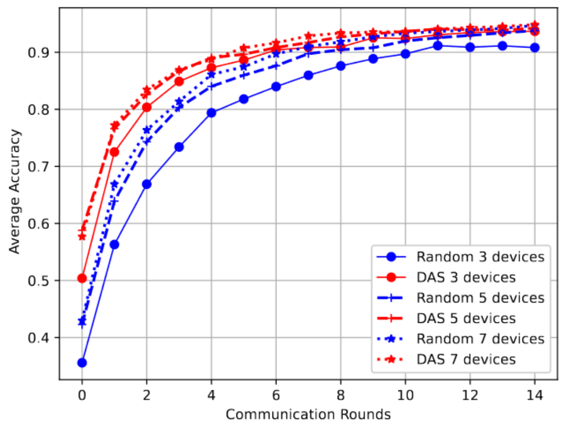
| | |
|---|---|
| Devices CPU frequency | [ 1,3 ] Ghz |
| Cycle /bit | [ 10,30 ] (cycles/bit) |
| Transmit Power | [1,5] |
| Model Size | 100 kbits |
| Bandwidth | 1MHz |
| Number of shards per device | [1,30] |

*2) Dataset and Trained Models:*

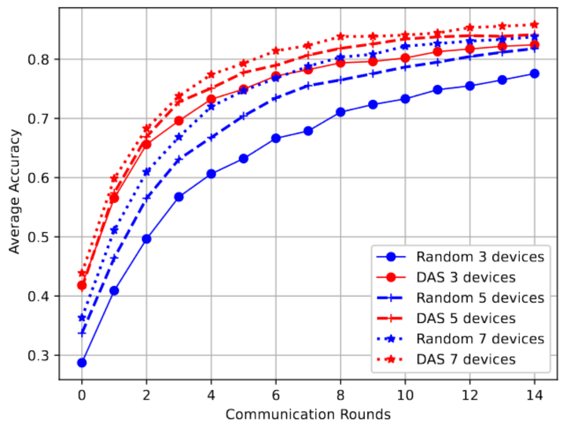
**Dataset:** We used benchmark image classification dataset MNIST [84], which we distribute randomly among the simulated devices. The data distribution we adopted is as follows: We first sort the data by digit label, then we form 1200 shards composed of 50 images each. Each shard is composed of images from one class, i.e. images of the same digit. In the beginning of every simulation run, we randomly allocate a minimum of 1 shard and a maximum of 30 shards to each of the 100 devices considered in this simulation. This method of allocation allows us to create an unbalanced and non-IID distribution of the dataset. We keep 10% of the distributed data for test, and use the remaining for training.

**Models:** convolutional neural networks (CNNs) and multi-layer perceptrons (MLPs) are widely used in classification problems. More specifically, MLPs are used for classification problems in general, while CNNs are especially powerful for image classification as they allow scale independence. We used these two different models in order to show the effect of the proposed algorithm in different models' convergence. The CNN model is given with two 5x5 convolution layers (the first with 10 channels, the second with 20, each followed with 2x2 max pooling), two fully connected layers with 50 units and ReLu activation, and a final softmax output layer. We also train a simpler MLP model with two fully connected layers. Since our goal through using these models is to evaluate our scheduling algorithm and not to achieve state-of-the-art accuracy on MNIST, therefore they are sufficient for our goal. Furthermore, the selected models are fairly small, thus they can be realistically trained on resource-constrained and legacy devices, using reasonable amounts of energy in short time windows.

We first evaluate the diversity index through several experiments. Then we evaluate DAS over wireless networks by varying the model size and the number of local epochs.

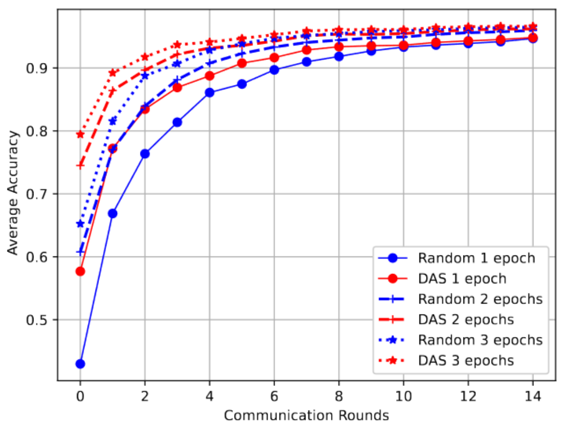


(a) CNN

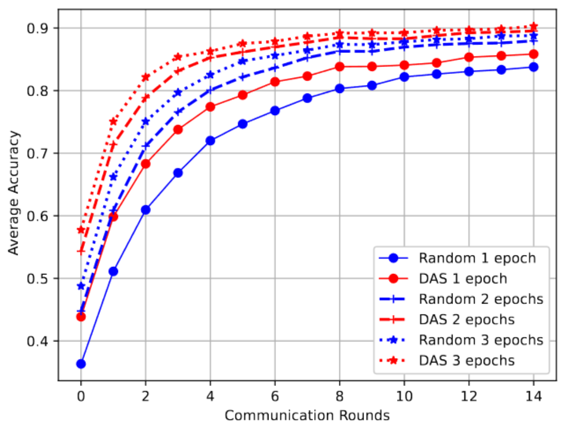


(b) MLP

Figure 4.2 Average test accuracy by varying number of selected devices

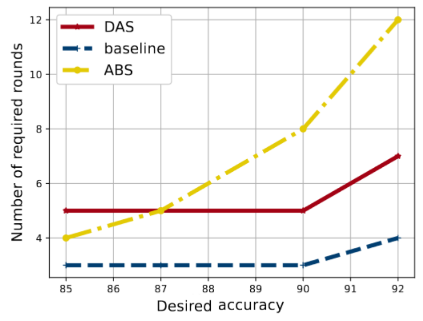


**(a)** CNN

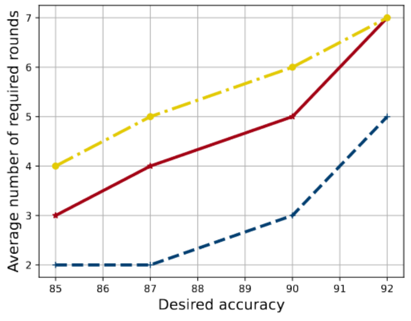


**(b)** MLP

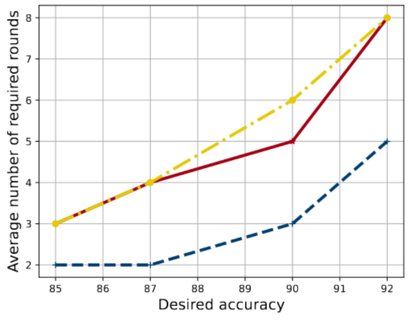
Figure 4.3 Average test accuracy by varying number of local epochs with fixed number of devices



**(a)** $s = 100k$



**(b)** $s = 150k$



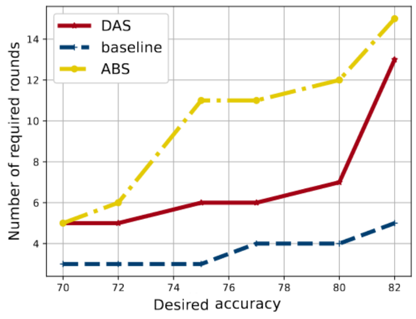
**(c)** $s = 200k$

Figure 4.4 The impact of the model size on the required number of rounds to achieve the desired accuracy using the CNN model.

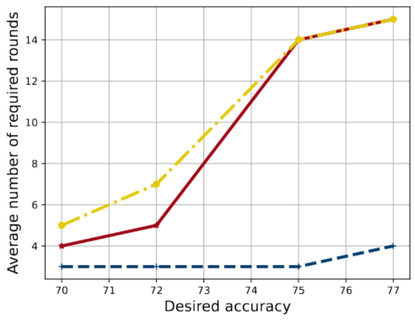
### 4.7.2 Diversity Index Evaluation

To test the proposed index, we train both models (i.e., CNN and MLP), using the FedAvg algorithm [74] over a total of 15 rounds. Since MNIST is essentially an image classification task, and we have generated highly unbalanced datasets where several devices only have a subset of the target classes, we used Gini-Simpson index to evaluate the datasets diversity. We set the weights of the index equally to 1/3. We compare DAS performance to random selection-based scheduling.

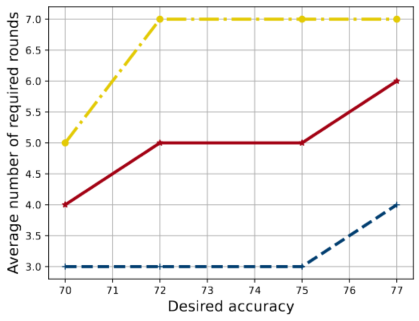
To clearly illustrate the efficiency of the diversity index in devices' selection, we stress-test the selection by limiting the number of selected devices. Fur-



**(a)** $s = 100k$

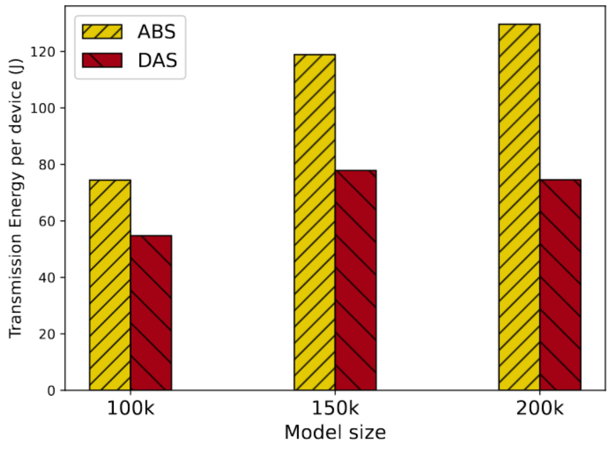


**(b)** $s = 150k$

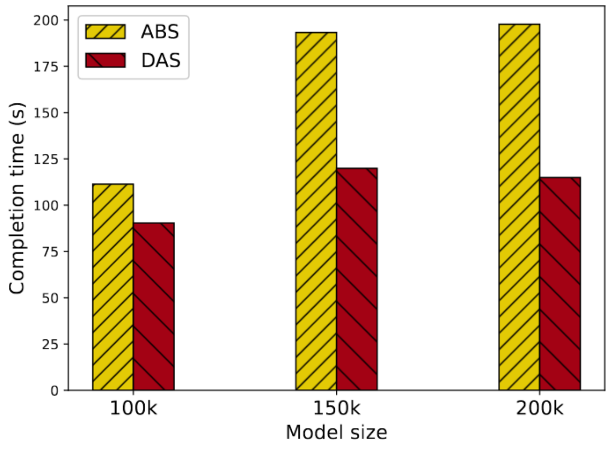


**(c)** $s = 200k$

Figure 4.5 The impact of the model size on the required number of rounds to achieve the desired accuracy using the MLP model.



**(a)** Energy per device

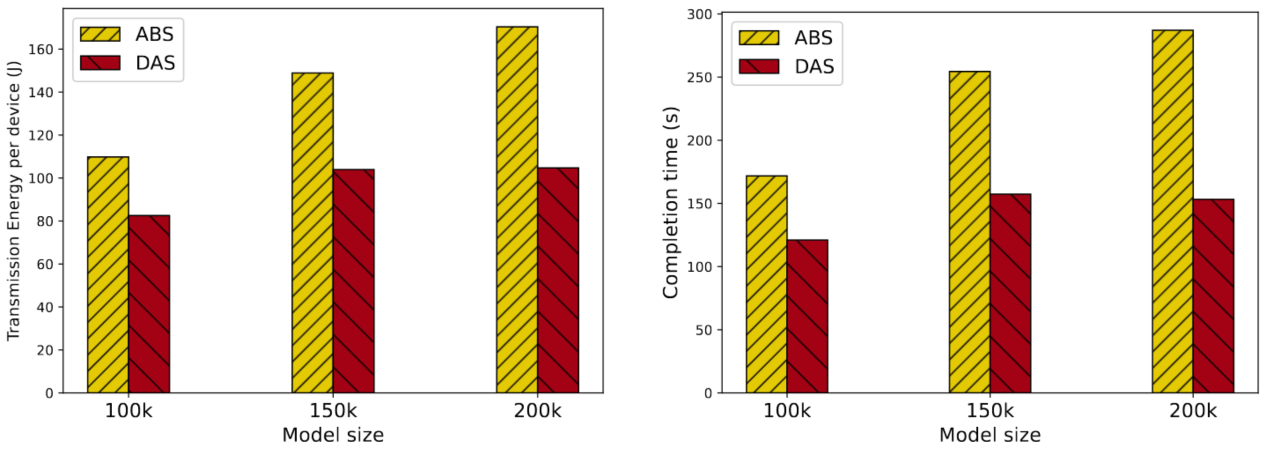


**(b)** Completion time

Figure 4.6 Energy per device and completion time for training the CNN model for a goal accuracy of 92%.

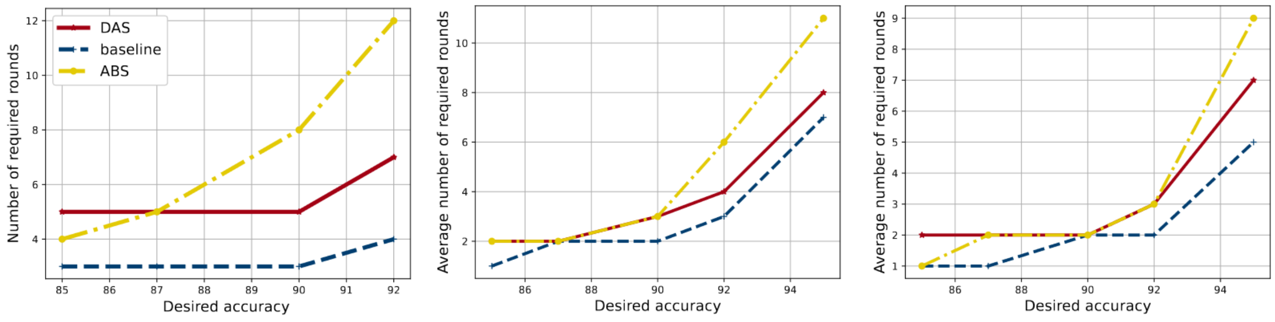
thermore, the trade-off between local update and global aggregation imposes the evaluation of the performance when varying the number of local iterations.

As a first experiment, we limited the number of selected devices to 3, 5 and 7. Fig. 4.2 shows the obtained accuracy throughout training rounds. The average accuracy obtained using DAS is significantly higher across different simulations. Thus, it is clear that using the diversity index as a criteria for scheduling devices allows to accelerate the learning significantly, especially when the number of devices that can be scheduled is low.



**(a)** Energy per device **(b)** Completion time

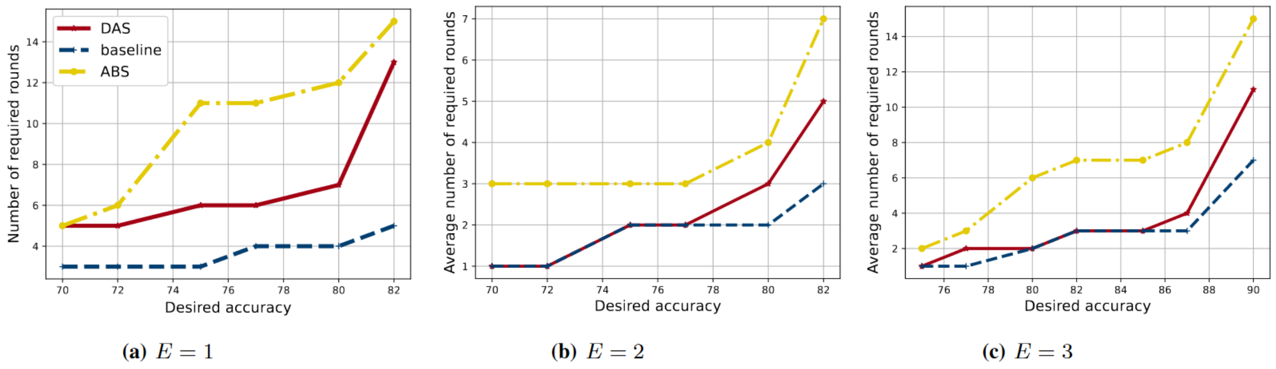
Figure 4.7 Energy per device and completion time for training the MLP model for a goal accuracy of 77%.



**(a)** $E = 1$ **(b)** $E = 2$ **(c)** $E = 3$

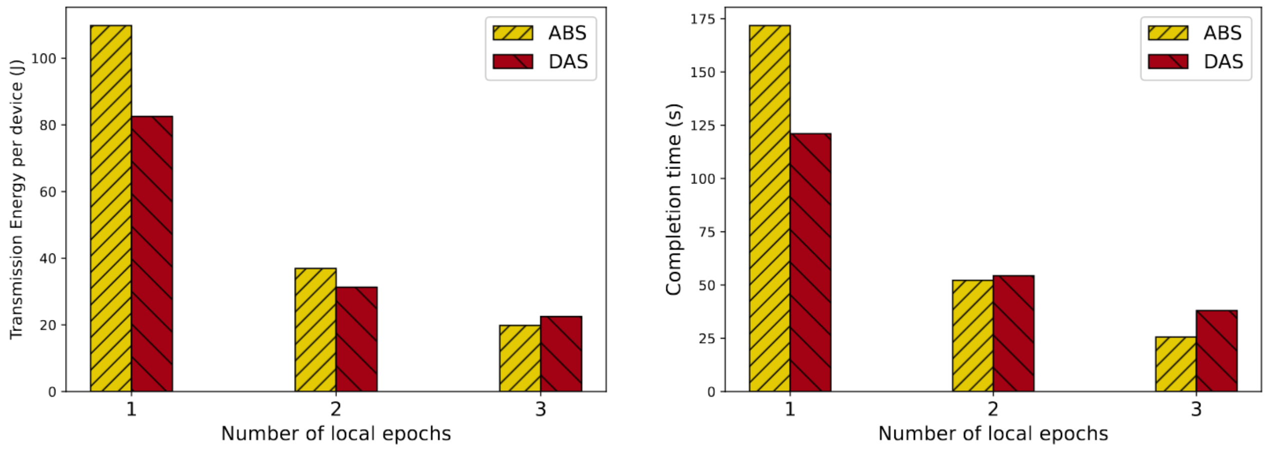
Figure 4.8 The impact of the number of local epochs on the required number of rounds to achieve the desired accuracy using the CNN model.

In a second experiment, in Fig. 4.3, we fix the number of selected devices to 7, and we vary the number of local epochs $E \in \{1, 2, 3\}$ while choosing a random scheduling algorithm or DAS algorithm. Fig. 4.3 illustrates the obtained results for the CNN and the MLP models. Adding more local computations allows significant gains in communication and in accuracy, especially for the MLP model. We notice that the data-aware device selection still surpasses random selection in these simulations when adding more local computation. All in all, these results validate our hypothesis involving the importance of dataset diversity.



**(a)** $E = 1$ **(b)** $E = 2$ **(c)** $E = 3$

Figure 4.9 The impact of the number of local epochs on the required number of rounds to achieve the desired accuracy using the MLP model.

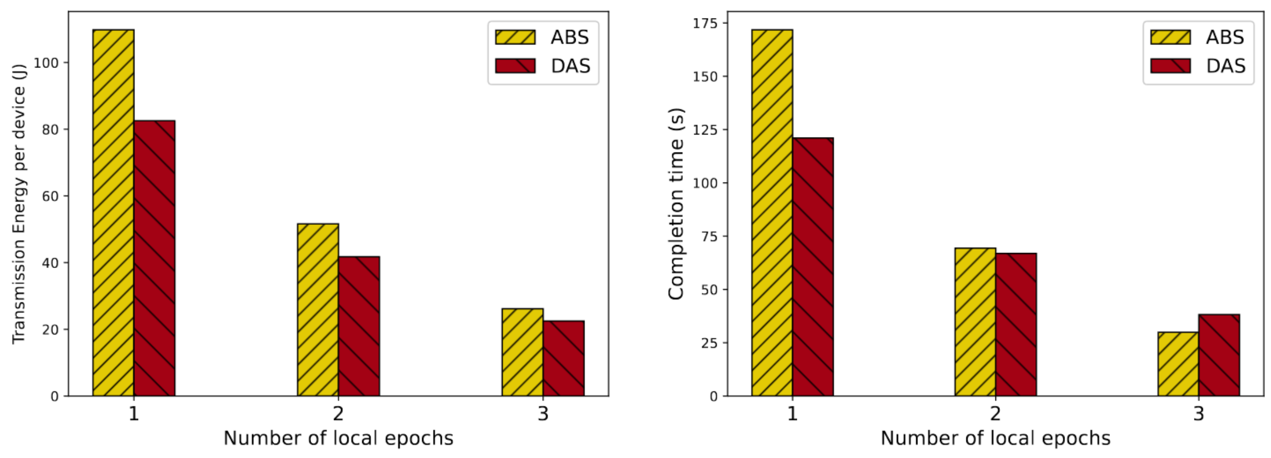


**(a)** Energy per device **(b)** Completion time

Figure 4.10 Energy per device and completion time for training the CNN model for a goal accuracy of 92%.

### 4.7.3 Effect of Model Size

Fig. 4.4 and Fig.4.5 show that using DAS algorithm, the number of required communication rounds to reach the desired accuracy is always lower than (or at least equal to) the one needed using ABS. For both models, the size can affect the convergence speed of the learning. It should be noted that ABS tends to select more devices in early communication rounds, which can yield higher accuracy than DAS similarly to what is shown in Fig.4.4.a. Nonetheless, ABS gives higher priority to devices that did not participate, which leads



**(a)** Energy per device **(b)** Completion time

Figure 4.11 Energy per device and completion time for training the MLP model for a goal accuracy of 77%

to a decrease in the number of devices that can be selected, and thus a slower convergence over the following rounds.

Indeed, the baseline scenario requires less rounds to reach high accuracy levels, nonetheless, it is hard to scale for large number of devices in rapidly changing environments. In fact, while Fig. 4.7 and Fig.4.6 show that the ABS and DAS are comparable in terms of energy and completion time, the consumed energy and time only represent a fraction of those of the baseline scenario requirements. This is mainly due to scheduling only a fraction of the devices, which for both algorithms did not exceed 20%.

Considering a goal accuracy of 77% for the MLP model and 92% for the CNN, the values in Fig.4.6 and Fig.4.7, represent gains in energy compared to the baseline as follows: On total, the consumed energy per device for the ABS represents a gain of 68,85% on average for training the MLP model and 76,56% for the CNN model. Even higher gains are achieved for DAS, where a gain of 78,86% is reached when training the MLP model and 84,96% CNN. In terms of completion time, the required time across different experiments for DAS is significantly less than the required time for ABS. These results are consistent with the number of required rounds for training presented in Fig. 4.4 and Fig.4.5. Additionally, the ABS algorithm selects more devices in the first few rounds, and much less devices in the later rounds, thus leading to longer rounds in the beginning of the training, and shorter rounds later.

These results show that the careful selection of participating devices jointly with the optimized bandwidth allocation, make DAS scalable in terms of time and energy when training large models.

### 4.7.4 Local Computation

In this section, we study the effect of increasing the computation per device. We fix $s = 100$kbits, and add more local computation per client on each round.

By increasing the number of local epochs $E$, we take full advantage of available parallelism on the client hardware, which leads to higher accuracy on the test set with less communication rounds. However, in previous work [47, 74], long local computation may lead to the divergence training loss. Furthermore, due to the changing environment due to the mobility of the devices, it is hard to plan ahead the communication rounds for large $E$. Therefore, we limit the experiments to $(1, 2, 3)$ local epochs.

Fig. 4.8 and Fig. 4.9 show that adding more local epochs per round can produce a dramatic decrease in communication costs. We see that increasing the number of local epochs $E$ can benefit largely from DAS, as it achieves a closer behaviour to the baseline, especially for the less powerful MLP model.

Fig. 4.10 and Fig. 4.11 show how trading frequent communication with more local computation has several benefits, as it reduces the required transmission energy, as well as the FEEL completion time. Similarly to the previous experiments, we considered a goal accuracy of 77% for the MLP model and 92% for the CNN, the gains in energy compared to the baseline when increasing the number of local epochs are as follows: On total, the consumed energy per device for the ABS represents gains of 83.5% and 95.97% for the MLP when training for 2 epochs and 3 epochs respectively 88.19%, 96,95% on average for training the CNN model. Even higher gains are achieved for DAS, where a gain of 86.65% and 96.54% is reached when training the MLP model and 90% and 96,54% for the CNN model. These results are consistent with the fraction of selected devices which does not exceed 20% on average.

We noticed that when increasing the number of local epochs, the required completion time using DAS becomes slightly higher than the required completion time for ABS. This is mainly due to prioritizing devices that have larger datasets, which leads to longer training duration. Nevertheless, the difference can be seen as marginal when considering the gain in energy.

These results show that the careful selection of participating devices and the resource allocation, make the DAS scalable.

## 4.8 Conclusion

In this paper, we have investigated the problem of devices scheduling in federated edge learning by formulating the following question: Can the use of a suitable diversity index help achieve a better accuracy in fewer rounds? To answer this question, we consider data properties as the key motor of the selection of devices, as we designed a diversity index which can be adapted to a wide variety of use-cases. Additionally, we integrated the diversity index in a novel scheduling strategy in wireless networks, where the completion time and energy efficiency of the transmission are also of high importance. To this end, we derived the time and energy consumption models for FEEL based on devices and channels properties. With these models, we have formulated a joint selection and bandwidth allocation problem, aiming to minimize a multi-objective function of the completion time and the total transmission energy, while balancing with a goal to maximizing the diversity of the selected devices. We have proposed to solve this problem through an iterative algorithm that starts with the selection of the devices and then allocates the bandwidth. Through extensive evaluations, we proved the importance of the data properties in FEEL and the efficacy of the diversity index. Furthermore, we showed that our proposed scheduling algorithm can effectively reduce the number of required rounds to achieve high accuracy levels especially for large models, which results also in savings in time and energy.

# Chapitre 5: Avant-propos

**Auteurs et affiliation:**

Afaf Taïk: étudiante au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Hajar Moudoud: étudiante au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Soumaya Cherkaoui: Professeure, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

**Date de l'acceptation:** Juin 2021.

**État de l'acceptation:** version finale publiée.

**Revue:** IEEE Conference on Local Computer Networks (LCN) 2021.

**Titre français:** Ordonnancement des dispositifs basé sur la qualité des données pour l'apprentissage fédéré en périphérie.

**Résumé français:**

L'apprentissage fédéré à la périphérie du réseau (FEderated Edge Learning FEEL) s'est imposé comme une technique de référence pour l'apprentissage machine (en anglais machine learning ML) distribué et privé. Dans ce cadre, les dispositifs périphériques entraînent collectivement des modèles de ML sous l'orchestration d'un serveur. Cependant, en raison des communications fréquentes, FEEL doit être adapté à la bande passante de communication limitée. En outre, l'hétérogénéité statistique des distributions des données locales et l'incertitude quant à la qualité des données posent des problèmes

importants pour la convergence des modèles. Par conséquent, une sélection méticuleuse des dispositifs participants et une allocation analogue de la bande passante sont nécessaires. Dans cet article, nous proposons un algorithme d'ordonnancement basé sur la qualité des données (en anglais Data-Quality based Scheduling DQS) pour FEEL. DQS donne la priorité aux dispositifs fiables avec des ensembles de données riches et variés. Dans cet article, nous définissons les différents composants de l'algorithme d'apprentissage et l'évaluation de la qualité des données. Ensuite, nous formulons le problème de la sélection des dispositifs et de l'allocation de la bande passante. Enfin, nous présentons notre algorithme DQS pour FEEL, et nous évaluons sa performance dans différents scénarios d'empoisonnement des données.

# Chapitre 5: Foreword

**Authors and affiliation:**

Afaf Taïk: Ph.D. Student, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Hajar Moudoud: Ph.D. Student, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Soumaya Cherkaoui: Professor, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

**Date of acceptance:** June 2021.

**Acceptance status:** Final version published.

**Conférence:** IEEE Conference on Local Computer Networks (LCN) 2021.

**Title:** Data-Quality Based Scheduling for Federated Edge Learning [1]

1. https://github.com/afaf-taik/ReputationFL

# CHAPTER 5

# Data-Quality Based Scheduling for Federated Edge Learning

## 5.1 Abstract

FEderated Edge Learning (FEEL) has emerged as a leading technique for privacy-preserving distributed training in wireless edge networks, where edge devices collaboratively train machine learning (ML) models with the orchestration of a server. However, due to frequent communication, FEEL needs to be adapted to the limited communication bandwidth. Furthermore, the statistical heterogeneity of local datasets' distributions, and the uncertainty about the data quality pose important challenges to the training's convergence. Therefore, a meticulous selection of the participating devices and an analogous bandwidth allocation are necessary. In this paper, we propose a data-quality based scheduling (DQS) algorithm for FEEL. DQS prioritizes reliable devices with rich and diverse datasets. In this paper, we define the different components of the learning algorithm and the data-quality evaluation. Then, we formulate the device selection and the bandwidth allocation problem. Finally, we present our DQS algorithm for FEEL, and we evaluate it in different data poisoning scenarios.

## 5.2 Introduction

Training centralized machine learning (ML) models requires the collection of large datasets from sparse and highly distributed sources. Nonetheless, in addition to the communication overheads, data collection raises an increasing concern about sharing private information. FEderated Edge Learning (FEEL) [143, 144] is a ML setting that utilizes multi-access edge computing

(MEC) to tackle these concerns. In contrast to centralized ML, FEEL consists of training the model on the user equipment (UE) under the orchestration of a MEC server, where only the resulting parameters are sent to the edge servers to be aggregated.

The predominant FEEL system model is a synchronous time-division resource allocation system [145]. This means that the MEC server waits for all UEs to send their updates before aggregating them. However, in reality, not all updates can be aggregated. In fact, updates from UEs who do not report before a fixed deadline are cancelled. A real deployment of FEEL is therefore subject to the following challenges:

**Limited Resources**: In a contrast to cloud servers, the computing and storage resources at the edge are rather limited [146, 114]. The gap in computational resources of the UEs causes significant delays due to stragglers. Furthermore, to run the iterative FEEL learning, the Base Station (BS) generally needs to connect a large number of UEs across a resource-limited spectrum and therefore can only support a limited number of UEs sending their model updates over unreliable channels for global aggregation. Additionally, the size of the updates can be very large in the case of deep neural networks. As a result, the communication overhead becomes a bottleneck for FEEL.

**Unbalanced and non-IID**: Unlike traditional ML systems, in which an algorithm operates on a large data set distributed homogeneously over several servers in the cloud, FEEL is typically trained on a large, often unbalanced, and non-identically distributed (non-IID) dataset that is generated by separate distributions across different UEs. A large number of UEs participate in the training, where several own small datasets, which makes local models prone to overfitting. Moreover, the dataset of a given UE is usually based on a particular user's usage pattern, and thus a particular user's local data set will likely be redundant and not representative of the remainder of the population distribution.

**Unreliable and malicious devices**: Although FEEL has improved privacy protection, it still suffers from several problems. First, all devices involved in the FEEL process are expected to contribute their resources unconditionally, which is not accepted by all UEs. Without reward, only a small fraction of the devices are willing to participate in the training process. On the other hand, the UEs involved in the training are unreliable and may act maliciously, intentionally, or unintentionally, which may affect the overall model and lead to erroneous model updates.

Most existing work on FEEL proposed scheduling algorithms aiming to optimize resource utilization. The considered resources are time [147], transmission energy of participating UEs [148], and local computation energy [149]. Consequently, the number of scheduled UEs is often restricted in order to meet latency and energy constraints. However, optimization should consider learning related aspects, especially data distributions and quality [147, 150, 151]. Furthermore, providing a reliable device selection is a necessary stepping stone for enabling efficient and useful updates [152].

The main motivation behind this paper is to design a scheduling algorithm for FEEL that considers UEs datasets' different properties at its heart. In fact, while datasets are unbalanced and non-IID, but a UE with a large dataset is not necessarily in possession of useful data. Furthermore, malicious UEs can train the model on poisoned datasets, which requires additional mechanisms to reduce their damages to the training process. In this paper, we consider diversity of the datasets and the reliability of the UEs as the baseline criterion for choosing participating devices in FEEL. The diversity evaluation allows to give priority to UEs with potentially more informative datasets to speed up the training process, and the reputation mechanism sanctions malicious UEs with poisoned datasets. Accordingly, we propose a method for incorporating datasets' properties in FEEL scheduling, by designing a novel data-quality value measure, and designing a data-quality based scheduling algorithm (DQS).

The contributions of this paper can be summarized as follows:

1) we design a suitable priority indicator for the selection of participating UEs;
2) we formulate a joint device selection and bandwidth allocation problem taking into account data properties;
3) we prove that the formulated problem is NP-hard and we propose a DQS scheduling algorithm based on greedy knapsack; and
4) we evaluate the proposed data-quality value measure and the DQS algorithm through insightful simulations.

The remainder of this paper is organized as follows. In Section II, we present the background for FEEL and related work. In Section III, we present the design of the proposed diversity measure, starting with the used uncertainty measures and their integration in FEEL. In Section IV, we integrate the proposed measure in the design of the joint selection and bandwidth allocation algorithm. Simulation results are presented in Section V. At last, conclusions and final remarks are presented in Section VI.

## 5.3 Related work

In this section, first we discuss the related work regarding device scheduling and data quality in FEEL. Next, we illustrate the existing research gaps and motivate the need for designing a data-quality based scheduling algorithm for FEEL.

Among the first works to explore device selection is work in [67], where authors mitigated the straggler problem by giving priority to end devices with good communication and computation capabilities. However, this method is not suitable in FEEL context, where datasets' distributions vary, as it discards UEs with larger and potentially more informative datasets. Several works proposed scheduling algorithms that aim to maximize the number of participating devices while optimizing used resources [153, 154]. For instance, authors in [50] proposed an energy-efficient scheduling algorithm aiming to collect the maximum number of updates possible as a guarantee for the training speed. Another example is in [69], where authors study different scheduling strategies based on the wireless channel's state. Other works focus on the staleness of updates to calculate the priority [69, 155], where higher priority is given to UEs that did not participate in previous rounds or have stale updates. Despite the variety of the scheduling algorithms in the literature, the design and evaluation of FEEL scheduling under heterogeneous and uncertain dataset distributions remains a topic that is not well addressed, as data-quality issues are overlooked and under-explored.

To overcome the reliability problem, several FL works use reputation systems for UEs selection, as UEs with high reputation are more likely to provide high quality data for the learning procedure. For instance, the authors in [156] proposed a decentralized peer-to-peer approach to overcome the problem of unreliable UEs, where poisoning is prevented by verifying peer contributions to the model. However, due to stastical heterogeneity and high communication costs, P2P verification may not always be suitable. In [157] and [158], the authors presented a blockchain-based reputation system for FL, where a UE's reputation is traceable. In [158], the authors presented a reputation-based UE selection using subjective logic. This solution evaluates the reputation of UEs based on their past interactions with other task publishers in the network. Nonetheless, such approach relies on the willingness of several other similar task publishers to share their opinions. Based on our analysis of these works, we found that most of them did not consider the data heterogeneity when evaluating device reputation, and might not be suitable for realistic FEEL scenarios. Consequently, we found it necessary to build a scheduling algorithm for FEEL that considers UEs datasets' different properties (*i.e.*, limited bandwidth, unbalanced/ non-IID data) alongside UEs reliability.

## 5.4 System Model

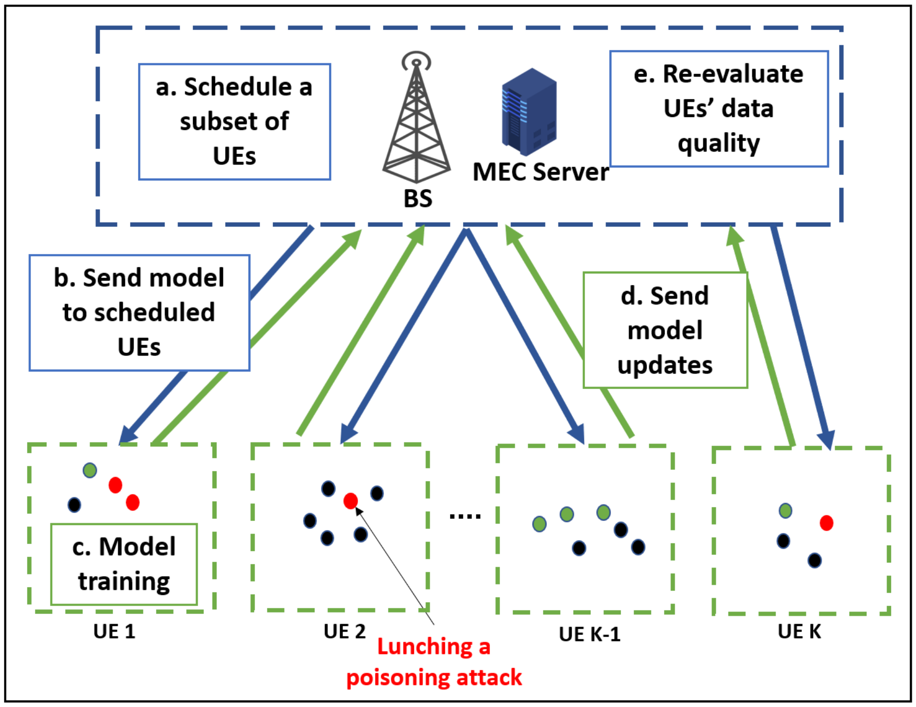

a. Schedule a subset of UEs
BS
MEC Server
e. Re-evaluate UEs' data quality
b. Send model to scheduled UEs
d. Send model updates
c. Model training
....
UE 1
UE 2
UE K-1
UE K
Lunching a poisoning attack

Figure 5.1 Illustration of a communication round of FEEL with $K$ UEs. U$E_2$ launches a poisoning attack which will affect the learning.

In this section, we introduce the different components of the system model. The system's main components are the learning model, the data-quality model, and the communication model. Based on these components, we formulate a joint UE selection and bandwidth allocation problem for FEEL.

### 5.4.1 Learning Model

In contrast to centralized training, FEEL keeps the training data locally, and learns a global model through the shared parameters sent by a federation of participants (e.g., individual users, organizations). By keeping data locally, the training can use ephemeral data and leverage the computing resources of the sparse UEs. FEEL is an iterative process that starts with the model and parameters initialization. In our proposed framework, in each round, the procedure in Algorithm 3 is repeated. Fig. 5.1 illustrates the different steps in the procedure taking into account the statistical heterogeneity of the data distributions alongside the possibility that malicious devices may launch data poisoning attacks. In the figure, $UE_2$ launches a label-flipping attack. Due to the local nature of the training, this attack is hard to detect. As a result, in our proposed process, each training round $t$ starts by the UEs sending their information to the MEC server (i.e., dataset information). With the evaluated channel state information, and the dataset information, the MEC server schedules a set of UEs using Algorithm 4 and sends them the global model $g$. Upon receiving the updates, the MEC server evaluates the quality of each model $\Omega_k$ by testing it on publicly available data. UEs also report local accuracy of their models. These evaluations are then used to update the perceived reputation of the UEs at the MEC server level. The last step is the model aggregation, which is typically achieved using weighted aggregation [143].

### 5.4.2 Data Quality and Data Poisoning

*1) Data Poisoning in FEEL* In FEEL, UE data may include personal information (*e.g.*, home address, credit card number, etc.). The disclosure of this data is not only harmful to the UE, but the intentional/unintentional alteration of the data can also cause security problems [159]. One of the main types of attacks that can affect model severally is poisoning attack. A poisoning attack occurs when the attacker is capable of injecting false data or altering the training samples of the FEEL model's learning pool, and thus causing it to learn on wrong or erroneous data. Poisoning attacks can occur during the training period and are primarily aimed at availability or integrity of the data. Generally, there are two main approaches to generate poisoned attacks, namely: label-flipping [160] and backdoor [161]. In this paper, we mainly focus on label-flipping where an adversary modifies a small number of examples/training data and maintains the characteristics of the data unchanged to degrade the performance of the model.

*2) Reputation evaluation:* At each communication round $t$, each participating UE $k$ reports the local accuracy $acc_k^{local}$ and sends its newly computed model $\Omega_k$. The model is then evaluated on a test-set to evaluate its quality, alongside the UE's honesty. A UE's reputation decreases when it uploads a malicious or bad-quality model, or when it declares a very high local accuracy compared to the obtained accuracy on the test-set. This reputation measure allows to detect not only malicious UEs, but also UEs with overfitting models and low quality dataset. We update the value of the reputation of UE $k$ in each round $t$, as follows:

$$R_k^t = R_k^{t-1} - \eta(\beta_1(acc_k^{local} - avg(acc)) + \beta_2(acc_k^{local} - acc_k^{test})) \tag{5.1}$$

where $\eta \in [0.1]$ is the reputation rate with which we decrease the reputation value.

*3) Dataset diversity evaluation:*

Each dataset can be characterized by how diverse its elements are (i.e., diversity of the elements), their number(i.e., dataset size) and how many times the model was trained on it (i.e., age). We set the value of each metric as [151]: $v_i\gamma_i$, where $\gamma_i$ is the adjustable weight for each metric assigned by the server and $v_i$ is the normalized value of the metric $i$. Using the aforementioned characteristics, the diversity index of dataset $k$ can be defined as:

$$I_k = \sum_i v_{i,k}\gamma_i, \tag{5.2}$$

with $i \in \{\text{elements diversity}, \text{dataset size}, \text{age}\}$ Depending on the application, other quality measures can also be used. For instance, in image classification, measures regarding image quality (e.g., blur) can heavily affect the learning [110, 44]. Such measures can be evaluated on a subset of images and normalized for a seamless integration in the index.

*4) Data-Quality measure* The data-quality value can be inferred from the reputation of UE $k$'s reputation and dataset diversity index. We define the value for each UE as:

$$V_k = \omega_1 R_k + \omega_2 I_k \tag{5.3}$$

with $\omega_1, \omega_2 \in [0, 1]$ are weighted values for each metric.

### 5.4.3 Communication and Computation Models

As communication is the bottleneck of synchronous FEEL, it is crucial to judiciously allocate the bandwidth. Hereinafter, we consider orthogonal frequency-division multiple access (OFDMA) for local model uploading from the UEs to the MEC server, with total available bandwidth of $B$ Hz. We define $\alpha = [\alpha_1, ..., \alpha_K]$, where each UE $k$ is allocated a fraction $\alpha_k \in [0, 1]$ from the total bandwidth $B$. Additionally, we denote the channel gain between UE $k$ and the BS by $h_k$. The achievable rate of UE $k$ when transmitting the model to the BS is given by:

$$r_k = \alpha_k B \log_2(1 + \frac{g_k P_k}{\alpha_k B N_0}), \qquad \forall k \in [1, K], \tag{5.4}$$

where $P_k$ is the transmit power of UE $k$, and $N_0$ is the power spectral density of the Gaussian noise.

Based on the synchronous aggregation assumption with a fixed deadline, the scheduled UEs in a communication round must upload before a deadline $T$. For all UEs, the time constraint is defined as follows:

$$(t_k^{train} + t_k^{up}) x_k \leq T, \qquad \forall k \in [1, K], \tag{5.5}$$

Table 5.1 List of Notations.

| **Notations** | **Description** |
|---|---|
| $T$ | Communication round's deadline |
| $B$ | Bandwidth |
| $s$ | Model size |
| $\epsilon$ | Number or local epochs |
| $g$ | Global model |
| $\Omega_k$ | Model update of UE $k$ |
| $t_k^{train}$ | Training time for UE $k$ |
| $r_k$ | Achievable data rate of UE $k$ |
| $t_k^{up}$ | Upload time for UE $k$ |
| $P_k$ | Transmit power of UE $k$ |
| $\alpha_k$ | Bandwidth fraction allocated to UE $k$ |
| $N_0$ | Power spectral density of the Gaussian noise |
| $\|D_k\|$ | UE $k$'s dataset size |
| $\zeta_k(cycles/bit)$ | The number of CPU cycles |
| $f_k$ | The computation capacity |
| $R_k^t$ | Reputation of UE $k$ at round $t$ |
| $I_k$ | Dataset diversity index of UE $k$ |
| $V_k$ | Data-quality value of UE $k$ |
| $\eta$ | Reputation rate |
| $\beta_i, \omega_i, \gamma_i$ | Weights of different measures |

where $t_k^{train}$ and $t_k^{up}$ are, respectively, the training time and upload time of UE $k$. The training time $t_k^{train}$ depends on UE $k$'s dataset size as well as on the model. It can be estimated using Eq.5.6:

$$t_k^{train} = \epsilon \, |D_k| \, \frac{\zeta_k}{f_k}, \tag{5.6}$$

where $\zeta_k$(cycles/bit) is the number of CPU cycles required for computing one sample data at UE $k$ and $f_k$ is its computation capacity.

In order to send an update of size $s$ within a transmission time of $t_k^{up}$, we must have:

$$t_k^{up} = \frac{s}{r_k}. \tag{5.7}$$

### 5.4.4 Joint UE selection and bandwidth allocation problem

Considering the data-quality aspect at the key selection criterion, and taking into account the communication bottleneck of FEEL, we formulate the following problem:

$$\underset{x,\,\alpha}{\text{maximize}} \quad \sum_{k=1}^{K} x_k V_k \tag{5.8a}$$

$$\text{subject to}$$

$$(t_k^{train} + t_k^{up})x_k \leq T, \qquad \forall k \in [1, K], \tag{5.8b}$$

$$\sum_{k=1}^{K} \alpha_k \leq 1, \qquad \forall k \in [1, K], \tag{5.8c}$$

$$0 \leq \alpha_k \leq 1, \qquad \forall k \in [1, K], \tag{5.8d}$$

$$x_k \in \{0, 1\}, \qquad \forall k \in [1, K]. \tag{5.8e}$$

The goal of the problem 5.8a is to select UEs to participate in a training round, and allocate the bandwidth to these UEs so as they upload their models before the deadline. The goal is therefore to maximize the weighted sum of the selected UEs under time and bandwidth constraints. In fact, constraint (5.8b) guarantees that the selected UEs will finish training and uploading before the deadline $T$. Due to limited bandwidth budget, the bandwidth allocation ratio should respect the constraint (5.8c). Constraints (5.8d) determines the bounds for the bandwidth allocation ratios and constraint (5.8e) defines $x_k$ as a binary value for each UE.

Problem 5.8a is mixed integer non-linear problem, which makes it very challenging to solve. Indeed, a restricted version of the problem 5.8a can be shown to be equivalent to a knapsack problem and thus it is NP-hard. In fact, the problem of maximizing the weighted number of devices, i.e., $\sum_k V_k x_k$ is subject to a knapsack capacity (i.e., limited bandwidth) given by $\sum_k \alpha_k x_k \leq 1$. Hence, the problem is equivalent to a knapsack problem, and since the latter is NP-hard, so is the problem 5.8a.

**Algorithm 3** Data-quality based training procedure

```
1: for $t \in [1 \dots t_{max}]$ do
2:     if $t = 0$ then
3:         initialize the model's parameters at the MEC server
4:         initialize the reputation values as 1 for each UE.
5:     end if
6:     Receive UEs information (transmit power, available data size, dataset diversity in-
       dex)
7:     Schedule a subset $S_t$ of UEs with at least $N$ UEs using Algorithm 4
8:     for UE $k \in S_t$ do
9:         $k$ receives model $g$
10:        $k$ trains on local data $D_k$ for $\epsilon$ epochs
11:        $k$ sends updated model $\Omega_k$ to MEC server and reports local accuracy $acc_k^{local}$
12:    end for
13:    MEC server computes new global model using weighted average: $g \leftarrow \sum_{k \in S_t} \frac{D_k}{D_t} \Omega_k$
14:    MEC server updates the reputation values using the received models using a test-set
       and the average reported accuracies.
15:    start next round $t \leftarrow t + 1$
16: end for
```

## 5.5 Data-Quality based Scheduling

In this section, we present our proposed solution for data-quality based scheduling to solve 5.8a. As problem 5.8a can be transformed into a knapsack problem, we chose to follow a greedy knapsack algorithm to solve it. A greedy knapsack algorithm has low complexity and will allow fast and efficient scheduling under the rapidly changing wireless edge environment. In fact, while the ranked list solution of the knapsack approach might enhance scheduling performance in terms of overall data quality, it is not efficient in defining the optimal bandwidth allocation decision. Moreover, selecting the highest ranked UE with the best data-quality value without considering the extent of bandwidth required by each UE does not lead to an optimized performance benefit. Consequently, it is preferable to follow a greedy algorithm that iteratively selects a UE with better ratio $\frac{V_k}{\alpha_k}$ [162].

As the required bandwidth is not a fixed value, we calculate the minimum required bandwidth for each UE to be able to send the update by the deadline $T$. In order for a UE $k$ to upload the model before the deadline $T$, each selected UE must respect $t_k^{up} \leq T - t_k^{train}$. The corresponding minimum data rate is therefore $r_{k,min} = \frac{s}{T - t_k^{train}}$ for each UE $k$. Calculating the value of $\alpha_k$ from this expression is not possible. Thus, we estimate the cost of allocating a fraction of the bandwidth to UE, by calculating the required number of fractions needed if we were to uniformly allocate the bandwidth. More specifically, we define

$$r_k(c) = \frac{c}{K} B \log_2(1 + \frac{g_k P_k}{\frac{c}{K} B N_0}), c \in \mathbb{N} \cap [1, K] \tag{5.9}$$

As a result, we define the cost for bandwidth allocation for UE $k$ $c_k = min\{r_k(c) \geq r_{k,min}, c \in \mathbb{N} \cap [1, K]\}$. The cost is then used to order the UEs based on a new ratio $V_k/c_k$, as well as to allocate the bandwidth. The DQS is presented in Algorithm4.

## 5.6 Numerical results

### 5.6.1 Experimental Setup

We conduct the simulations on a desktop computer with a 2,6 GHz Intel i7 processor and 16 GB of memory and NVIDIA GeForce RTX 2070 Super graphic card. We used Pytorch [141] for the ML library. The presented numerical results are the average of 10 independent runs in each setting.

**Dataset and Data poisoning:** We evaluate the efficiency of DQS on MNIST [84], a handwritten digit images dataset. The dataset contains 50,000 training samples and 10,000 test samples labeled as 0-9.

**Algorithm 4** Data-quality based scheduling algorithm

```
Input  A queue of $K$ UEs waiting for scheduling total available bandwidth $B$;
Output  Array $\alpha$, Scheduling decision of the UEs $x = [x_1, \ldots, x_K]$;
1: // Cost Evaluation
2: for $k = 1, \ldots, K$ do
3:     $r_k = 0, c = 1$;
4:     while $r_k(c) \leq r_{k,min}$ and $c \leq K$ do
5:         $c \leftarrow c + 1$;
6:         $c_k \leftarrow c$;
7:     end while
8: end for
9: return  $C = [c_1, \ldots, c_K]$
10: // Bandwidth Allocation
11: order UEs according to their ratio $(\frac{V_k}{c_k})$ decreasingly and index them from 1 to $K$;
12: for $k = 1 \ldots K$ do
13:     $x_k \leftarrow 0$;
14: end for
15: $A \leftarrow K$;
16: $k \leftarrow 1$;
17: while $A \geq 0$ do
18:     if $A - c_k \geq 0$ then
19:         $x_k \leftarrow 1$;
20:         $A \leftarrow A - c_k$;
21:         $\alpha_k \leftarrow c_k/K$;
22:         $k \leftarrow k + 1$;
23:     end if
24: end while
25: return  $x$ and $\alpha$
```

*Data distribution:* In order to simulate non-iid and unbalanced datasets, after keeping 10% of the data for test, the data distribution we adopted for training is as follows: We first sort the data by digit label, then we form 1200 groups of 50 images. Each group contains images of the same digit. In the beginning of every simulation run, we randomly allocate a minimum of 1 group and a maximum of 30 groups to each of the 50 UEs.

*Data poisoning:* According to work in [163, 160], when launching targeted poisoning attacks (i.e., label flipping attack) on a handwritten digits classifier, the easiest and hardest source and target label pairs are (6,2) and (8,4),

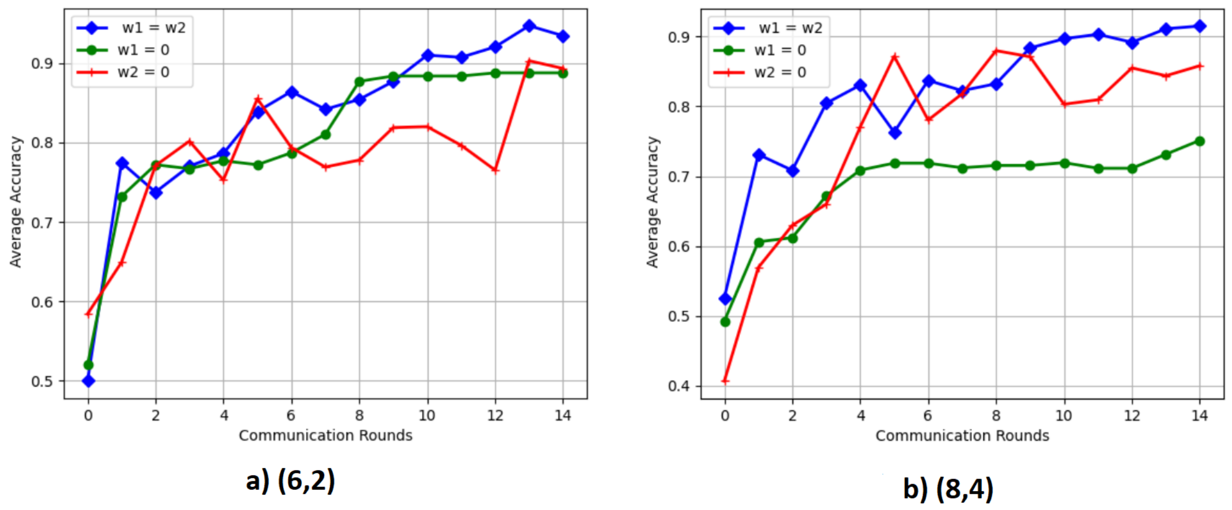


Figure 5.2 Model accuracy depending on the targeted label following different selection strategies

respectively. Accordingly, we study both targeted labels. In fact, in each run, 5 UEs chosen at random will display a malicious behaviour by poisoning data through label flipping.

**Model:** We train a simple multi-layer perceptron (MLP) model with two fully connected layers using the FedAvg algorithm [74] over a total of 15 rounds. This model is lightweight, thus it can be realistically trained on resource-constrained and legacy UEs. The model size is $S = 100Ko$, and each communication round lasts $T = 300s$. The training time for each simulated UE is inferred from its training time on our setup.

## 5.6.2 Evaluation and Discussion

*1) Data-quality evaluation* In this part of the evaluation, we focus on the data-quality aspect independently from the wireless environment. We weight the value of dataset diversity indicator $I$ and the reputation $R$ under the two label flipping attacks. In each round, we select 5 UEs with the highest values for $V_k$.

Since MNIST is essentially an image classification task, and we have generated highly unbalanced datasets where several UEs only have a subset of the digits, we used Gini-Simpson index to evaluate the datasets' elements diversity [151]. We set the weights of the diversity index equally to $\gamma_i = 1/3$ and $\eta = 1$. We evaluate the different parts of the data-quality aspects by setting different values of $\omega_1$ and $\omega_2$.

Fig 5.2 shows the results for both label flipping attacks. While in both simulations, considering both aspects by setting $\omega_1 = \omega_2$ yielded better accuracy, we noticed different response to each aspect. For the tuple (6,2), Fig 5.2.a shows that following a selection strategy based on data-diversity can be a good strategy, while for the harder task (8,4), as shown in Fig 5.2.b, it seldom fails to converge.

*2) DQS evaluation:* In this part of the evaluation, we evaluate DQS under the two label flipping attacks. We model the cellular network as a square of side 500 meters with one BS located in the center of the square. The $K = 50$ UEs are randomly deployed inside the square following uniform distribution. The OFDMA bandwidth is $B = 1$ MHz. We set $P_k = -23dBm$ for all UEs. The channel gains $g_k$ between UE $k$ and the BS includes large-scale pathloss and small-scale fading following Rayleigh distribution, i.e., $|g_k|^2 = d_k^{-\alpha}|h_k|^2$ where $h_k$ is a Rayleigh random variable and $\alpha$ is the pathloss exponent and $d_k$ is the distance between UE $k$ and the BS.

While the results using DQS in Fig.5.3.a are in concordance with the results in Fig 5.2.a, Fig.5.3.b revealed different results. It is likely due to the varying number of participating UEs in each iteration that the results are different. Interestingly, through the results in both Fig.5.3.a and Fig.5.3.b, we noticed the long-term importance of the reputation aspect, on the contrast to the importance of dataset diversity in the early training rounds. This observation becomes clear with the pair (8,4), as it makes the learning harder. The role of reputation is far clearer at later rounds as the model becomes more sensitive to changes. As a result, an adaptive change of the weights $\omega_1$ and $\omega_2$ should be considered when using DQS.

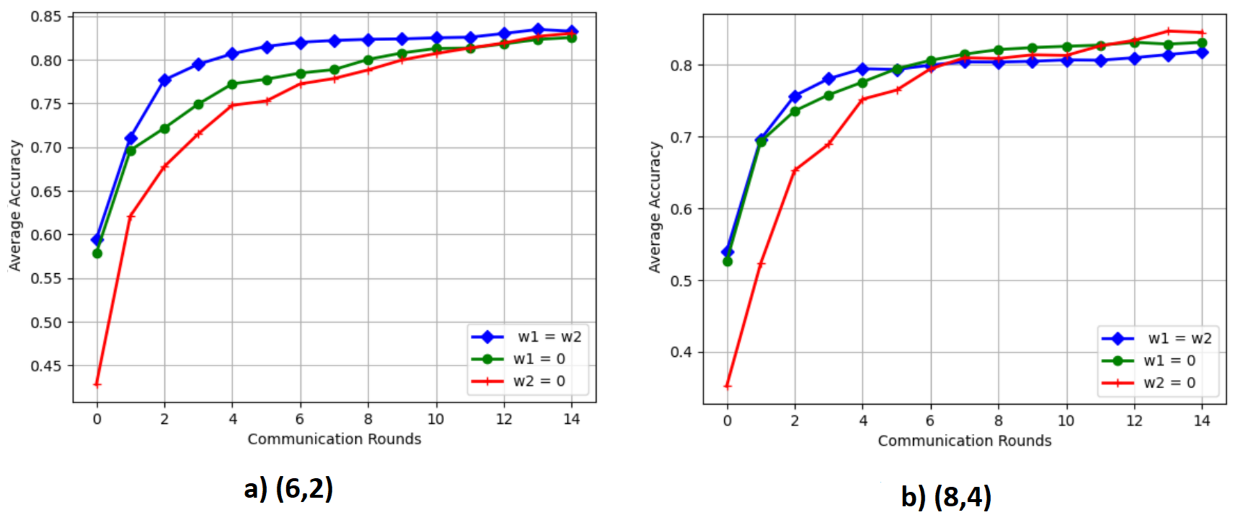


Figure 5.3 Model accuracy depending on the targeted label using DQS

## 5.7 Limitations and Future Work

Through our work on this paper, we have identified several future directions and open issues that need further investigation:

- **Use-case specific measures:** While our proposed approach is general, it remains necessary to choose the adequate data-quality measures for optimal results in each use-case.
- **Outliers:** Outliers in the context of FEEL might be wrongfully considered as malicious. A tractable solution can be combining the proposed approach with clustering techniques.
- **Other poisoning attacks:** Our proposed algorithm can be extended to handle other attacks such as model poisoning and multi-task poisoning attacks.

## 5.8 Conclusion

FEEL is the future of distributed training in wireless edge networks by virtue of its privacy preserving aspect. In this paper, we investigated scheduling of participant UEs in the collaborative training based on data-quality aspects. To this end, we proposed a data-quality based scheduling (DQS) algorithm for FEEL. DQS prioritizes devices with rich and diverse datasets, and punishes devices with poisoned datasets. We first defined the different components of the learning algorithm and the data-quality evaluation, namely dataset diversity and UE reputation. Then, we formulated a joint UE selection and bandwidth allocation problem, which we proved to be NP-hard. We presented our DQS algorithm for FEEL, which solves the problem in a greedy fashion. Finally, we evaluated the algorithm in different data poisoning scenarios, which showed the importance of data-quality evaluation components in scheduling UEs. In the future, we will enhance our proposed algorithm to handle other attacks such as model poisoning and multi-task poisoning attacks. Furthermore, we will run evaluations on larger scale by testing on other datasets and larger number of UEs.

# Chapitre 6: Avant-propos

**Auteurs et affiliation:**

Afaf Taïk: étudiante au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Zoubeir Mlika: Stagiaire Postdoctoral, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Soumaya Cherkaoui: Professeure, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

**Date d'acceptation:** Février 2022

**État de l'acceptation:** Version finale publiée.

**Revue:** IEEE Transactions on Intelligent Transportation Systems (IEEE T-ITS).

**Titre français:** Apprentissage fédéré par groupe dans les réseaux véhiculaires : Processus et optimisation

**Résumé français:**

L'apprentissage distribué implique l'apprentissage collaboratif d'un seul modèle ML parmi les dispositifs périphériques sur leurs ensembles de données distribués tout en conservant les données localement. Bien que l'apprentissage automatique nécessite moins de communication que l'apprentissage distribué classique, il reste difficile à mettre à l'échelle pour les grands modèles. Dans les réseaux de véhicules, l'apprentissage distribué doit être adapté aux ressources de communication limitées, à la mobilité des nœuds de bord et à l'hétérogénéité

statistique des distributions de données. En effet, une utilisation judicieuse des ressources de communication et de nouvelles méthodes d'apprentissage perceptif sont essentielles. À cette fin, nous proposons une nouvelle architecture pour les FL véhiculaires et les processus d'apprentissage et d'ordonnancement correspondants. L'architecture utilise les ressources de véhicule à véhicule (V2V) pour contourner le goulot d'étranglement de la communication où des grappes de véhicules forment des modèles simultanément et où seul l'agrégat de chaque grappe est envoyé au serveur du bord à accès multiple (MEC). La formation des clusters est adaptée à l'apprentissage mono et multi-tâches, et prend en compte les aspects de communication et d'apprentissage. Nous montrons par des simulations que le processus proposé est capable d'améliorer la précision de l'apprentissage dans plusieurs distributions d'ensembles de données non indépendantes et distribuées de manière identique (non-i.i.d) et non équilibrées, sous des contraintes de mobilité, en comparaison avec le FL standard.

# Chapitre 6: Foreword

**Authors and affiliation:**

Afaf Taïk: Ph.D. Student, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Zoubeir Mlika: PostDoctoral Researcher, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Soumaya Cherkaoui: Professor, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

**Date of acceptance:** February 2020.

**Acceptance status:** final version published.

**Journal:** IEEE Transactions on Intelligent Transportation Systems (IEEE T-ITS).

**Title:** Clustered Vehicular Federated Learning: Process and Optimization [1]

1. https://github.com/afaf-taik/vehicularFL

# CHAPTER 6

# Clustered Vehicular Federated Learning: Process and Optimization

## 6.1 Abstract

FL involves the collaborative training of a single ML model among edge devices on their distributed datasets while keeping data locally. While FL requires less communication compared to classical distributed learning, it remains hard to scale for large models. In vehicular networks, FL must be adapted to the limited communication resources, the mobility of the edge nodes, and the statistical heterogeneity of data distributions. Indeed, a judicious utilization of the communication resources alongside new perceptive learning-oriented methods are vital. To this end, we propose a new architecture for vehicular FL and corresponding learning and scheduling processes. The architecture utilizes vehicular-to-vehicular(V2V) resources to bypass the communication bottleneck where clusters of vehicles train models simultaneously and only the aggregate of each cluster is sent to the multi-access edge (MEC) server. The cluster formation is adapted for single and multi-task learning, and takes into account both communication and learning aspects. We show through simulations that the proposed process is capable of improving the learning accuracy in several non-independent and-identically-distributed (non-i.i.d) and unbalanced datasets distributions, under mobility constraints, in comparison to standard FL.

## 6.2 Introduction

Autonomous driving (AD) requires little-to-no human interactions to build an intelligent transportation system (ITS). Consequently, AD helps in reducing accidents caused by human driving errors. Artificial intelligence (AI) plays an essential role in AD by empowering several applications such as object detection and tracking through machine learning (ML) techniques [164, 10].

With the raise of AI research and deployment over the last decade, the development of autonomous vehicles has seen significant advancements. Indeed, vehicle manufacturers put a lot of effort to deploy AI schemes aiming to achieve human-level situational awareness. However, owing to technical difficulties and several ethical and legal challenges, it is still challenging for vehicles to achieve full autonomy. In fact, autonomous vehicles need to fulfill strict requirements of reliability and efficiency, and achieve high levels of situational awareness. Vehicle manufacturers are deploying efforts to achieve these goals. Autonomous vehicles will be capable of sensing their network environment using embedded sensors and share information with other vehicles and equipment through wireless communication. Autonomous vehicles can be equipped with LiDAR sensors, camera sensors, and radar sensors that collect important amounts of data to share with the vehicular network.

With the prevalence of connected vehicles and the transition toward autonomy, it is expected that vehicles will no longer rely only on locally collected data for localization and operation. Instead, enhanced situational awareness can be attained through exchanging raw and processed sensor data among large networks of interconnected vehicles [54]. In contrast to status data sharing, sensor data sharing becomes a pivotal operation for different safety applications, such as HD map building [165] and extended perception [166]. These data are also necessary to produce or enhance ML models that will be capable of performing AD tasks, such as dynamically adjusting the vehicle's speed, braking, and steering, by observing their surrounding environment.

Nonetheless, extensive sensor data sharing raises alarming privacy issues since vehicle sensor sharing involves sharing raw and processed data among vehicles. These data expose sensitive information about the vehicle, the driver, and the passengers, and could be used in a harmful way by a malicious entity. While privacy in vehicle status sharing has been already been extensively addressed and regulated by vehicle manufacturers—through a dynamic change of media access control (MAC) address and data anonymization, these regulations have not been extended to sensor data sharing. Moreover, to attain fully AD and enhance the overall ML models' performance, the deployed ML/AI models in the vehicle need to be updated and improved periodically by original equipment manufacturers (OEM). This requires the vehicles to upload the collected data to the OEMs, which further violates data privacy. Indeed, when data is uploaded to multi-access edge computing (MEC) servers, or to the cloud, it may be subject to be malicious interception and misuse.

Federated learning (FL)[143] has emerged as an attractive solution for privacy-preserving ML. FL consists of the collaborative training of ML models among edge devices without data-sharing, which makes it a promising solution for the continuous improvement of ML models in AD. Indeed, with FL, edge devices share their models parameters instead of their private data and then the models are aggregated at MEC servers to obtain a global accurate model.

When FL is used in a vehicular network context, a centralized entity (e.g., a MEC server) initializes a model and distributes it among participant vehicles. Each vehicle then trains the model using local data and sends the resulting model parameters to the central entity for aggregation.

The predominant FL training scheme is a synchronous aggregation. Accordingly, the MEC server waits for all vehicles to send their updates before aggregating them.

The assumption of FL is that the goal for participating end devices (also called end users throughout the article) is to approximate the same global function. Nevertheless, this is not the case for non-i.i.d data, particularly in the case of competing objectives, where a single joint model cannot be optimal for all end devices simultaneously. Consequently, clustering [78, 77] was proposed to group users with similar objectives and build multiple versions of the trained model. However, these works suppose the availability of all the end users and require their participation in the training for cluster-formation. Therefore, even if vehicle clustering for FL is interesting for the above mentioned reasons, due to the high-speed mobility, Doppler effect, and frequent handover (short inter-connection times), not all vehicle updates can be collected at the MEC servers. Further, due to the different mobility patterns, not all vehicles can have strong signal quality with the MEC servers. As a result, participating vehicles should be carefully selected and communication must be efficiently scheduled.

Vehicle-to-vehicle (V2V) communication offers a new opportunity for FL deployment that bypasses the communication bottleneck with the MEC server. A cluster of vehicles can collaboratively train models and a chosen cluster-head can aggregate their updates so as only one model is sent to the MEC server. To achieve this, two main questions need to be addressed: how to adequately form FL clusters under mobility constraints; and how to select the cluster-heads in such settings.

In this article, we propose a cluster-based scheme for FL in vehicular networks. The clustering scheme consists of grouping vehicles with common characteristics, not only in terms of direction and velocity, but also from a learning perspective through the evaluation of the updates' similarity. Thus, the proposed scheme allows to accelerate the models' training through ensuring (i) a larger number of participants (ii) possibility to train several models to adapt to non-i.i.d and unbalanced data distributions.

The main contributions of this article can be summarized as follows:

1) we design an architecture and corresponding FL process for clustered FL in vehicular environments;
2) we formulate a joint cluster-head selection and resource block allocation problem taking into account mobility and data properties;
3) we formulate a matching problem for cluster formation taking into account mobility and model preferences;
4) we prove that the cluster-head problem is NP-hard and we propose a greedy algorithm to solve it;
6) we evaluate the proposed scheme through extensive simulations.

The remainder of this article is organized as follows: In Section II, we present the background for FL and related work. In Section III, we present the design of the learning process and considered system model components. In Section IV, we formulate the cluster-head selection and vehicle association problems, and we present the proposed solution. Simulation results are presented in Section V. At last, conclusions and future work are presented in Section VI.

## 6.3 Background

In this section, we first present a background on FL and challenges tackled in this paper, then we present related work that enables and motivates our work.

### 6.3.1 Federated Learning

FL is a privacy-preserving distributed training framework, which consists of the collaborative training of a single ML model among different participants (e.g.,IoT devices) on their local datasets. The training is an iterative process that starts with the global model initialization by a centralized entity (e.g., a server). In every communication round $i$ , a selected subset of $N$ participants receive the latest global model $\theta_t$. Then, every participant $k$ trains the model by performing multiple iterations of stochastic gradient descent (SGD) on minibatches from its local dataset $D_k$. The local training results in a several

Table 6.1 List of Notations.

| **Notations** | **Description** |
|---|---|
| $T_k$ | Rate of stay of vehicle $k$ |
| $Total_{RB}$ | Total available resource blocks |
| $s$ | Model size |
| $\epsilon$ | Number or local epochs |
| $g$ | Global model |
| $\theta_k$ | Model update of vehicle $k$ |
| $t_k^{train}$ | Training time for vehicle $k$ |
| $r_k$ | Achievable data rate of vehicle $k$ |
| $t_k^{up}$ | Upload time for vehicle $k$ |
| $P_k$ | Transmit power of vehicle $k$ |
| $N_0$ | Power spectral density of the Gaussian noise |
| $\|D_k\|$ | vehicle $k$'s dataset size |
| $I_k$ | Data-diversity of vehicle $k$ |
| $R_{k,h}$ | Relationship of vehicles $k$ and $h$ |

weight-update vectors $\Delta\theta_k^{t+1}$, which are sent to the server. The last step is the model aggregation at the server, which is typically achieved using weighted aggregation [143] following Eq.6.1. The process is then repeated until the model converges.

$$\theta_{t+1} = \theta_t + \sum_{k=1}^{N} \frac{|D_k|}{|D|} \Delta\theta_k^{t+1} \tag{6.1}$$

While this aggregation method takes into account the unbalanced aspect of datasets' size, it is not always suitable for non-i.i.d distributions. Furthermore, FL in wireless networks in general, and in vehicular networks in particular, is subject to the following challenges:

**Statistical heterogeneity:** One of the underlying challenges for training a single joint model in FL settings is the presence of non-i.i.d data. For instance, some nodes only have access to data from a subset of all possible labels for a given task, while other nodes may have access to different input features. Furthermore, varying preferences for instance can lead to concept shift (i.e., nodes classify same features under different labels, or vice-versa). In practice, these non-i.i.d settings are highly likely to be present in a given massively distributed dataset. Thus, training models under these settings requires new sets of considerations.

**Partial Participation:** Given the scarcity of the communication resources, the number of participating nodes is limited. In fact, the generated traffic grows linearly with the number of participating nodes and the model size. Moreover, the heterogeneity of the nodes in terms of computational capabilities and mobility (i.e., velocity and direction) introduces stringent constraints on the communication. Hence, enabling FL on the road in a communication-efficient way is far from an easy task.

### 6.3.2 Related Work

Several works consider FL as a key enabler for vehicular networks in general, and AD in particular [167], such as secure data sharing [54], Autonomous Controllers [168], caching [169], and travel mode identification from non-i.i.d GPS trajectories [170]. Nonetheless, deploying FL on the road remains a challenging task due to uncertainties related to mobility and communication overhead. To overcome the communication bottleneck, works [171, 172, 173] have proposed judicious node selection and resource allocation for efficient training. However, these schemes are specifically designed for the topology and dynamics of standard wireless/cellular networks with high node density but relatively low mobility. In contrast, vehicular networks have rather low node density and very high node mobility [63]. As a result, new schemes are required for FL on the road. Meanwhile, V2V communication offers a new possibility for FL deployment that bypasses the bottleneck of communication with the MEC server. In vehicular networks, some vehicles serve as edge nodes to which neighboring nodes offload computation and data analysis tasks [174]. Edge vehicles are also used to provide a gateway functionality by ensuring continuous availability of diversified services such as multimedia content sharing [175]. A common practice among such works is creating clusters of vehicles where the edge vehicle acts as a cluster head. The clusters are formed based on several metrics such as the distance between the vehicles, their velocity and direction. Yet, these clustering schemes cannot be directly exploited in the context of FL. Recent VANET clustering works principally design algorithms based on their primary application [176, 177]. This is a logical approach since the design of a clustering algorithm highly influences the performance of the application for which it is used. A popular approach for cluster head selection widely used in the literature [177, 178, 179] requires each vehicle to calculate an index quantifying its fitness to act as a cluster head for its neighbours. Vehicles wishing to affiliate with a cluster head rank all neighbours in their neighbour table and request association with the most highly-ranked candidate node. The index is calculated as a weighted sum of several metrics, such as the degree of connectivity and link stability, with weights chosen depending on the importance of the considered metrics. However, due to the nature of FL applications, metrics related to learning/data should also be considered.

Furthermore, clustering is already used in FL as a means to accelerate the training by grouping nodes with similar optimization goals, which train different versions of the model instead of one global model [78, 180, 77, 181]. In fact, one of the fundamental challenges in FL is the presence of non-i.i.d and unbalanced data distributions [182, 144]. These challenges go against the premise of FL which aims to train one global model. Such settings require new mechanisms to be put in place in order to ensure models' convergence. Clustered FL has attracted several research efforts, as it has generalization [183] and convergence [181] guarantees under non-i.i.d settings. By creating different models to adapt to different end users' distributions, clustered FL allows better model performance in the case of concept-shift. Concept-shift [77] occurs when different inputs do not have the same label across users as preferences vary. Moreover, in clustered FL, training becomes resilient to poisoning attacks [184] such as label flipping [185] (i.e., nodes misclassify some inputs under erroneous labels).

For instance, authors in [78], develop a clustered FL procedure. Their work allows to find an optimal bipartitioning of the users based on cosine similarity for the purpose of producing personalized models for each cluster. The bipartitioning is repeated whenever FL has converged to a stationary point. In [77], a single clustering step, in a predetermined communication round, is introduced. In this step, all the users are required to participate and the similarity of the updates is used to form clusters using hierarchical clustering. Nonetheless, the proposed approach requires knowing a distance threshold on the similarity values between the updates to form the clusters. Furthermore, cluster-based approaches assume that all the users participate, which is unfeasible under dynamic and uncertain vehicular networks.

To the best of our knowledge, our work is the first to address the problem of clustered FL in hierarchical mobile architectures, while considering the users' data distributions, wireless communication characteristics, and resource allocation constraints. Specifically, unlike other studies, we consider the learning aspect (i.e., nodes dataset characteristics and model dissimilarities), in addition to communication constraints (i.e., wireless channel quality, mobility, and communication latency). Henceforth, we propose a practical way to deploy FL in vehicular environments.

## 6.4 System Model

We consider a vehicular network composed of a set $V$ of $K$ vehicles and a set $U$ of $N$ gNodeBs. Both the communication among vehicles and with the gNodeBs are through wireless links. Additionally, gNodeBs are connected to the Internet via a reliable backhaul link. The vehicles have enough computing and storage resources for the training, and the gNodeBs are equipped with MEC servers. MEC servers are used to schedule the vehicles nearby, aggre-

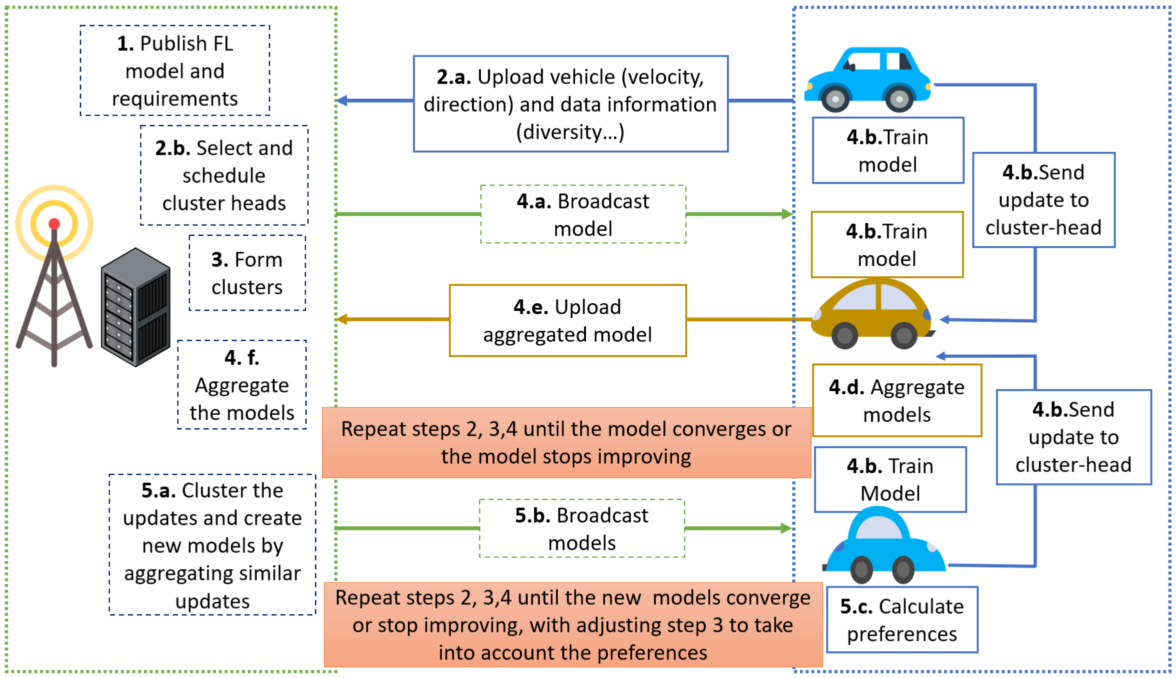

1. Publish FL model and requirements
2.b. Select and schedule cluster heads
3. Form clusters
4. f. Aggregate the models
5.a. Cluster the updates and create new models by aggregating similar updates
2.a. Upload vehicle (velocity, direction) and data information (diversity…)
4.a. Broadcast model
4.e. Upload aggregated model
Repeat steps 2, 3,4 until the model converges or the model stops improving
5.b. Broadcast models
Repeat steps 2, 3,4 until the new models converge or stop improving, with adjusting step 3 to take into account the preferences
4.b.Train model
4.b.Send update to cluster-head
4.b.Train model
4.d. Aggregate models
4.b.Send update to cluster-head
4.b. Train Model
5.c. Calculate preferences

Figure 6.1 Illustration of the different steps in clustered vehicular federated learning

gate the updates and manage the clusters. In the following, we explain the proposed cluster-based training process and the different components of the considered system model (i.e., communication and computation) in a vehicular environment.

## 6.4.1 Process Overview

FL in vehicular networks is subject to several challenges related to data, mobility, and communication and computation resources. In this paper, we consider these aspects in the design and optimization of the FL process in vehicular networks.

The first set of challenges are related to data, where the learning process should be adapted to take into account data heterogeneity in order to accelerate the model convergence. Data generated across different applications in vehicular networks depend on the specific vehicle sensors and these sensors' data acquisition activities which often leads to heterogeneous data distributions among FL participants (i.e., different dataset sizes and different data distributions). Furthermore, the dependence on data acquisition activities from vehicles with similar sensing capabilities makes the collected data highly redundant. As a result, local datasets cannot be regarded the same in terms of information richness, as some datasets may have more diverse and larger datasets than other participants. Furthermore, communication resources in this context are limited. In fact, in addition to the bandwidth's scarcity, the possible time for communication with the MEC server is limited by the time where a vehicle is in the area covered by the base station. For all these reasons, the participant selection and the bandwidth allocation mechanisms should be carefully designed for FL in vehicular networks. Hence, in this article, we use the data properties to guide the participants' selection in the training and communication process.

Furthermore, the model convergence speed is highly dependent on the number of collected updates. Vehicle-to-vehicle (V2V) communication offers a great alternative to bypass the communication bottleneck in vehicular networks by allowing some select vehicles to act as mediators between other vehicles and the MEC server. We propose to use V2V in order to maximize the collected updates under the communication uncertainty.

In these perspectives, we propose to prioritize the vehicles with the most informative datasets and use them as cluster heads, while the remainder of the vehicles are associated with them. In this setting, each cluster-head aggregates the models of the vehicles in its cluster and uploads the resulting model. In fact, instead of sending all the collected updates, the cluster-head will aggregate the updates and send one aggregated model which is more communication-efficient. In this case, hierarchical FL is used as a means to optimize the communication in vehicular networks, where the MEC server will do a second round of aggregation.

Another aspect that needs to be considered is mobility and how it affects the communication among vehicles and with the MEC server. In order for the cluster-heads to successfully upload their models to the MEC server, the upload should be completed before the vehicles leave the coverage area of the BS. Furthermore, for the vehicles to be able to send their models to the cluster-head, their link lifetime (LLT) should be longer than the required time for training and uploading the models.

In order to adapt this approach to the case where multiple models need to be trained, other considerations need to be taken into account in this approach. In fact, in the case where data distributions are subject to concept-shift, a single model is not enough. Concept-shift is another kind of data heterogeneity that arises in cases where data is subjective and depends on the preferences of end users, or in the presence of adversaries. In classification problems for instance, concept shift is when similar inputs have different labels depending on the end user. In the case of vehicles, the latter could simply not share the same model if they are not from the same OEM. The presence of different perspectives from different vehicles makes one model hard to fit all. In our paper, we use hierarchical clustering through evaluating the model updates and their cosine similarity. The clustering can be executed on a predetermined communication round or when the model's convergence slows down. The newly created models will be used to associate each vehicle to the most adequate cluster-head. The same model can be trained among several clusters as such redundancy is worthwhile when it comes to system robustness in the case of user dropout, and it also helps the model's convergence through collecting more updates.

All in all, to address the challenges linked to mobility and data heterogeneity, we design a mobility-aware scheme for clustered FL, that takes into account the data and model heterogeneity. The data heterogeneity is mainly considered in the selection of cluster-heads, while the model heterogeneity is used to create new models and in matching vehicles to cluster-heads. In the following subsections, we start with detailing the overall learning model, then we present the mathematical formulation of its different aspects. We detail the steps of the clustered vehicular FL training procedure, then we give the formulations of the different metrics used in the procedure.

### 6.4.2 Learning Model

A summary of the process is given in Algorithm 5, and more details of the scheme are given as follows:

- **Step 1** (Publish FL model and requirements, and receive feedback ) : A global model is published by the MEC server, alongside its data and computation resource requirements (e.g., data types, data sizes, and CPU cycles). Each vehicle $k$ satisfying the requirements sends positive feedback, in addition to other information such as its data diversity index $I_k$ (see Eq. 6.2) and current velocity $v_k$.
- **Step 2** (Select and schedule cluster-heads $H$): The MEC server chooses the cluster-heads according to the received information. The selection is based on the dataset characteristics (i.e., quality of the dataset and the quantity of the samples), defined in subsection 6.4.2, in addition to the state of the wireless channel and the projected duration of the communication reflected by the rate of stay (See Eq. 6.5). In fact, the quality of local dataset directly determines the quality and the importance of model updates, while the velocity and the state of the wireless channels determine whether the model update can be received during the communication round. The details about the data evaluation are given in subsection 6.4.2 , and the algorithm (Algorithm 4) is explained in Section 6.5.1.
- **Step 3** (Clusters formation): After cluster-head selection, the set of the remaining vehicles $NH$ are matched to cluster-heads (set $H$). The

matching requires that the sum of training and upload time of vehicle $k$ is less than the Link Lifetime (LLT) (defined in Eq. 6.8) between $k$ and $h \in H$ if they are to be matched. Furthermore, the matching aims to maximize the weighted sum of $R_{k,h}$. $R_{k,h}$ symbolizes the relationship between $k$ and $h$, and its definition changes depending on whether there is only one global model or several versions (See Eq.6.4). In the simple case of a single joint model, the clustering depends only on the mobility and accordingly for all the pairs $k \in NH, h \in H$ the value of $R_{k,h} = 1$. Otherwise, each vehicle should train its preferred model. The preference is defined as the accuracy of the model trained by $h$ on the local data of $k$. This definition is due to the fact that not all vehicles can participate in the updates clustering step (See Step 5).

- **Step 4** (Model broadcast and training) : The model is broadcasted to the participants, where each vehicle trains on its local data for $\epsilon$ local epochs, before sending the update to the corresponding cluster-head. Each cluster-head then aggregates the received models and sends the update to the MEC server, which in its turn aggregates the global updates of the clusters. Such hierarchical FL aggregation is widely adopted in the literature of FL [186, 187] and allows for more participation. Aggregating the updates at the MEC server level is required because each model version can be trained within several clusters, resulting in several global models. Such redundancy is necessary in the case of vehicular networks, as it allows more robustness to client drop-out.

- **Step 5** (Updates Clustering and Preference Evaluation): If the global model does not converge after several communication rounds, or the goal accuracy is not attained, we perform a communication round (or several communication rounds) involving a large fraction of the vehicles on the global joint model. This step requires the collection of the updates at the MEC server without prior aggregation by cluster-heads as the aggregated models would mask the divergence of the different models. The updates are used to judge the similarity (defined in Eq.6.3) between participants using the hierarchical clustering algorithm. It is employed to iteratively merge the most similar clusters of participants up to a maximum number of clusters defined by the OEM. Fixing the maximum number of clusters allows to create clusters without prior knowledge of the possible distances between updates, while controlling the number of models in circulation. Once the clusters are created, new models are generated through aggregation. The models are broadcasted to the available vehicles. Each vehicle evaluates the models on its local data and send them back to the MEC server. These values are later used to evaluate $R_{k,h}$ for each vehicle $k$. The resulting models are then trained independently but simultaneously using the same process. This preferences' evaluation makes the difference between our work and previous work in clustered FL, as these works necessitate the participation of all the nodes, while in our work we tolerate partial participation.

The iterations and the steps' order are illustrated in Fig.6.1. Next, we present the formulations of the different elements in the system model, starting with the learning aspects (i.e., dataset charasteristics and models similarity), to the different mobility and communication aspects considered throughout the proposed approach.

#### Dataset characteristics

Considering the fact that datasets are non-i.i.d and unbalanced, a judicious cluster-head selection (Step 2) is necessary. In fact, each dataset can be characterized by how diverse its elements are, its size and how many times the model was trained on it (i.e., age of update). In this paper, we focus on the non-i.i.d and unbalanced aspect, however, other metrics can be considered depending on the learned task, including the quality of the datasets and their reliability. We set the value of each metric as [151]: $\varphi_j \gamma_j$, where $\gamma_j$ is the adjustable weight for each metric assigned by the server and $\varphi_j$ is the normalized value of the metric $j$. Using the aforementioned characteristics, the diversity index of dataset at node $k$ can be defined as:

$$I_k = \sum_j \varphi_{j,k} \gamma_j, \tag{6.2}$$

with $j \in \{\text{elements diversity}, \text{dataset size}, \text{age}\}$. The metric can be easily adjusted to include other task-specific considerations.

### Updates similarity

In order to handle the non-i.i.d aspect, the updates' similarity is evaluated using cosine similarity [78, 77] in Step 5 of the algorithm, and new models are created by aggregating the most similar models. Given two model updates $\Delta\theta_k$ and $\Delta\theta_l$, the similarity is calculated according to:

$$sim(k, l) = \frac{\langle \Delta\theta_k, \Delta\theta_l \rangle}{\|\theta_k\| \, \|\theta_l\|} \tag{6.3}$$

where $\langle ., . \rangle$ is the dot product of two vectors. The dot product is divided by the product of the two vectors' lengths (or magnitudes). The values of $sim(.)$ are between 0 and 1, and the dissimilarity (i.e., cosine distance metric) $1 - sim(.)$ is used to cluster the updates. The cosine distance metric is invariant to scaling effects and therefore indicates how closely two vectors (and in our case updates) point in the same direction. The models' similarity is then used to created clusters using the hierarchical clustering algorithm [77], and the most similar models are aggregated to create new models.

### Vehicles Relationships

During the cluster formation in Step 3, each cluster is created based on the relationship between the vehicles. The definition of this relationship depends on whether only one global model is trained, or there are several versions of the model that are created. In the case of multiple models, we define the preference of a model through its accuracy on the $k$th vehicle's dataset. We define the relationship between two vehicles $R_{k,h}$ as follows:

$$R_{k,h} = \begin{cases} \text{accuracy of } h & \text{if more than 1 model} \\ 1 & \text{otherwise} \end{cases} \tag{6.4}$$

### 6.4.3 Communication Model

In Step 2, due to mobility and communication constraints, the RB allocation is jointly executed with the cluster-head selection. In fact, the mobility imposes a deadline for the upload based on the standing time of the vehicle. Additionally, in Step 3, the cluster formation must also consider the relationship between the vehicles in terms of mobility, which is modelled through the link lifetime (LLT). The different aspects of the communication model are formulated as follows:

#### Standing time

While typically in FL, the duration of a communication round is fixed by the centralized entity (e.g., MEC server), the latency in FL in vehicular networks is dictated by the standing time of participating nodes. Let the diameter of coverage area of a gNodeB be denoted as $D$. For each vehicle $k$, the standing time in the coverage area of current gNodeB is defined by Eq. 6.5 [169]:

$$T_k = \frac{D - x_k}{v_k} \tag{6.5}$$

To ensure the communication with the gNodeB, the rate of standing time of a vehicle $k$ selected as cluster-head should respect $(t_k^{train} + t_k^{up} + T_{agg} + \delta) \leq T_k$. Where $t_k^{train}$ and $t_k^{up}$ are the estimated training time and upload time of vehicled $k$ respectively, $T_{agg}$ is the time required for aggregation and $\delta$ is a waiting time for the updates' collection. We can notice that what varies the most among the vehicles are $t_k^{train}$ and $t_k^{up}$, as $t_k^{train}$ depends on the size of the dataset, and $t_k^{up}$ depends on the channel gain and the resource block allocation.

### Resource Blocks

For each vehicle $k$, we can infer the maximum $t_k^{up}$ by setting $(t_k^{train} + t_k^{up} + T_{agg} + \delta) = T_k$. As a result, we can determine the minimum required data rate $r_{k,min}$ to send an update of size $s$ within a transmission time of $t_k^{up}$ as follows:

$$t_k^{up} = \frac{s}{r_{k,min}}. \tag{6.6}$$

The achievable data rate of a node $k$ over the RB $q$ is defined as follows:

$$r_k^q = B \log_2(1 + \frac{P_k G_{k,q}}{N_0}) \tag{6.7}$$

where $B$ is the bandwidth of a RB, $P_k$ is the transmit power of node $k$, and $N_0$ is the power spectral density of the Gaussian noise. The data rate of a vehicle is the sum of the datarates on all the RBs assigned to it.

### Link Lifetime

In Step 3, in order to associate a vehicle $k \in NH$ to a cluster-head $h \in H$, it is necessary to evaluate the sustainability of the communication link, so as to ensure that the update of the node $k$ will be successfully sent to $h$. Link Lifetime (LLT) [188] defines the link sustainability as the duration of time where two vehicles remain connected. LLT is defined in [188, 189, 190] by Eq. 6.8, for two vehicles $k$ and $h$ moving in the same or opposite directions. Assuming that the trajectory of all vehicular nodes to be a straight line, as the lane width is small, the y-coordinate can be ignored. We denote the positions of $k$ and $h$ by $x_k$ and $x_h$ , respectively.

$$LLT_{k,h} = \frac{-\Delta v_{kh} \times D_{kh} + |\Delta v_{kh}| \times TR}{(\Delta v_{kh})^2} \tag{6.8}$$

with $\Delta v_{kh} = v_k - v_h$ and $D_{kh} = x_k - x_h$ and $TR$ denotes the transmission range. Accordingly, the training time of $k$ and upload time from $k$ to $h$ must be less or equal to $LLT_{kh}$ (i.e., $(t_k^{train} + t_k^{up,h}) \leq LLT_{kh}$.

## 6.5 Problem formulation & Proposed Solution

### 6.5.1 Problem Formulation

Considering the collaborative aspect of FL and the communication bottleneck, we define the following goals for the cluster-head selection and cluster association:

– From the perspective of accelerating learning and maximizing the representation, the scheduled cluster-heads must have diverse and large datasets, as a result the goal of cluster-head selection is:

$$\max_{h,\alpha} \sum_{k=1}^{K} h_k I_k. \tag{6.9}$$

– In order to guarantee that each vehicle trains its preferred model, the cluster assignment can be defined as a matching problem where we aim to maximize the relationship $R_{k,h}$.

$$\max_{m} \sum_{h \in H} \sum_{v \in NH} R_{v,h} m_{v,h} \tag{6.10}$$

Several constraints related to communication are imposed by the vehicular environment. Consequently, the first problem considered is a joint cluster-head selection and RB allocation. For each vehicle $k$ and RB $q$ we define $\alpha_{k,q}$ as:

$$\alpha_{k,q} = \begin{cases} 1 & \text{if } q \text{ is assigned to k} \\ 0 & \text{otherwise} \end{cases} \tag{6.11}$$

The cluster-head selection and RB allocation problem is formulated as follows:

$$\underset{h,\alpha}{\text{maximize}} \quad \sum_{k=1}^{K} h_k I_k \tag{6.12a}$$

subject to

$$(t_k^{train} + \delta + t_k^{up} + T_{agg})h_k \leq T_k, \qquad \forall k \in [1, K], \tag{6.12b}$$

$$\sum_{k=1}^{K} \alpha_k \leq Total_{RB}, \qquad \forall k \in [1, K], \tag{6.12c}$$

$$h_k \in \{0, 1\}, \qquad \forall k \in [1, K]. \tag{6.12d}$$

Taking into account the results from the previous problem, we define $H = \{k, h_k = 1\}$ (i.e., the cluster-heads) and $NH = \{k, h_k = 0\}$ (i.e., the remainder of the vehicles). The next step is matching the set of vehicles $NH$ to selected cluster-heads $H$. We consider that a maximum capacity $N_{max}$ is fixed for each cluster in order to reasonably allocate the V2V communication resources. Additionally, if a vehicle $v$ is to be matched with a cluster-head, it needs to respect the time constraints, where it should be able to finish training and uploading before a deadline $T_h = t_h^{train} + \delta$, and the $LLTv, h$ should at least outlast the training and upload. We define $m_{v,h}$ as a binary variable equal to 1 if $v$ is matched with $h$ and 0 otherwise. Accordingly, we define the second problem as follows:

$$\underset{m}{\text{maximize}} \quad \sum_{h\in H}\sum_{v\in NH} R_{v,h}m_{v,h} \tag{6.13a}$$

subject to

$$\sum_{h\in H} m_{v,h} \le 1, \qquad \forall v \in NS, \tag{6.13b}$$

$$\sum_{v\in NH} m_{v,h} \le N_{max}, \qquad \forall v \in NS, \tag{6.13c}$$

$$(t_v^{train} + t_v^{up})m_v, h \le LLT_{v,h}, \qquad \forall v \in NH, \tag{6.13d}$$

$$(t_v^{train} + t_v^{up})m_v, h \le T_h, \qquad \forall v \in NH, \tag{6.13e}$$

$$m_{v,h} \in \{0,1\}, \qquad \forall v \in NH. \tag{6.13f}$$

### 6.5.2 Proposed Algorithm

In this section, we present our proposed solution for cluster-head selection and RB allocation alongside the matching algorithm to solve 6.12a and 6.13a. The challenging aspect of the problem 6.12a is that it requires maximizing the weighted sum of the selected vehicles and jointly allocating the bandwidth. A restricted version of problem 6.12a can be shown to be equivalent to a knapsack problem and thus it is NP-hard [191]. In fact, the problem aims to select vehicles that maximize the weighted sum $\sum_k I_k h_k$ subject to a knapsack capacity given by $\sum_k \alpha_k \le Total_{RB}$ in constraint (12c), which can be transformed to $\sum_k \alpha_k h_k \le Total_{RB}$ where $\alpha_k$ represent the weight of item $k$ (fixed for this restricted version) and $Total_{RB}$ represents the knapsack capacity. Thus, the problem is equivalent to a knapsack problem and since the latter is NP-hard, so is problem 6.12a. Constraint (12b) can be verified for each vehicle to filter out the ones that cannot upload the updates in time.

We chose to follow a greedy knapsack algorithm to solve the problem. In fact, we chose the greedy approach because it will allow us to select the best candidates with an optimal RB cost, unlike the ranked list solution, which would have optimized the sum of $I_k$ only [162]. Furthermore, the greedy knapsack algorithm has low complexity and will allow fast and efficient scheduling under the rapidly changing vehicular environment. We calculate the minimum required RBs for each vehicle $k$ to be able to send the update by the deadline $T_k$, which we consider the cost of the scheduling $c_k = \sum_{q \in RBs} \alpha_{k,q}$. The main time consuming step is the sorting of all vehicles in a decreasing order based on their diversity value / cost in RBs ratio. After the vehicles are arranged as an ordered list, the following loop takes $O(n)$ time. Taking into account that the worst-case time complexity of sorting can is $O(n \log n)$, the total time complexity of the proposed greedy algorithm is $O(n \log n)$.

The second formulated problem 6.13a is a maximum weighted bipartite matching problem [192, 193], where each $h \in H$ has a maximum capacity $N_{max}$ and each $v \in NH$ has a capacity of 1.

In order to include the remainder of the constraints, we define $\zeta_{v,h}$ as a binary value, where $\zeta_{v,h} = 0$ if constraint (13d) cannot be satisfied if $m_{v,h} = 1$, and $\zeta_{v,h} = 1$ otherwise. The goal is redefined so as to maximize a weighted sum of $R_{v,h} \times \zeta_{v,h}$. The problem becomes an integer linear program (ILP) and solved using an off-the-shelf ILP solver (e.g., Python's PulP [142]).

To illustrate the problem, we consider the example in Fig.6.2. The vehicles and their relationships can be considered as a graph, where the vehicles represent the edges and their relationship is represented through the vertices, which are weighted with $R_{v,h} \times \zeta_{v,h}$. The goal is to find a subgraph where the selected vertices have an optimal (in our case maximum) sum. The remaining constraints are the maximum capacities of the vehicles (in red). The cluster-heads (in yellow, on the right) have a maximum capacity $N_{max} = 3$ each (Constraint 6.13c), and the other vehicles have capacity of 1 (Constraint 6.13b). In the illustrated problem, the pairs $v_2, h_2$ and $v_3, h_1$ cannot be matched since the edges ( in dashes lines ) have null values, which can be either due to possible disconnection or of poor model performance. The choice of the optimal matching is then left among the remaining pairs. The optimal solution for the illustrated problem in yellow lines has a sum of 3.0.

We define our Algorithm 6, Clustered Vehicular FL (CVFL) that iteratively selects nodes with best ratio $\frac{I_k}{c_k}$ to be cluster heads, and then matches the rest of the vehicles to them after verifying the time constraints by creating clusters that maximize $\sum_{h \in H} \sum_{v \in NH} R_{v,h} \times \zeta_{v,h}$.

## 6.6 Performance Evaluation

### 6.6.1 Simulation Environment and Parameters

The simulations were conducted on a desktop computer with a 2,6 GHz Intel i7 processor and 16 GB of memory and NVIDIA GeForce RTX 2070 Super graphic card. We used Pytorch [141] for the machine learning library. In the following numerical results, each presented value is the average of multiple independent runs.

**Datasets:** We used benchmark image classification datasets MNIST [84],a handwritten digit images, and Fashion-MNIST [85], grayscale fashion products dataset, which we distribute randomly among the simulated devices. MNIST and FashionMNIST constitute simple yet flexible tasks to test various clustered settings and data partitions. Each dataset contains 60,000 training examples and 10,000 test examples. The data partition is designed specifically to illustrate various ways in which data distributions might differ between vehicles. The data partition we adopted is as follows: We first sort the data by digit label, then we form 1200 shards composed of 50 images each. Each shard is composed of images from one class, i.e. images of the same digit. In the beginning of every simulation run, we randomly allocate a minimum of 1 shard and a maximum of 30 shards to each of the $K$ vehicles.

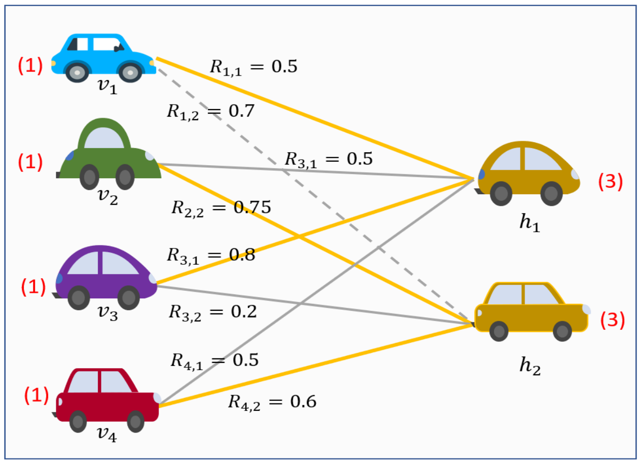


Figure 6.2 Illustration of the Matching problem

This method of allocation allows us to create an unbalanced and non-i.i.d distribution of the dataset, which is varied in each independent run.

Furthermore, in order to evaluate the updates' clustering and how adequate is the preferences' evaluation, we partition the vehicles' indexes into $N_{shifts}$ groups. For each group two digit labels are swapped. For instance, one group might swap all digits labelled as 1 to 7 and vice versa. The swapped tuples are: $\{(1,7),(3,5)\}$ for MNIST and $\{(1,3),(6,0)\}$ for FashionMNIST [194]. Each group is then evenly distributed to $\frac{K}{N_{shifts}}$. This partition allows us to test the proposed algorithm's ability to train models in the presence of concept shift and unbalanced data. The test set is divided into $N_{shifts}$ datasets and the average accuracy is then reported.

**FL Parameters:**

We consider $K = 30$ vehicles collaboratively training multi-layer perceptron (MLP) model with two hidden layers (64 neurons in each), and a convolutional neural network (CNN) model with two 5x5 convolution layers (the first with 10 channels, the second with 20, each followed with 2x2 max pooling), two fully connected layers with 50 units and ReLu activation, and a final softmax output layer. We use lightweight models as they can be realistically trained on end-devices in rapidly changing environments. For each participant, due to the mobility of the vehicles and in order to collect a maximum number of updates, it is more practical to choose a small number of local epochs, as a result, in the following simulations, the number of local epochs is set to $\epsilon = 1$. In the preliminary evaluations, the maximum number of communication rounds is $i_{max} = 30$. The clustering is set in round 25. $t_{k,train}$ for each vehicle is calculated locally using our configuration.

### 6.6.2 Preliminary evaluations: Parking Lot Scenario

In this part of the evaluations, we focus on the learning aspect by studying the proposed algorithm in less constrained environment.

#### Simple unbalanced and non-i.i.d distribution

In this part of the simulation, we ignore the constraint of LLT in problem (13) as the velocities are set to 0. The results in Fig.6.3 show that a significant improvement is reached through the use of V2V communication. With more participation, we also noticed that the training tends to be more stable with the loss function steadily declining in comparison to standard FL. Furthermore, higher accuracy scores are achieved by our proposed method.While the average local accuracy after the end of the training the MLP on MNIST is $80\% \pm 10\%$for vanilla FL, it reaches and average of $82\% \pm 9\%$ for our proposed approach. Similarly, on FashionMNIST the results $66.79\% \pm 10\%$ with vanilla FL and $68.74\% \pm 9\%$. Owing to its high suitability for image processing tasks, the CNN model yielded higher results as the vanilla FL reached $94.58\% \pm 7\%$ and our proposed method achieved $95.5\% \pm 5\%$. Such results can be considered as a baseline values in perfect conditions for the subsequent experiments as we can reflect on the robustness of CVFL under mobility and concept-shift. Based on these preliminary results, we expect to see more differences and variance in the results for the MLP model compared to the CNN model. We also can expect a better performance for the MLP model on the MNIST dataset compared to FashionMNIST.

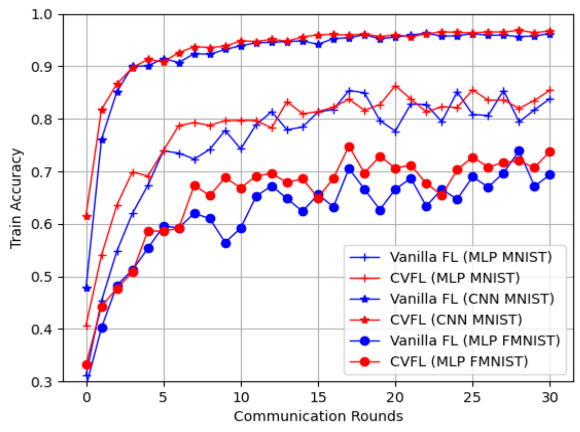


Figure 6.3 Preliminary results on non-i.i.d and unbalanced data without concept-shift

#### Unbalanced and non-i.i.d distribution with concept shift

The presence of concept-shift requires the clustering phase in order to improve the final results. In these simulations, we fixed the number of maximum clusters to 2, and studied the effect of partial participation on the clustering. Given the presence of concept shift for 4 out of 10 digits, we expect the accuracy to be around 60%.

To study the effect of the fraction of participants in the partial clustering phase , we run multiple independent runs for each fraction in $\{20\%, 60\%, 100\%\}$. The results are shown in Fig.6.4. For both vanilla FL and the proposed partial clustering approach, the number of participants in each round is 6. For the standard FL, the average accuracy is 65%, while For 20% the average 68% (+3%) and for 60% the average is 69% (+5%). It should be noted that the dissimilarity of the updates is harder to detect as only 2 out of 10 digits are swapped for each group.

### 6.6.3 Freeway scenario

We consider that the $K = 30$ vehicles are randomly distributed on 6 lanes on a radius $D = 2km$. The vehicular communication model parameters and mobility are based on parameters in [195] and are summarized in Ta-

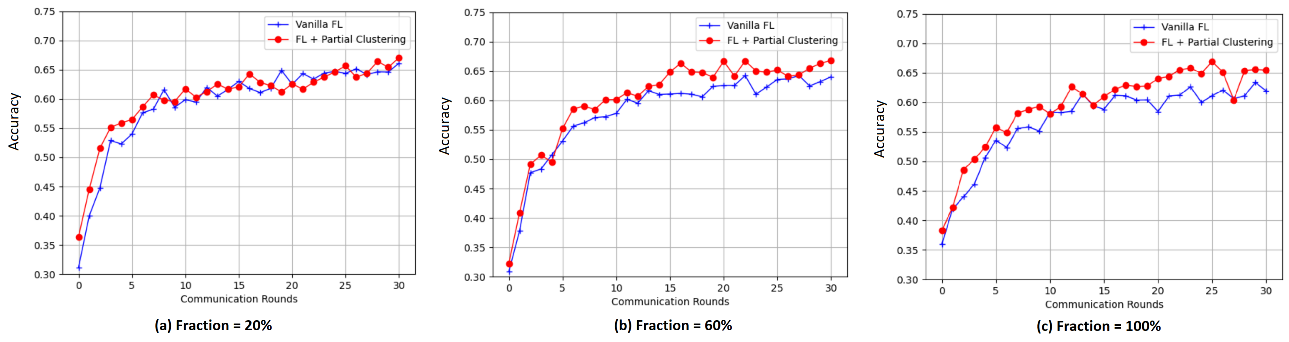


Figure 6.4 The importance of the fraction of the participants in the clustering step under concept-shift

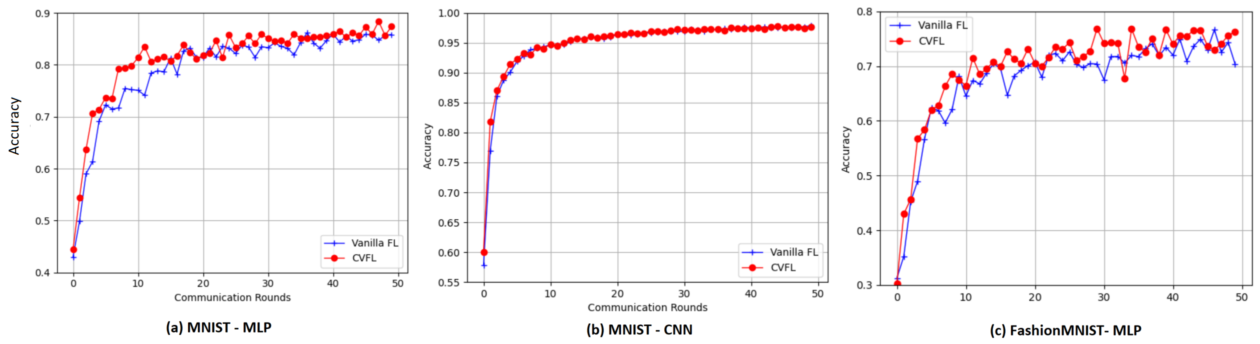


Figure 6.5 Evaluation of CVFL when the relationship is defined through mobility only

ble 6.2. The velocities of vehicles are assumed to be i.i.d, and they are generated by a truncated Gaussian distribution. In contrast to the normal Gaussian distribution or constant values, the truncated Gaussian distribution is more realistic for modelling vehicles' speed as it can generate different values in a certain limited range. This assumption is widely adopted in many state-of-the-art works of vehicular networks [169]. The lower and upper bounds for the velocity values on the 3 lanes going in the same direction are $(60, 80), (80, 100), (100, 120) km/h$.

## Key Performance results

In this part of the evaluation, we vary the model size and the number of RBs in order to evaluate how the CVFL algorithms adapts to different training and upload requirements. We evaluated how the number of selected cluster-heads and how the total number of participants change in each scenario. We

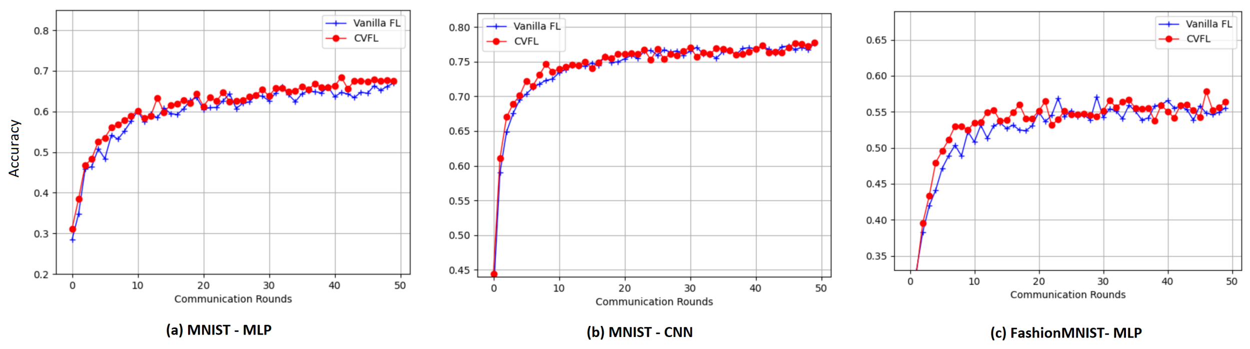


Figure 6.6 Evaluation of CVFL under concept shift

Table 6.2 Generated Values

| | |
|---|---|
| Vehicle Antenna height | 1.5m |
| Vehicle antenna again | 3dBi |
| Shadowing distribution | Log-normal |
| Shadowing standard deviation | 3 db |
| Noise power $N_0$ | -114 dBm |
| Fast fading | Rayleigh fading |
| Transmit Power | 0.1 Watt |
| Vehicles generation model | Spatial Poisson Process |
| Velocities generation model | Truncated Gaussian |
| Model Size | 160 kbits |
| Bandwidth/ RB | 180 Khz |
| $N_{max}$ | 2 |
| Total RBs | 4 |
| $\delta$ | 2s |

also evaluated how the average running time of the matching algorithm when the number of participants varies.

Table-6.3 shows the average number of cluster-heads selected in each communication round and Table-6.4 shows the average number of participants in each communication round. It is clear from the results that the number of RBs is the defining factor of the number of cluster-heads and consequently the number of participants. The results also show that the proposed algorithm can safely scale up to handle large models or more local epochs in the case of small models.

Table 6.3 Average Number of cluster heads in each communication round

| | Model Size in Kbits | | |
|---|---|---|---|
| Number of RBs | 160 | 320 | 640 |
| 2 | $2.43 \pm 0.26$ | $2.39 \pm 0.30$ | $2.39 \pm 0.24$ |
| 3 | $4.13 \pm 0.57$ | $4.056 \pm 0.51$ | $4.216 \pm 0.48$ |
| 4 | $5.565 \pm 0.69$ | $5.504 \pm 0.71$ | $5.568 \pm 0.54$ |

Table 6.4 Average Number of participants in each run

| | Model Size in Kbits | | |
|---|---|---|---|
| Number of RBs | 160 | 320 | 640 |
| 2 | $7.28 \pm 0.80$ | $7.16 \pm 0.91$ | $7.168 \pm 0.74$ |
| 3 | $12.26 \pm 12.05$ | $1.36 \pm 0.51$ | $12.44 \pm 1.39$ |
| 4 | $16.00 \pm 1.46$ | $15.81 \pm 1.47$ | $16.04 \pm 1.24$ |

Calculating the analytical expression of time complexity of the ILP-based algorithm used for the matching is not obvious since the low-level implementation details of the solver are not available to us. However, we evaluated the running time in different settings with varying the number of nodes to see how it scales with large number of participants. The average running time values in seconds on our machine are summarized in Table-6.5. In general, the matching algorithm can easily handle large pools of participants without high impact on the execution time.

**Effect on the accuracy**

To study the proposed approach in a mobility scenario, we first studied a simple case of unbalanced and non-i.i.d distribution, then we stress tested CVFL under concept-shift. The number of available RBs in each communication round is limited to 4, and the simulations were conducted for $i_{max} = 50$ communication rounds.

Table 6.5 Average Running time of the matching algorithm

| Number of vehicles | Average CPU time (s) |
|---|---|
| 25 | 0.02 |
| 50 | 0.03 |
| 75 | 0.03 |
| 100 | 0.03 |
| 125 | 0.04 |

Fig. 6.5 shows the results for unbalanced and non-i.i.d distribution in the mobility scenario.

Owing to larger numbers of participants (see Table 6.4), higher accuracy values are obtained across the experimnents. CVFL achieves accuracy of 87% ± 4% in contrast to 85% ± 5% for the standard FL under the same settings training MLP model on MNIST, and the CNN model achieves similar results for both CVFL (95% ± 5%) and vanilla FL (94% ± 7.5%). The average accuracy values on FashionMNIST is 69.66% ± 9% for CVFL and 66.46% ± 10% for vanilla FL. The larger values of the standard deviation of the results in vanilla FL across the experiments in this case is possibly due to the smaller number of participants in each round compared to CVFL where almost half of the vehicles train their models which provides more consistency throughout the experiments.

The second set of simulation runs are on unbalanced and non-i.i.d distribution with concept shift. Fig.6.6 shows how the models performed under these conditions in a freeway setting. Overall, accuracy values are significantly less than the obtained values in datasets where there is not concept shift. More specifically, the average accuracy of the MLP model achieved in the 50th round on MNIST dataset is 68% ± 9% in contrast to 65% ± 7% for vanilla FL. The CNN model yielded identical results for CVFL (80% ± 9%) and vanilla FL (80% ± 5%). The larger values of the standard deviations in CVFL are due to the fact that resulting models after clustering often perform differently on the test sets. The concept-shift appears to affect the accuracy on FashionMNIST in a higher level, as the accuracy drops to around 55% for both CVFL and vanilla FL. In contrast to the previous experiments, the difference is low in later rounds because only a small fraction of users participate in the clustering step. This can be overcome though the introduction of more communication rounds on the same version of the model in order to collect more updates. Additionally, the gaps in accuracy values are high in the earlier rounds of communication before the clustering round. The reason for the gap's narrowing is that new models are created and a smaller number of clients train the same model. As the number of clients training the models in each round constitutes a key factor for the convergence speed, we suspect that it might be the reason.

## 6.7 Limitations and future work

Through this work, we have identified several potential future research directions and open issues that are worthwhile being explored.

– **Large-scale collaboration:** Extending the proposed model to take into account handover between base stations etc in order to enable continuous training throughout vehicles' trips and reduce lost updates. Furthermore, fully decentralized training can be implemented for areas with low coverage, while also taking into consideration model convergence.

– **Adversarial attacks and outliers:** The updates' clustering is useful to detect local models that diverge from the majority of the received updates. This step can be furthered exploited to eliminate outliers and adversaries. Additionally, due to the collaborative and hierarchical nature of the proposed approach, trust among vehicles and reliability of their models can be further enhanced through traceability and incentive/punishment mechanisms.

– **Experimental values:** Set thresholds concerning LLT and rate of stay through experimental/ real data traces. Other values related to training can also be adjusted dynamically, such as the number of local epochs and the batch size.

– **Enhance Privacy:** While FL can provide some privacy concerning the raw data of each user, the model updates can be reverse-engineered to reveal sensitive information about the users. Several techniques such as

Differential privacy can be used to enhance the privacy-preservation in FL in vehicular environments.

## 6.8 Conclusion

In this paper, we have investigated the problem of clustered FL in vehicular networks. We aimed to fill the gap between clustering in vehicular networks and clustering in FL by designing a mobility-aware learning process for clustered FL. In the proposed architecture, we consider the v2v communication as an asset to overcome the communication bottleneck of FL in vehicular networks. Accordingly, in each communication round, a subset of vehicles are selected to act as cluster-heads, and the remainder of vehicles are matched with them. The selection favors vehicles with diverse datasets and good wireless communication channels with the gNodeB. Furthermore, clustering based on the similarity of the updates is introduced to subdue the slow convergence of single joint FL model in non-i.i.d settings, especially in the presence of concept-shift. This step leads to the creation of new models which are sent to the non-participants and newly joint vehicles, who will evaluate them and score their preferences of these models. The resulting preference values are used to match each vehicle to their preferred model (cluster-head). Both the cluster-head selection and cluster matching are formulated as optimization problems with learning goals and mobility constraints. We have proposed a greedy algorithm for the selection and RB allocation of cluster-heads, and a maximum weighted bipartite matching algorithm for the cluster formation. Simulations show the efficacy of using V2V communication to accelerate the learning as well as the importance of clustering based on updates to control concept shift. In the future, we aim to make the proposed approach resilient to outliers and malicious attacks such as false data injection.

**Algorithm 5** Clustered Vehicular Training procedure

```
1: for $i \in [1 \ldots i_{max}]$ do
2:    if $i = 1$ then
3:       Step 1:
4:       initialize or download the newest model's parameters at the MEC server
5:       initialize the number of models with 1
6:       Publish model and training requirements
7:    end if
8:
9:    Step 2: Receive vehicles information (transmit power, available data size, dataset
      diversity, CSI, velocity, preferred model)
10:   Schedule cluster-heads $H$ using Algorithm 1
11:   Step 3: Assign the remainder of vehicles (i.e., $NH$) to clusters using Algorithm 2
12:   Step 4:
13:   for vehicle $k \in NH$ do
14:      $k$ receives model $\theta_t$
15:      $k$ trains on local data $D_k$ for $\epsilon$ epochs
16:      $k$ sends updated model $\theta_k^{t+1}$ to MEC server
17:   end for
18:   for cluster head $h \in H$ do
19:      $h$ trains on local data $D_h$ for $\epsilon$ epochs
20:      $h$ receives model updates from vehicles in its cluster
21:      $h$ aggregates the model and sends new global model to MEC server
22:   end for
      Step 5:
23:   if $i = t_c$ then
24:      At step $i = t_c$ MEC server evaluates the similarities of the received models
25:      MEC server creates clusters based on the similarities and computes new global
         models using weighted average
26:      nodes receive new global models and evaluate their preferences
27:   end if
28:   aggregate updates
29:   start next round $i \leftarrow i + 1$
30: end for
```

**Algorithm 6** Clustered Vehicular Federated Learning (CVFL)

```
   Input  A queue of $K$ vehicles total available resource blocks $Total_{RB}$;
   Output  $\alpha$, $h = [h_1, \dots, h_K]$;
 1: // Cost Evaluation
 2: for $k = 1, \dots, K$ do
 3:     $r_k = 0, c = 1$;
 4:     order the RBs using $r_k, q$;
 5:     while $r_k \leq r_{k,min}$ and $c \leq Total_{RB}$ do
 6:         $q^* \leftarrow \arg\max_{q \in Z} G_{k,q}$;
 7:         $r_k \leftarrow r_k + r_{k,q}$;
 8:         $c \leftarrow c + 1$;
 9:         $c_k \leftarrow c$;
10:     end while
11: end for
12: return  $C = [c_1, \dots, c_K]$
13: // RB Allocation
14: order vehicles according to their ratio ($L = [\frac{I_k}{c_k} \forall k]$) decreasingly;
15: for $k = 1 \dots K$ do
16:     $h_k \leftarrow 0$;
17: end for
18: $A \leftarrow Z$;
19: $k \leftarrow \arg\max(L)$;
20: while $A \neq \oslash$ do
21:     order the RBs using $r_k, q$;
22:     while $r_k \leq r_{k,min}$ and $c \leq Total_{RB}$ do
23:         $q^* \leftarrow \arg\max_{q \in A} G_{k,q}$;
24:         $r_k \leftarrow r_k + r_{k,q}$;
25:         $\alpha_{k,q} \leftarrow 1$;
26:         $A \leftarrow A \setminus \{q\}$;
27:     end while
28:     $h_k \leftarrow 1$;
29: end while
30: return  $h$ and $\alpha$
31: // Matching
32: Use $h$ to form $H$ and $NH$ sets;
33: Infer values of $R_{k,h} \forall k \in NH, h \in H$;
34: Estimate $LLT_{k,h} \forall k \in NH, h \in H$;
35: verify time constraints and calculate $\zeta$;
36: Solve matching problem using Maximum weight bi-partite matching algorithm [192]
    using off the shelf solver such as Python's PulP [142].
37: Uniformly allocate the RBs of V2V links to the associated vehicles.
```

# Chapitre 7: Avant-propos

**Auteurs et affiliation:**

Afaf Taïk: étudiante au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Boubakr Nour: Chercheur post-doctoral, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

Soumaya Cherkaoui: Professeure, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Laboratoire de recherche INTERLAB.

**Date d'acceptation:** Septembre 2021.

**État de l'acceptation:** version finale publiée.

**Revue:** IEEE Wireless Communication.

**Titre français:** Autonomisation des communautés de prosommateurs dans les réseaux intelligents grâce aux communications sans fil et à l'apprentissage fédéré en périphérie.

**Résumé français:**

La croissance exponentielle des ressources énergétiques distribuées permet la transformation des consommateurs traditionnels du réseau électrique intelligent en prosommateurs. Cette transition offre une opportunité prometteuse pour le négoce d'énergie durable. Cependant, l'intégration des prosommateurs dans le marché de l'énergie impose de nouvelles considérations dans la conception de cadres unifiés et durables pour une utilisation efficace de l'infrastructure électrique ainsi que les ressources de communication. En outre, plusieurs problèmes doivent être résolus pour promouvoir de manière adéquate l'adoption de systèmes décentralisés orientés vers les énergies renouvelables, tels que le coût de communication, la confidentialité des données, l'évolutivité et la durabilité. Dans cet article, nous présentons les différents aspects et défis à relever pour construire des marchés de négoce d'énergie efficaces par rapport à la communication et la prise de décision intelligente. En tenant compte de ces défis, nous proposons un cadre de décisions proactives dans les communautés de prosommateurs afin d'atteindre des objectifs individuels et collectifs. Les décisions individuelles des prosommateurs sont d'abord motivées par des objectifs d'autosuffisance individuelle. C'est ainsi que le cadre donne la priorité aux décisions individuelles des prosommateurs et s'appuie sur le réseau sans fil 5G pour une coordination rapide entre les membres de la communauté. En fait, chaque prosommateur prédit la production et la consommation d'énergie pour prendre des décisions proactives en réponse aux demandes au niveau collectif. De plus, la collaboration de la communauté est étendue en incluant l'entraînement collaboratif des modèles de prédiction à l'aide de l'apprentissage fédéré (en anglais Federated Learning FL), assisté par des serveurs à la périphérie du réseau (en anglais multi-access edge computing MEC) et des équipements domestiques des prosommateurs. En plus de préserver les données privées des consommateurs, nous montrons par des évaluations que l'entraînement des modèles de prédiction à l'aide

de l'apprentissage fédéré permet d'obtenir une meilleure précision pour différentes ressources énergétiques tout en réduisant le coût de communication par rapport aux modèles centralisés.

**Mots clés:**

# Chapitre 7: Foreword

**Authors and affiliation:**

Afaf Taïk: Ph.D. Student, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Boubakr Nour: Postdoctoral Research Fellow, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

Soumaya Cherkaoui: Professor, INTERLAB Research Laboratory, Faculty of Engineering, Department of Electrical and Computer Science Engineering, Université de Sherbrooke.

**Date of acceptance:** September 2021.

**Acceptance status:** Final version published.

**Journal:** IEEE Wireless Communication.

**Title:** Empowering Prosumer Communities in Smart Grid with Wireless Communications and Federated Edge Learning [1]

1. https://github.com/afaf-taik/Smart-Grid-FL

# CHAPTER 7

# Empowering Prosumer Communities in Smart Grid with Wireless Communications and Federated Edge Learning

## 7.1 Abstract

The exponential growth of distributed energy resources is enabling the transformation of traditional consumers in the smart grid into prosumers. Such transition presents a promising opportunity for sustainable energy trading. Yet, the integration of prosumers in the energy market imposes new considerations in designing unified and sustainable frameworks for efficient use of the power and communication infrastructure. Furthermore, several issues need to be tackled to adequately promote the adoption of decentralized renewable-oriented systems, such as communication overhead, data privacy, scalability, and sustainability. In this article, we present the different aspects and challenges to be addressed for building efficient energy trading markets in relation to communication and smart decision-making. Accordingly, we propose a multi-level pro-decision framework for prosumer communities to achieve collective goals. Since the individual decisions of prosumers are mainly driven by individual self-sufficiency goals, the framework prioritizes the individual prosumers' decisions and relies on the 5G wireless network for fast coordination among community members. In fact, each prosumer predicts energy production and consumption to make proactive trading decisions as a response to collective-level requests. Moreover, the collaboration of the community is further extended by including the collaborative training of prediction models using Federated Learning, assisted by edge servers and prosumer home-area

equipment. In addition to preserving prosumers' privacy, we show through evaluations that training prediction models using Federated Learning yields high accuracy for different energy resources while reducing the communication overhead.

## 7.2 Introduction

The smart grid is an evolved power system with an integrated two-way flow of energy and information whose purpose is to enhance the efficiency of the system. The ubiquity of wireless communications has broadly promoted the deployment of sustainable power systems and helped monitor and control various operations of smart grids [196]. The higher performance and improved efficiency of fifth-generation (5G) communication networks are expected to reinforce this trend by delivering the ultra-low latency, the massive connectivity, and ultra-reliability required by smart grid services.

Communication technologies have not only changed the nature of the power system in terms of monitoring and operating processes, but they have also enabled the emergence of new behaviors and market models. In particular, the emergence of prosumers (*i.e.,* users that can produce and consume energy) is one of the strongest trends in the field of renewable energy in the smart grid. Multiple prosumers can now collectively create a *Prosumer Community Group* (PCG). PCG aims to producing and sharing energy in large amounts between users and utilities, providing a unified platform for information trading among neighbors within the local community, as well as interfacing with external prosumers and other energy entities. Indeed, PCG can effectively supply distributed generation, storage, and demand response [197]. Nevertheless, with the new opportunities offered by the advent of prosumers, unprecedented challenges are emerging in the management of energy production and demand in the grid. Indeed, the relationship between energy consumption and production is not always equiponderant. Hence, many techniques, including machine learning (ML), have been integrated to predict energy consumption, and adjust the available capacity accordingly. With the emergence of PCGs, there is also a need to forecasting energy production, which is becoming as important as predicting energy consumption, in order to adjust available resources effectively.

Current prediction systems in smart grids are based on collecting demand/production information and analyzing data at cloud servers, where the value chain relies on access to data. In 5G enabled smart grids, it is expected that prosumer systems will involve massive machine type communication (mMTC) where downlink communication is used for power control and uplink communication is used for data collection [198]. However, this data collection model results in considerably large datasets to be fed into the prediction system. More importantly, in many jurisdictions, privacy and data protection legislation requires the development of new operating models in which there is no personal data collection.

Federated Learning (FL) [199], an extension of machine learning, enables multiple devices/servers to collaboratively learn a prediction model in a distributed manner while keeping all training data locally on the device, thus decoupling machine learning capability from the need to store data in a centralized entity. In Federated Edge Learning (FEEL) [200], the global model training is performed at the Edge of the network. FEEL is particularly useful for delay-sensitive applications or congested backhaul networks, and it has been proposed as an interesting tool for household electrical load forcasting [201]. It provides a promising paradigm to enable decentralized learning without compromising data privacy.

The motivation behind this work is not only to design a privacy-aware integration mechanism for prosumers in 5G-enabled smart grid architectures but also to enable prosumers to make optimal decisions using short-term and long-term predictions, as well as allow PCGs to build reliable decision processes collaboratively that maximize the length of the energy production-consumption relationships. Coupling the reliable wireless communications provided by 5G networks with intelligent learning supported by FEEL will help maximize stakeholder's profits in the energy market.

The purpose of this work is twofold:

– To enable a multi-level decision process at the aggregator and the individual prosumer levels.

– To collaboratively build prediction models through federated learning and preserve the eco-system data privacy.

Specifically, the contribution of this work can be summarized as follows: (1) we discuss key elements for 5G empowered energy markets and highlight challenges related to their design, (2) we design a multi-stage energy forecasting framework and a decision process for PCGs empowered with FL using edge equipment, and (3) using real datasets, we evaluate the accuracy of load forecasts and the potential network load gain through simulations.

The remainder of this paper is structured as follows. The next section overviews prosumers' integration in the smart grid along with existing issues and challenges. Then, we present the proposed architecture and detail the decision process framework using FEEL. Later, we discuss the simulation and obtained results. Finally, we conclude the paper and present some future work.

## 7.3 Prosumers in Smart Grid: An Overview

In this section, we present the key elements for prosumers' integration in smart grids. We focus on the integration models, communication, and decisions and planning. Then, we present the existing issues and challenges in relation to these elements.

### 7.3.1 Key Elements in 5G Empowered Prosumers Markets

The increase in population and industrial/commercial sectors drives an unprecedented growth in energy demand. This expansion leads to peak times when most users are simultaneously using electricity.

To overcome the electricity shortages, users may contribute to energy production and feed the network when needed. A user that simultaneously produces and consumes energy is known as a *prosumer*. A prosumer can produce energy using small renewable energy sources and store it in battery banks or Electric Vehicles (EVs) [202]. In fact, an ever-increasing number of customers have local generation capability (*i.e.,* Distributed Energy Resources – DERs), in addition to several adaptable loads, such as thermostatically-controlled loads and distributed energy storage devices. Moreover, EVs are also appealing as controllable loads because they can be restricted for significant periods of time with no significant impact on end-use function.

The emergence of prosumers imposes new considerations for seamless smart grid integration. These considerations can be labeled into three levels: (i) integration and control model (*e.g.,* market model), (ii) communication (*e.g.,* requests, coordination), and (iii) pro-active decision making (*e.g.,* forecasting).

**Prosumers Integration Models:** The increasing number of prosumers imposes significant changes on the electricity market and provides various opportunities for exchanging and balancing electricity production and demand. Future strategies require an efficient integration of prosumers into the competitive electricity market. In fact, several promising approaches have been proposed, such as peer-to-peer models, indirect customer-to-customer trading, and prosumer community groups.

- Peer-to-Peer (P2P): In this model, prosumers are interconnected directly with each other in a completely decentralized manner. While this approach offers high flexibility and total autonomy, it comes with a high communication overhead underlying the search for suitable trading prosumers. Furthermore, an individual prosumer may not be able to produce energy that sufficient to match the fluctuating demand of a peer.
- Indirect Trading: Due to the nature of P2P networks, the search for suitable trading prosumers may be challenging and time-consuming. A potential solution to overcome this issue is energy brokers. The latter can be used to create a match between producers and consumers. However, relying on a central entity to manage the trading operations is not a scalable solution, as the broker will be overwhelmed when all the individual prosumers send the trading requests simultaneously.
- Prosumer Community Groups: On account of the freedom offered by the P2P paradigm and the organizational aspect of energy brokers, a group of prosumers can collectively – based on their behavior profiles and geographical structures, form a community known as PCG. The goals of a PCG are: (i) achieve a sustainable energy exchange, (ii) fulfill the energy demands for external customers, (iii) increase the income, and (iv) reduce the costs.

The integration of various DERs and EVs provides further opportunities for the development of innovative business models and energy ecosystems. However, the heterogeneous nature of energy sources [196] and their requirements urge novel approaches for prosumers' market design. Determining the integration model of the prosumers is indispensable to determine the required infrastructure for power distribution and communication.

**Communication:** Given the high uncertainties of consumption patterns, modern measurement units support fast monitoring with data refresh up to

50 times per second. Such updated frequencies open up enormous possibilities for fine-grained network control to support the complex energy markets' transition towards more decentralized renewable-oriented systems. Smart grids change the way energy is stored and delivered through vital information sharing among all the connected components. Home Area Network (HAN), smart meters, and home energy management systems (HEMSs) are largely used to prompt behavioral changes in energy consumption and storage, while among individual prosumers and PCGs, 5G networks facilitate information management through, for instance, the broadcast of information (*e.g.,* price signal, weather data). Therefore, a reliable communication system is critical for the management of smart grids.

**Pro-active Decisions and Planning:** Analyzing and predicting prosumers' behavior profiles are crucial during energy trading planning and future market design. Load forecasting is a key element for proactive decision-making, as it allows to measure the projected energy supply and the future consumption and thus change the operation strategy accordingly. Different horizon forecasts are subject to several studies in smart grids, as each serves a specific purpose according to the length of the forecast duration. For prosumer markets, the following forecast horizons are considered:

– Very short-term forecasting (seconds to minutes ahead) can be used for storage control for Vehicle-to-Grid (V2G), which are used to adjust small fluctuations. It is also necessary for electricity market clearing.

– Short-term forecasting (24 to 72 hours ahead) is crucial for key decision-making problems involved in the electricity market, such as economic dispatch and unit commitment.

– Medium and long-term forecasting (weeks to years ahead) are useful for maintenance scheduling in future systems and market planning.

An overall understanding of the factors that dominate prosumers' behaviors and their interactions within the smart grid enables designing an efficient decision-making framework. While ML techniques enable designing intelligent and responsive energy markets, many challenges still lack extensive studies.

### 7.3.2 Issues and Challenges

The proliferation of DERs yields many challenges to market design for proactive distribution systems. The limitations of existing work are not essentially based on their working principle; instead, they are more related to security, privacy, with seamless integration into 5G networks.

In the following, we dissect the existing challenges and explore the technical requirements for an optimal solution.

**Heterogeneous DERs:** Heterogeneous DERs, such as wind power and photo-voltaic (PV) are affected by different external conditions. For instance, solar radiation and solar altitude have substantial effects on PV, while wind power is directly related to wind speed. Since accurate predictions of these weather factors are challenging to obtain, their uncertainties inevitably lead to large errors when predicting DER production. Moreover, the topology information is often unavailable due to frequent changes at the individual prosumers level. For instance, EVs can appear once every few hours per day, making it hard for an aggregator to forecast EVs' willingness to share energy. Besides, these individual prosumers are unable to compete with traditional energy generators, as their energy supply is small and often unpredictable. The high fluctuations in the consumption and generation profiles are due to the volatility of the production and consumption profiles, which motivates coordination among prosumers on a shorter time scale and personalized level to attain more accurate results. The coordinated control of heterogeneous DERs can provide further opportunities for achieving better energy regulation.

**Consumption Load Forecasting:** While energy production depends heavily on uncertain weather conditions, consumption, on the other hand, is affected by the users' habits and behavior. Several prediction methods (*e.g.,* Support Vector Regression, Long Short-Term Memory) [201, 203] have been widely used in the literature to predict consumption profiles. Hybrid approaches combining time-series forecasting with other methods, such as decision trees and consumer clustering help improve the forecasting results. Nonetheless, the individual short-term load profiles remain uncertain and hard to predict compared to an aggregate prediction. Therefore, a PCG selling energy to the grid is more reliable in providing a more sustainable energy supply in contrast to individual prosumers.

**Communication Overhead:** As system operators need to assess the reliability of DERs and measure their participation to maintain the market equilibrium, reliable communication is a crucial enabler for smart energy markets. Moreover, the rapid growth of EVs adoption is paving the way for the deployment of at-scale smart EV chargers offering new ways to communicate with prosumers and near-real-time interfacing with the EV battery management system. However, due to the shift towards decentralized models and the rapid changes in the demand, the energy market imposes extreme communication requirements as metering and decision processes should be performed at very high frequencies. Several solutions have been designed to reduce the communication overhead, such as relying on reinforcement learning [203] and federated learning [201]. **Prosumer Group Formation:** The formation of stable PCGs is essential for sustainable energy sharing [204, 205] [206]. Therefore, it is essential to design strategies for optimal formation, growth, and management of PCGs. The first approach towards the formation of prosumer groups that can be considered is geographical proximity. As the trading market evolves, the similarity of prosumers' energy-sharing behaviors can be used to form stable coalitions. Additionally, the efficiency of PCGs can be further enhanced through unified goals, such as fulfilling the energy demands for external customers, maximizing profits, and reducing costs [205].

**User & Data Privacy:** One of the critical operations in energy market planning is knowledge extraction from historical consumption data. Nonetheless, these data contain sensitive information about users (*e.g.,* device usage, household occupancy), which imposes new requirements related to data privacy and necessitates the design of knowledge extraction methods immune to malicious interception and misuse. The centralized management paradigm is inadequate for the privacy-sensitive power distribution market. In contrast, the decentralized structure for the energy market has become an essential subject in smart grid literature. Additional techniques, such as Battery-based Data Masking, authentication and authorization, and Federated Learning, have been used in this setting to enhance privacy [207].

An inadequately planned prosumer market and poorly designed decision processes will severely impact consumer empowerment and the sustainability of energy markets. It is, therefore, crucial to take into account different issues when designing architectures and mechanisms for energy markets.

## 7.4 Smart Grid Prosumer Community Empowered Federated Edge Learning

### 7.4.1 System Architecture

Figure 7.1 illustrates a reference architecture for a smart grid. A smart grid is mainly built upon a bi-directional communication between users and utilities. The underlying connectivity between smart grid elements is ensured by a wireless 5G network via base stations and gNodeB. A centralized controller (*e.g.,* at the gNodeB level) is employed for traffic monitoring.

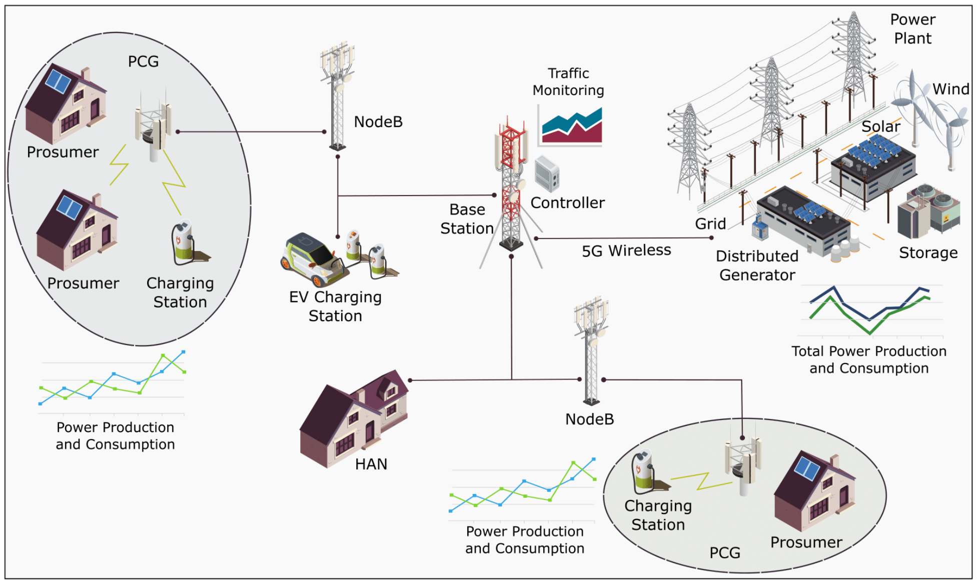


Figure 7.1 Prosumers in smart grid: overview and use-cases.

To ensure a sustainable energy infrastructure, the grid utilizes a Distributed Generator (DG) and incorporates heterogeneous DERs and produces a time-varying capacity. The integration of DERs requires controlling and monitoring, which can be achieved through computing and communication capabilities of edge and local devices. For instance, smart buildings integrate an agent CPU connected to HAN's sensors/actuators which allows a seamless management of the appliances.

A set of prosumers within the same administrative domain can collectively form a PCG. A PCG aggregates heterogeneous DERs' capacities and enables energy sharing among prosumers and with external entities, either for profit or for free [205]. However, due to the heterogeneity of the DERs, price fluctuations, the automation of energy sharing is rather challenging and requires a high level of coordination among different entities. As a result, information (*e.g.,* weather data, pricing data) and commands flow between the utility, the controller, and prosumers.

The main objective is to efficiently utilize each energy source. For example, during the day, PV can be considered the primary source of energy, while

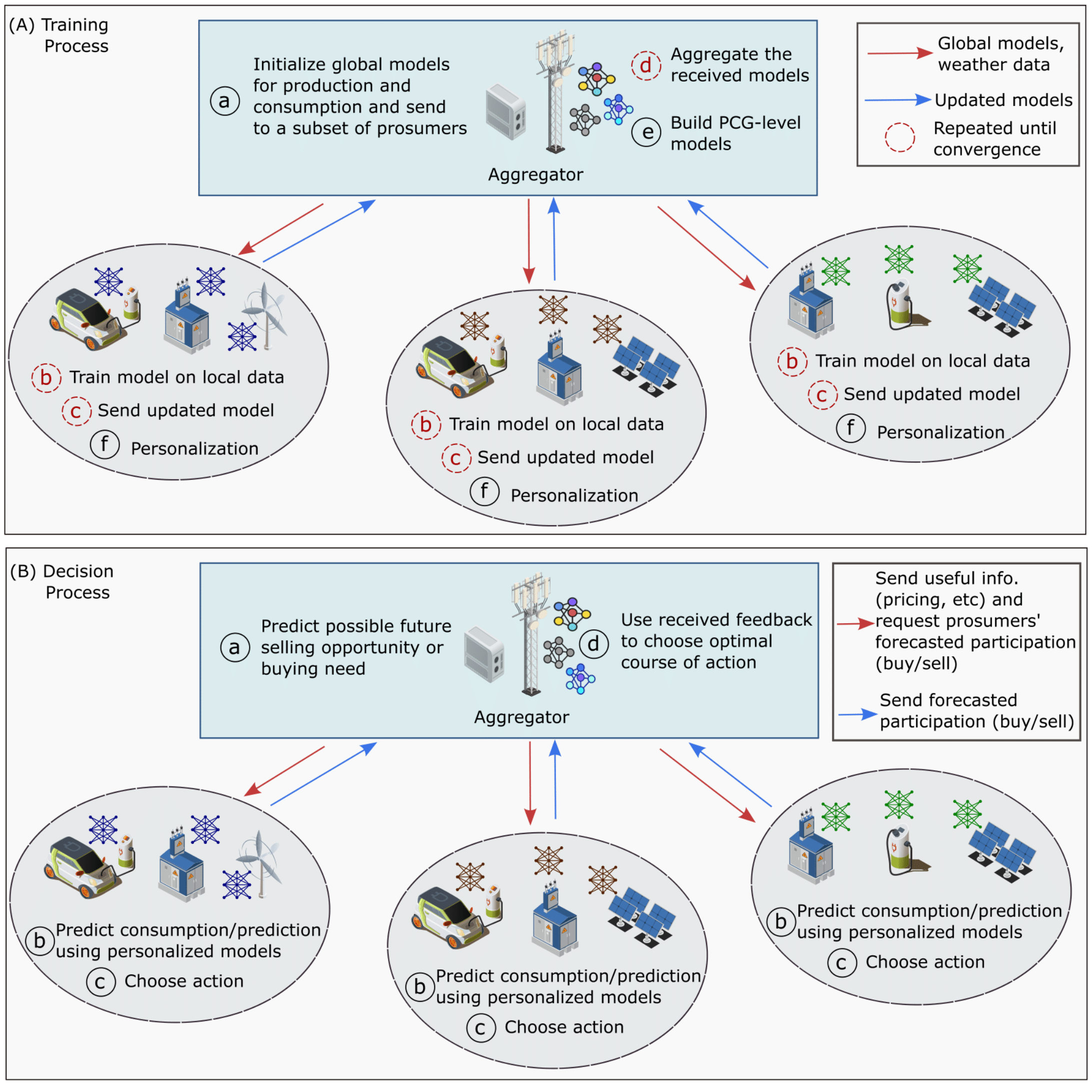

(A) Training Process
a Initialize global models for production and consumption and send to a subset of prosumers
Aggregator
d Aggregate the received models
e Build PCG-level models
Global models, weather data
Updated models
Repeated until convergence
b Train model on local data
c Send updated model
f Personalization
(B) Decision Process
a Predict possible future selling opportunity or buying need
Aggregator
d Use received feedback to choose optimal course of action
Send useful info. (pricing, etc) and request prosumers' forecasted participation (buy/sell)
Send forecasted participation (buy/sell)
b Predict consumption/prediction using personalized models
c Choose action

Figure 7.2 Training and decision processes of the proposed scheme.

batteries, more specifically the V2G model, can help regulate the needs during the day and be the most relied on at night. In doing so, a PCG utilizes forecasted consumption and production values to determine whether the production goals can be met and make trading decisions accordingly.

In this article, we design a multi-stage framework for collaborative and sustainable energy sharing for PCGs.

Figure 7.2 shows in detail the two processes of the proposed framework: Training and Decision. The training is performed using FEEL, which is mainly privacy-preserving collaborative training of the load forecasting models. While in the decision process, individual prosumers use the resulting models to make predictions to make optimal decisions.

### 7.4.2 Training Process: FEEL-based Short-term Load Forecasting

Current energy trading markets use one-day ahead predictions. However, these predictions are often inaccurate, which in consequence leads to unmatched energy production. By leveraging the HAN edge equipment and the rapid 5G communication, decisions can be taken in near-real-time to match the unpredictable demand [208]. For instance, EVs charging loads are small and unpredictable in the long-term [209], making them inadequate for one-day ahead planning in the V2G context. In contrast, the solar power production curve over a day is usually bell-shaped and can be easily predicted based on weather data. Consequently, prediction horizons should be chosen depending on the predictability of each DER type and the required data to make the prediction. Values related to residents' behaviors should have shorter prediction horizons (*e.g.,* a few seconds or minutes ahead ) compared to values that depend on the weather (*e.g.,* up to several hours ahead).

For each prosumer, the prediction model takes as an input a series of surplus power data for the last few intervals and produces an estimation of the surplus or shortage power in the next interval. Targeting more accurate predictions, we use separate models for production and consumption to calculate the estimate. Energy production depends mostly on weather conditions. The required meteorological data for forecasting can be easily obtained by local weather stations and broadcasted by the PCG aggregator. DER power prediction uses models that map historical data to output power through mining the potential rules and relationships of the training data. For instance, the PV production curve is usually bell-shaped with fluctuations during the day depending on solar irradiance, whereas wind energy depends on wind speed and direction. While these factors can be forecasted accurately a few hours ahead, load profiles for EVs and consumption can only be forecasted up to a few minutes ahead due to individual behaviors' uncertainty.

Moreover, the model should be adapted for each individual prosumer. In fact, the consumption load depends on the individual prosumer's habits, whereas for energy, there are additional factors that affect production, such as the direction solar panels are facing and their position. Due to the scarcity of local data, collaboration among PCG members, in addition to energy sharing, involves a distributed training of the prediction models. FEEL allows training models among the PCG members at the edge of the network while preserving the privacy of the prosumers by keeping data locally [201]. Training models using FEEL ensures that the obtained model is not overfitting as it is trained on a large dataset. In FL, the training of the model takes several communication rounds before converging. Typically, a global model is initialized by the PCG aggregator and sent to a subset of the prosumers who independently train the model using their local historical data and upload their gradient updates to the PCG aggregator. The participating subset is ideally selected at random, however, due to the communication bottleneck in wireless edge networks, the selection is based on the wireless channel state, the computing resources, and the amount and quality of the training data. The received local updates are aggregated by averaging, and a global model is obtained. Afterward, the aggregator sends the global model to a new subset of prosumers, and a new iteration begins where each device computes the gradient updates and uploads it, until the model converges. The final model is then broadcasted to the community. To make the model personalized for each prosumer, the obtained model is retrained locally for each prosumer. The FEEL process is repeated periodically to adapt the models to different changes in the prosumers' side (*e.g.,* new appliances, different habits) and the external conditions.

### 7.4.3 Multi-stage Decision Process

As mentioned above, the PCG leverages communication and energy storage to achieve collective values and goals. However, individual decisions of prosumers are based first on individual self-sufficiency goals. To maintain PCG stability, the trading decisions should take into account the individual prosumers' decisions. As a result, the decision process for energy trading comprises two distinct levels:

**At the aggregator level:** Based on the overall predicted values of production and consumption, the aggregator decides the next action regarding energy trade: (i) sell energy if the local production is higher than the overall consumption, (ii) buy energy from external parties if the local production is less than the overall consumption, and/or (iii) request V2G to regulate small fluctuations in the demand. Regardless of the preliminary decision, the aggregator requests local decisions made by individual prosumers.

**At the individual prosumer level:** By using local prediction models, each prosumer estimates the difference between short-term local production and overall consumption. The predicted value serves as a basis for the prosumer's participation or to signal a projected shortage. These values are then sent to the aggregator for finalizing the trading decisions. Individual prosumers may develop different decisions processes depending on their goals and preferences. For instance, a prosumer with altruistic values would use the energy infrastructure to share electricity with the community rather than selling it. In contrast, a prosumer with monetary goals would use it to sell electricity rather and gain revenue than share it.

Forecasting the difference between consumption and production has a significant influence over prosumer decisions for a subsequent period, which is indeed a fundamental operation for all decisions. For production, and for each type of DERs, different factors can be considered in the decision process. For example, each prosumer in V2G may take into account the battery level and estimate the length of time the vehicle will still be charging.

## 7.5 Numerical Results

This section first introduces the used dataset and then describes the prediction model and implementation details. Later, we discuss the evaluation results.

**Dataset:** This research was conducted using real data from the Pecan Street Inc. [210]. We used circuit-level electricity use data at 15 minutes intervals for a PCG, with PV generation and EV charging data for a subset of the PCG. The PCG is simulated by a subset of 18 prosumers who have similar properties from this dataset. A subset of 5 prosumers has EV data, and comprises the same kind of houses (detached-family homes), located in the same areas (*i.e.,* Austin, Texas). The dataset is composed of records between May 2018 and August 2018, with 15 minutes resolution data for PV and overall consumption. We also used one-minute resolution data of EV consumption over a period of two weeks.

**Prediction Models:** We used identical models for consumption and solar energy production (*Model 1*), where each has two Long-Short Term Memory (LSTM) hidden layers composed of 128 neurons each. We used the Mean Squared Error as the Loss function and Adam as the optimizer. The models are trained using normalized data, transformed into sliding windows using 48-time steps to predict the next value. For the EVs, we trained a similar model (*Model 2*) with 200 LSTM cells in each layer, using 15 past minutes to predict the next 5 minutes.

We used 90% of data for training and 10% for testing. Each model is trained over 25 rounds and retrained using 8 epochs locally, with a subset of 5 prosumers chosen randomly in each round. The simulations were conducted on a Laptop with a 2.6 GHz Intel i7 Processor, 16GB of RAM memory, and

NVIDIA GeForce RTX 2070 graphic card. We used Tensorflow Federated 0.4.0 with Tensorflow 1.13.1 backend.

**Results:**

*1) Aggregator level models:* The aggregator has access to the overall consumption of the PCG and the production of the DERs. In our case, it uses the prediction of the consumption alongside the forecasted PV power to make decisions. Figure 7.3 shows the prediction and the actual power for the PCG for the first 24 hours in the test set. The Root Mean Square Error (RMSE) of the PV power is negligible 3.99 $W$, whereas the overall consumption can be predicted with an RMSE of 4.71 $W$.

*2) Prosumer level models:* In order to evaluate the models obtained using FEEL, we compared the average RMSE of these models to the average RMSE of a centrally trained model. Table 7.1 summarizes the obtained RMSE for different models. In our case, the load forecast is on a granular level (single house) and on a very short term (*i.e.,* 15 minutes or less); therefore, the values of RMSE are achieved in Table 7.1 for various models are reasonable. The error margin is anticipated as similar values have been reported by previous work including ours [201].

Table 7.1 Average Power RMSE for the PCG.

| | **Central Model** | **Personalized Models** |
|---|---|---|
| **PV** | $0.25 \pm 0.05$ | $0.18 \pm 0.04$ |
| **Consumption** | $0.84 \pm 0.28$ | $0.65 \pm 0.23$ |
| **EV (1min)** | $0.13 \pm 0.04$ | $0.12 \pm 0.04$ |
| **EV (5min)** | $0.265 \pm 0.04$ | $0.277 \pm 0.04$ |

Figure 7.4 illustrates the improvements on predictions using personalization with FEEL for a prosumer from the PCG. Both models (PV and consumption) fit the actual data of the prosumer. As energy trading has high accuracy

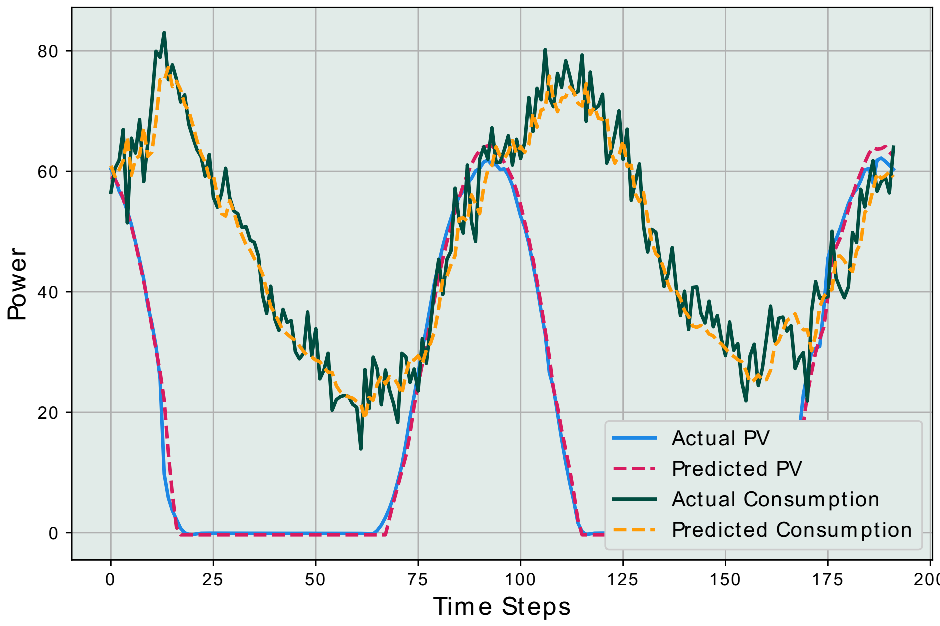


Figure 7.3 Overall prediction of production and consumption of the PCG at the aggregator level.

requirements, the improvement in the precision of the prediction will have a significant impact.

*3) Communication and Computation Overhead:* Figure 7.5 shows the total data size sent over wireless networks when using FEEL compared to sending the data in one-minute resolution. While scheduling a larger number of prosumers in each round is preferable, more updates are exchanged over the network leading to costly communication (*e.g.,* in terms of bandwidth, delay, and quality of experience). By adopting FEEL, where training is performed at the edge level, data exchange is less costly than frequent data uploads to a central server. With the growth of the need of more granular data, the gain when it comes to communication will be even more significant. Additionally, several compression and partial participation techniques can be explored to further reduce the communication overhead [199].

For the computation overhead, our configuration is able to predict the next step for PV and consumption using *Model 1* just by taking an average of

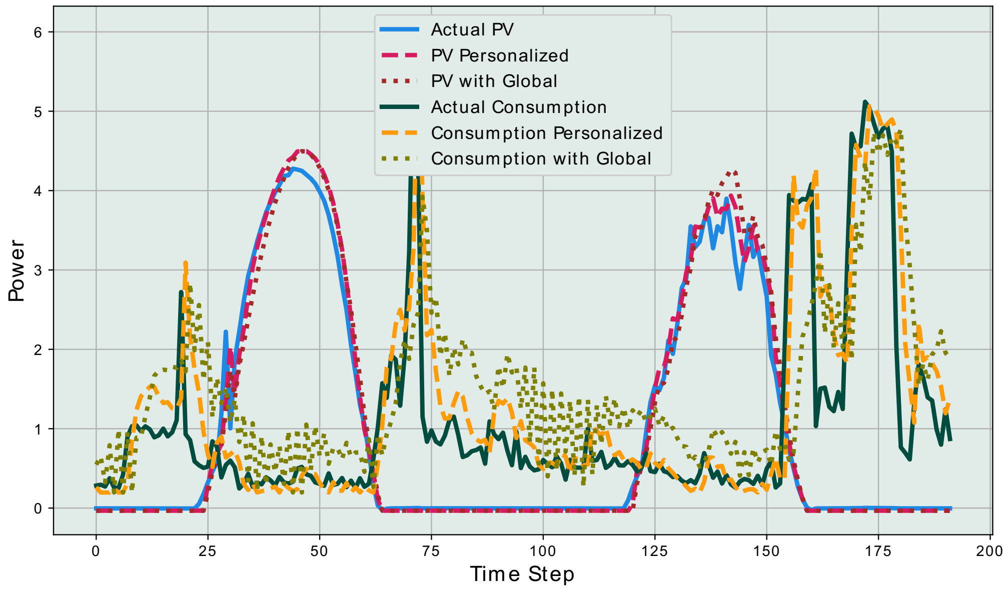


Figure 7.4 Prediction of production and consumption of the first 24 hours in the test set for a sample prosumer.

0.37 $ms$, while predicting the next 5 steps using *Model 2* for EV power takes an average of 38 $ms$. The multi-step prediction is a costly operation and requires deeper and more powerful models. Nonetheless, it remains necessary to maintain a sustainable prosumer participation.

## 7.6 Open Issues and Future Research Directions

Through this work, we have identified different open issues and interesting research directions that deserve further investigation:

**Prosumers Regrouping:** Individual prosumer behavior is impacted by the community's organization, norms, and goals. A sustainable PCG can be extended through virtualization techniques to make a match among a wider number of prosumers who share similar goals and preferences.

**Peer to Peer trading:** To achieve a fully decentralized collaboration, Peer-to-Peer FL can be leveraged in the context of smart grid and energy markets. This approach removes the single point of failure that can be inherent in a

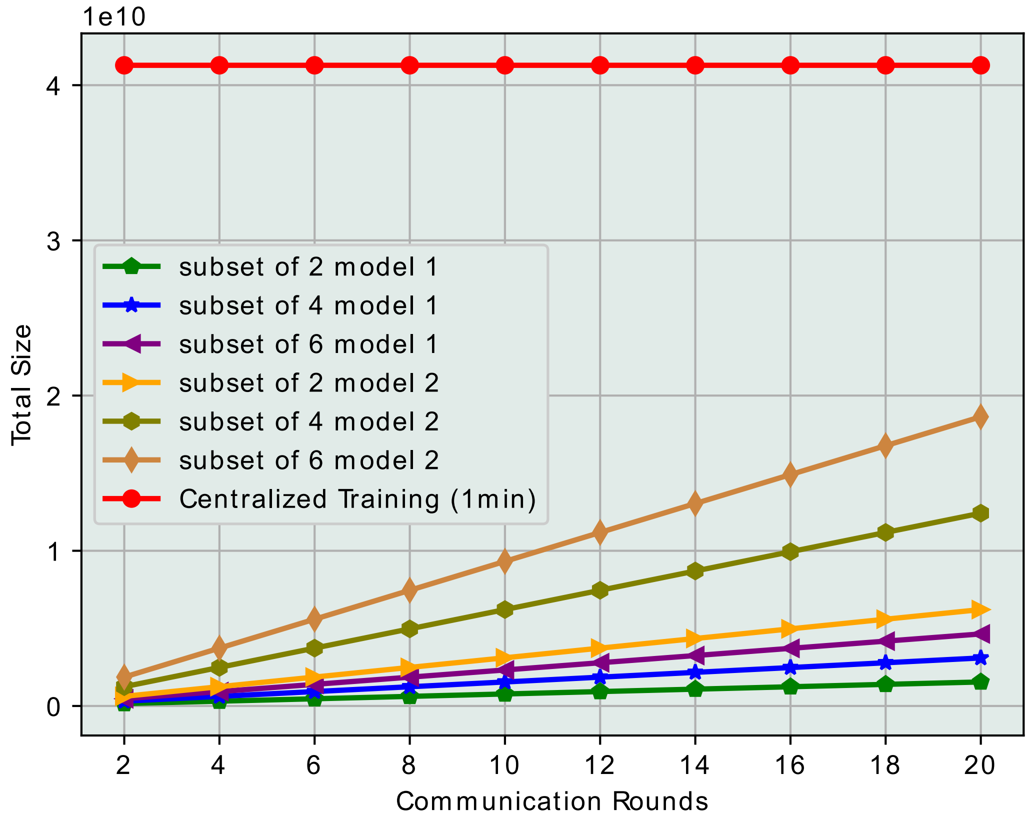


Figure 7.5 Total size of data exchanged over wireless networks when training models using FEEL vs. centralized training.

PCG aggregator-based system. Furthermore, blockchain technology can be used for reliable transactions [206].

**Privacy and Security:** Although FEEL is designed with privacy in mind, it is still vulnerable to several attacks such as Backdoor attacks and Poisoning attacks. It is, therefore, necessary to investigate mechanisms for more reliable and resilient FEEL in smart grid environments.

## 7.7 Conclusion

Managing prosumers over wireless networks in smart grids is a challenging task that requires proactive decisions and optimal planning. Given the stochastic nature of consumption and production load profiles and the privacy-sensitive aspect of these data, building predictive models for energy trading automation becomes a challenging operation. In this article, we presented the key enablers for integrating prosumers in the energy market. We discussed several challenges and issues concerning communication, privacy, and planning. Keeping these issues in mind, we designed a multi-stage energy forecasting framework using decentralized decisions based on short-term predictions. By leveraging edge equipment, we show that FEEL is a promising solution for tackling privacy challenges related to model training in PCGs. Furthermore, we proposed the integration of individual prosumers in the energy trading decisions as a key factor for a sustainable PCG.

# CHAPTER 8

# Conclusions and Future Work

In this thesis, we set several objectives related to paving the last mile of EI. Accordingly, we presented several approaches to enable FL in wireless edge networks under challenges related to data and resources, and demonstrated the potential of FEEL in and IoT scenarios. This chapter concludes the thesis and suggests new research directions for future work.

## 8.1 Conclusions

The first goal of this thesis was to design a suitable metric to evaluate how informative and rich a local dataset is, without revealing sensitive information about the data. In chapter 3, several diversity metrics were identified, and a general framework for FEEL was proposed. In chapter 4, we investigated how to select clients with potentially more informative datasets while also optimizing the communication round's time and total required energy. This is captured through a multi-objective optimization problem that was solved with an iterative algorithm which we evaluated through extensive experiments. The proposed diversity indicator is then extended with a reputation score, thus avoiding malicious clients launching data poisoning attacks.

The second goal was to enhance the FL process to handle use-cases with high mobility, namely FL in vehicular networks, and data concept-shift, where a single model might not be enough. To tackle these challenges, we proposed in chapter 6, a clustered process for vehicular FL. The process consists of an optimized cluster formation, starting with cluster-head selection, then matching the remainder of vehicles to the selected cluster-heads. The matching algorithm takes into consideration the velocity of the vehicles and their direction, and can be adapted in the presence of more than one model.

Lastly, since most of the work in FL was simulated using synthetically distributed datasets, and work on FEEL deployments is still in its infancy, it is necessary to evaluate the potential of FEEL in real use-cases. Accordingly, we designed a pro-active and collaborative decision process for energy trade in smart grid. The decision is based on energy consumption and production forecasts which we train using FEEL and enhance with personalization. The gains in accuracy and communication costs make it appealing to adopt FEEL in smart grid scenarios in particular, and smart city in general.

## 8.2 Future Work

The work presented in this thesis can enable several future research directions but also let us raise new research questions. We list in the following some perspectives for future work.

- In chapter 4, the designed diversity index is a weighted sum of several measures and can be easily modified to include other metrics depending on the use-case. A dynamic and adaptive way to choose the weights can be envisioned for an optimal client selection.
- In chapter 5, we studied the case of malicious clients launching data poisoning attacks. However, this approach might wrongfully consider an outlier as a malicious entity. In the case of the establishment of an incentive or punishment mechanism, it becomes critical to make this distinction. Additional mechanisms should be implemented to handle outliers in this case.
- In chapter 6, we studied FL in vehicular networks in the case of one BS. The proposed process can be extended for continuous training through handovers, and include a larger scale collaboration using cloud based orchestration. Additionally, some parameters can be chosen using experimental values, such as the training deadlines and the number of local epochs.

- In chapter 7, we leveraged the geographical grouping of the clients in the smart grid and the similarity of the type of the building to create a community training the model collaboratively. Other clustering features can be used in future work to enhance the quality of the resulting models.

# CHAPTER 9
# Conclusions et travail futur

Dans cette thèse, nous avons fixé plusieurs objectifs liés à FL comme étant la dernière pièce du puzzle qui parachève l'intelligence artificielle à la périphérie du réseau. En conséquence, nous avons présenté plusieurs approches pour faciliter FL dans les réseaux périphériques sans fil, en tant compte des défis liés aux données et aux ressources. Nous avons démontré le potentiel de FEEL dans et les scénarios liés à l'internet des objets. Ce chapitre conclut la thèse et suggère de nouvelles directions de recherche pour les travaux futurs.

## 9.1 Conclusions

Le premier objectif de cette thèse était de concevoir une métrique appropriée pour évaluer le degré d'information et de richesse d'un ensemble de données locales, sans révéler d'information sensible sur les données. Ainsi, dans le chapitre 3, plusieurs métriques de diversité ont été identifiées, et un cadre général pour FEEL a été proposé. Dans le chapitre 4, nous avons étudié comment sélectionner les clients avec des ensembles de données potentiellement plus informatifs tout en optimisant la durée de chaque itération et l'énergie totale requise. Ceci est formulé sous forme d'un problème d'optimisation multi-objectif qui a été résolu avec un algorithme itératif que nous avons évalué par des simulations approfondies. Nous avons ensuite, dans le chapitre 5, combiné l'indicateur de diversité proposé avec un score de réputation, ce qui permet d'éviter les clients malveillants qui ont des données empoisonnées.

Le second objectif était d'améliorer le processus FL pour gérer les cas d'utilisation à forte mobilité, à savoir le FL dans les réseaux véhiculaires, et le changement de concept des données, où un seul modèle peut ne pas suffire. Pour relever ces défis, nous avons proposé dans le chapitre 6, un processus en grappes pour le FL véhiculaire. Le processus consiste en une formation optimisée de groupes, en commençant par la sélection de nœuds agissant comme coordinateurs de groupes, puis en faisant correspondre le reste des véhicules aux coordinateurs sélectionnées. L'algorithme d'appariement prend en compte la vitesse des véhicules et leur direction, et peut être adapté en présence de plus d'un modèle.

Enfin, étant donné que la plupart des travaux sur FEEL ont été simulés à l'aide d'ensembles de données distribuées de manière synthétique et que les travaux sur les déploiements de FEEL en sont encore à leurs débuts, nous avons jugé essentiel d'évaluer le potentiel de FEEL dans un cas d'utilisation réel. Plus précisément, nous avons conçu un processus de décision proactif et collaboratif pour le négoce d'énergie dans un réseau électrique intelligent. La décision est basée sur les prévisions de consommation et de production d'énergie que nous formons à l'aide de FEEL et que nous améliorons par la personnalisation. FEEL permet de renforcer la collaboration entre les membres des groupes de clients dans le réseau électrique intélligent. Les gains en précision et en coûts de communication rendent attrayante l'adoption de FEEL dans les scénarios de réseau intelligent en particulier, et de ville intelligente en général.

## 9.2 Directions futures

Le travail présenté dans cette thèse nous a permis d'identifier plusieurs directions de recherche potentielles et de soulever de nouvelles questions de recherche. Nous énumérons dans ce qui suit quelques perspectives de travaux futurs.

- Dans le chapitre 4, l'indice de diversité conçu est une somme pondérée de plusieurs mesures et peut être facilement modifié pour inclure d'autres mesures en fonction du cas d'utilisation. Une manière dynamique et adaptative de choisir les poids peut être envisagée pour une sélection optimale des clients.

- Dans le chapitre 5, nous avons étudié le cas de clients malveillants lançant des attaques par empoisonnement des données. Cependant, cette approche pourrait considérer à tort des clients possédant des données aberrantes comme une entité malveillante. Dans le cas de la mise en place d'un mécanisme d'incitation ou de punition, il devient critique de faire cette distinction. Des mécanismes supplémentaires devraient être mis en œuvre pour traiter les données aberrantes dans ce cas.
- Dans le chapitre 6, nous avons étudié FL dans les réseaux véhiculaires dans le cas d'une seule station de base. Le processus proposé peut être étendu pour maintenir l'apprentissage entre plusieurs stations de base par le biais de transferts, ainsi que l'orchestration basée sur l'infonuagique. De plus, certains paramètres, tel que le nombre d'itérations locales, pourront être choisis dynamiquement en se basant sur des valeurs expérimentales.
- Au chapitre 7, nous avons tiré parti du regroupement géographique des clients du réseau électrique intelligent et la similarité du type de bâtiment pour créer une communauté collaborative. D'autres caractéristiques de regroupement peuvent être utilisées dans des travaux futurs pour améliorer la qualité des modèles obtenus.

# LIST OF REFERENCES

[1] David Reinsel, John Gantz, and John Rydning. The Digitization of the World from Edge to Core, 2018. [Online] https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf , Accessed : 28-12-2021.

[2] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal*, 3(5):637–646, October 2016.

[3] Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge Intelligence: Paving the Last Mile of Artificial Intelligence with Edge Computing. *arXiv:1905.10083 [cs]*, May 2019. arXiv: 1905.10083.

[4] Xiaofei Wang, Yiwen Han, Victor C. M. Leung, Dusit Niyato, Xueqiang Yan, and Xu Chen. Convergence of Edge Computing and Deep Learning: A Comprehensive Survey. *arXiv:1907.08349 [cs]*, December 2019. arXiv: 1907.08349.

[5] Yaohua Sun, Mugen Peng, Yangcheng Zhou, Yuzhe Huang, and Shiwen Mao. Application of machine learning in wireless networks: Key techniques and open issues. *IEEE Communications Surveys Tutorials*, 21(4):3072–3108, 2019.

[6] Mugen Peng and Kecheng Zhang. Recent advances in fog radio access networks: Performance analysis and radio resource allocation. *IEEE Access*, 4:5003–5009, 2016.

[7] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. *arXiv:1610.05492 [cs]*, October 2017.

[8] Prateek Singhal, Prabhat Kumar Srivastava, Arvind Kumar Tiwari, and Ratnesh Kumar Shukla. A Survey: Approaches to Facial Detection and Recognition with Machine Learning Techniques. In Deepak Gupta, Ashish Khanna, Vineet Kansal, Giancarlo Fortino, and Aboul Ella Hassanien, editors, *Proceedings of Second Doctoral Symposium on Compu-*

*tational Intelligence*, Advances in Intelligent Systems and Computing, pages 103–125, Singapore, 2022. Springer.

[9] Andrew Hard et al. Federated learning for mobile keyboard prediction, 2019.

[10] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21918.

[11] Mahadev Satyanarayanan. The emergence of edge computing. *Computer*, 50(1):30–39, 2017.

[12] A. Filali, A. Abouaomar, S. Cherkaoui, A. Kobbane, and M. Guizani. Multi-Access Edge Computing: A Survey. *IEEE Access*, 8:197017–197046, 2020. Conference Name: IEEE Access.

[13] Davide Borsatti, Gianluca Davoli, Walter Cerroni, and Carla Raffaelli. Enabling Industrial IoT as a Service with Multi-Access Edge Computing. *IEEE Communications Magazine*, 59(8):21–27, 2021.

[14] Hongjun Dai, Xiangyu Zeng, Zhilou Yu, and Tingting Wang. A scheduling algorithm for autonomous driving tasks on mobile edge computing servers. *Journal of Systems Architecture*, 94:14–23, 2019.

[15] Alaa Awad Abdellatif, Amr Mohamed, Carla Fabiana Chiasserini, Mounira Tlili, and Aiman Erbad. Edge computing for smart health: Context-aware approaches, opportunities, and challenges. *IEEE Network*, 33(3):196–203, 2019.

[16] Mohammad Aazam and Eui-Nam Huh. Fog Computing Micro Datacenter Based Dynamic Resource Estimation and Pricing Model for IoT. In *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, pages 687–694, March 2015. ISSN: 2332-5658.

[17] Flavio Bonomi, Rodolfo Milito, Preethi Natarajan, and Jiang Zhu. Fog Computing: A Platform for Internet of Things and Analytics. In Nik Bessis and Ciprian Dobre, editors, *Big Data and Internet of Things: A Roadmap for Smart Environments*, Studies in Computational Intelligence, pages 169–186. Springer International Publishing, Cham, 2014.

[18] Sabine Dahmen-Lhuissier. ETSI - Multi-access Edge Computing - Standards for MEC.

[19] Malathi Veeraraghavan, Takehiro Sato, Molly Buchanan, Reza Rahimi, Satoru Okamoto, and Naoaki Yamanaka. Network Function Virtualization: A Survey. *IEICE Transactions on Communications*, E100.B(11):1978–1991, 2017.

[20] Diego Kreutz, Fernando M. V. Ramos, Paulo Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-Defined Networking: A Comprehensive Survey. *arXiv:1406.0440 [cs]*, October 2014. arXiv: 1406.0440.

[21] Liangzhi Li, Kaoru Ota, and Mianxiong Dong. DeepNFV: A Lightweight Framework for Intelligent Edge Network Functions Virtualization. *IEEE Network*, 33(1):136–141, January 2019.

[22] Ying-Dar Lin, Chih-Chiang Wang, Chien-Ying Huang, and Yuan-Cheng Lai. Hierarchical CORD for NFV Datacenters: Resource Allocation with Cost-Latency Tradeoff. *IEEE Network*, 32(5):124–130, September 2018.

[23] Shuiguang Deng, Hailiang Zhao, Jianwei Yin, Schahram Dustdar, and Albert Y. Zomaya. Edge Intelligence: the Confluence of Edge Computing and Artificial Intelligence. *arXiv:1909.00560 [cs]*, September 2019. arXiv: 1909.00560.

[24] Léon Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD.

[25] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of machine learning research*, 12(7):39, 2011.

[26] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017. arXiv: 1412.6980.

[27] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, May 2017.

[29] Shilpa Lakhanpal, Ajay Gupta, and Rajeev Agrawal. Discover trending domains using fusion of supervised machine learning with natural language processing. In *2015 18th International Conference on Information Fusion (Fusion)*, pages 893–900, July 2015.

[30] Peng Yang and Yunfang Chen. A survey on sentiment analysis by using machine learning methods. In *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pages 117–121, December 2017.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper With Convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[33] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[34] Peter Kairouz et al. Advances and open problems in federated learning, 2021.

[35] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K. Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive Federated Learning in Resource Constrained Edge Computing Systems. *IEEE Journal on Selected Areas in Communications*, 37(6):1205–1221, June 2019.

[36] Keith Bonawitz et al. Towards federated learning at scale: System design, 2019.

[37] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv:1704.04861 [cs]*, April 2017. arXiv: 1704.04861.

[38] Haichen Shen, Matthai Philipose, Sharad Agarwal, and Alec Wolman. MCDNN: An Execution Framework for Deep Neural Networks on Resource-Constrained Devices. *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, page 123–136, 2016.

[39] Edoardo Ragusa, Christian Gianoglio, Rodolfo Zunino, and Paolo Gastaldo. A Design Strategy for the Efficient Implementation of Random Basis Neural Networks on Resource-Constrained Devices. *Neural Processing Letters*, 51(2):1611–1629, April 2020.

[40] Nir Shlezinger, Mingzhe Chen, Yonina C. Eldar, H. Vincent Poor, and Shuguang Cui. Federated learning with quantization constraints. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8851–8855, 2020.

[41] Jingyan Jiang and Liang Hu. Decentralised federated learning with adaptive partial gradient aggregation. *CAAI Transactions on Intelligence Technology*, 5(3):230–236, May 2020. Publisher: IET Digital Library.

[42] Jason Posner, Lewis Tseng, Moayad Aloqaily, and Yaser Jararweh. Federated Learning in Vehicular Networks: Opportunities and Solutions. *IEEE Network*, 35(2):152–159, March 2021. Conference Name: IEEE Network.

[43] Afaf Taïk and Soumaya Cherkaoui. Electrical load forecasting using edge computing and federated learning. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.

[44] Mike Lakoju, Amir Javed, Omer Rana, Pete Burnap, Samuelson T. Atiba, and Soumaya Cherkaoui. "Chatty Devices" and edge-based activity classification. *Discover Internet of Things*, 1(1):5, December 2021.

[45] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2938–2948. PMLR, 26–28 Aug 2020.

[46] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. Federated

learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

[47] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A Benchmark for Federated Settings. *arXiv:1812.01097 [cs, stat]*, December 2019. arXiv: 1812.01097.

[48] Wenqi Shi, Sheng Zhou, and Zhisheng Niu. Device Scheduling with Fast Convergence for Wireless Federated Learning. *arXiv:1911.00856 [cs, math]*, November 2019.

[49] Luping Wang, Wei Wang, and Bo Li. CMFL: Mitigating Communication Overhead for Federated Learning. In *IEEE ICDCS 2019*, pages 954–964, Dallas, TX, USA, July 2019.

[50] Qunsong Zeng, Yuqing Du, Kin K. Leung, and Kaibin Huang. Energy-Efficient Radio Resource Allocation for Federated Edge Learning. *arXiv:1907.06040 [cs, math]*, July 2019.

[51] Yunlong Lu, Xiaohong Huang, Yueyue Dai, Sabita Maharjan, and Yan Zhang. Blockchain and federated learning for privacy-preserved data sharing in industrial iot. *IEEE Transactions on Industrial Informatics*, 16(6):4177–4186, 2020.

[52] Linghe Kong, Xiao-Yang Liu, Hao Sheng, Peng Zeng, and Guihai Chen. Federated tensor mining for secure industrial internet of things. *IEEE Transactions on Industrial Informatics*, 16(3):2144–2153, 2020.

[53] Yunlong Lu, Xiaohong Huang, Ke Zhang, Sabita Maharjan, and Yan Zhang. Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles. *IEEE Transactions on Vehicular Technology*, 69(4):4298–4311, 2020.

[54] Yunlong Lu, Xiaohong Huang, Ke Zhang, Sabita Maharjan, and Yan Zhang. Blockchain Empowered Asynchronous Federated Learning for Secure Data Sharing in Internet of Vehicles. *IEEE Transactions on Vehicular Technology*, 69(4):4298–4311, April 2020.

[55] Zhengxin Yu, Jia Hu, Geyong Min, Haochuan Lu, Zhiwei Zhao, Haozhe Wang, and Nektarios Georgalas. Federated learning based proactive content caching in edge computing. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2018.

[56] Kaiqiang Qi and Chenyang Yang. Popularity prediction with federated learning for proactive caching at wireless edge. In *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2020.

[57] Health Insurance Portability and Accountability Act of 1996 (HIPAA) | CDC, February 2019.

[58] Boyi Liu, Bingjie Yan, Yize Zhou, Yifan Yang, and Yixian Zhang. Experiments of federated learning for covid-19 chest x-ray images, 2020.

[59] Rajesh Kumar, Abdullah Aman Khan, Jay Kumar, Zakria, Noorbakhsh Amiri Golilarz, Simin Zhang, Yang Ting, Chengyu Zheng, and Wenyong Wang. Blockchain-federated-learning and deep learning models for covid-19 detection using ct imaging. *IEEE Sensors Journal*, 21(14):16301–16314, 2021.

[60] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[61] Hui Cao, Shubo Liu, Renfang Zhao, and Xingxing Xiong. IFed: A novel federated learning framework for local differential privacy in Power Internet of Things. *International Journal of Distributed Sensor Networks*, 16(5):1550147720919698, May 2020. Publisher: SAGE Publications.

[62] Anh Nguyen, Tuong Do, Minh Tran, Binh X. Nguyen, Chien Duong, Tu Phan, Erman Tjiputra, and Quang D. Tran. Deep federated learning for autonomous driving, 2021.

[63] Ahmet M. Elbir, Burak Soner, and Sinem Coleri. Federated Learning in Vehicular Networks. *arXiv:2006.01412 [cs, eess, math]*, September 2020. arXiv: 2006.01412.

[64] Yang Liu, Anbu Huang, Yun Luo, He Huang, Youzhi Liu, Yuanyuan Chen, Lican Feng, Tianjian Chen, Han Yu, and Qiang Yang. FedVision: An Online Visual Object Detection Platform Powered by Federated Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(08):13172–13179, April 2020. Number: 08.

[65] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated Learning with Non-IID Data. *arXiv:1806.00582 [cs, stat]*, June 2018.

[66] Naoya Yoshida, Takayuki Nishio, Masahiro Morikura, Koji Yamamoto, and Ryo Yonetani. Hybrid-FL for Wireless Networks: Cooperative Learning Mechanism Using Non-IID Data. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–7, June 2020. ISSN: 1938-1883.

[67] Takayuki Nishio and Ryo Yonetani. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. *ICC 2019*, pages 1–7, May 2019.

[68] Tiansheng Huang et al. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Transactions on Parallel and Distributed Systems*, 32(7):1552–1564, 2021.

[69] Howard H. Yang, Ahmed Arafa, Tony Q. S. Quek, and H. Vincent Poor. Age-Based Scheduling Policy for Federated Learning in Mobile Edge Networks. In *ICASSP 2020*, May 2020.

[70] Howard H. Yang, Zuozhu Liu, Tony Q. S. Quek, and H. Vincent Poor. Scheduling Policies for Federated Learning in Wireless Networks. *arXiv:1908.06287 [cs, eess, math]*, October 2019. arXiv: 1908.06287.

[71] Kevin Hsieh, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R. Ganger, Phillip B. Gibbons, and Onur Mutlu. Gaia: {Geo-Distributed} Machine Learning Approaching {LAN} Speeds. In *Usenix*, pages 629–647, 2017.

[72] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair Resource Allocation in Federated Learning. *arXiv:1905.10497 [cs, stat]*, February 2020.

[73] Kavya Kopparapu, Eric Lin, and Jessica Zhao. Fedcd: Improving performance in non-iid federated learning, 2020.

[74] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. *arXiv:1602.05629 [cs]*, 2016.

[75] Gan Sun, Yang Cong, Jiahua Dong, Qiang Wang, Lingjuan Lyu, and Ji Liu. Data poisoning attacks on federated machine learning. *IEEE Internet of Things Journal*, pages 1–1, 2021.

[76] Ronald Doku and Danda B. Rawat. Mitigating Data Poisoning Attacks On a Federated Learning-Edge Computing Network. In *2021 IEEE 18th*

*Annual Consumer Communications Networking Conference (CCNC)*, pages 1–6, January 2021. ISSN: 2331-9860.

[77] Christopher Briggs, Zhong Fan, and Peter Andras. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, July 2020. ISSN: 2161-4407.

[78] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2020.

[79] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of Personalization Techniques for Federated Learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 794–797, July 2020.

[80] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning, 2020.

[81] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated Evaluation of On-device Personalization. *arXiv:1910.10252 [cs, stat]*, October 2019. arXiv: 1910.10252.

[82] Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication, 2019.

[83] William Shakespeare. *The Complete Works of William Shakespeare.* Project Gutenberg, January 1994.

[84] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998. Conference Name: Proceedings of the IEEE.

[85] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv:1708.07747 [cs, stat]*, September 2017. arXiv: 1708.07747.

[86] CIFAR-10 and CIFAR-100 datasets.

[87] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J. Dally. Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training. *arXiv:1712.01887 [cs, stat]*, June 2020. arXiv: 1712.01887.

[88] Yangguang Cui, Kun Cao, Guitao Cao, Meikang Qiu, and Tongquan Wei. Client scheduling and resource management for efficient training in heterogeneous iot-edge federated learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1, 2021.

[89] Boning Li, Ananthram Swami, and Santiago Segarra. Power allocation for wireless federated learning using graph neural networks, 2021.

[90] Yuxuan Sun, Sheng Zhou, and Deniz Gündüz. Energy-Aware Analog Aggregation for Federated Learning with Redundant Data. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–7, June 2020. ISSN: 1938-1883.

[91] Cong Wang, Xin Wei, and Pengzhan Zhou. Optimize Scheduling of Federated Learning on Battery-powered Mobile Devices. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 212–221, May 2020. ISSN: 1530-2075.

[92] Zhaohui Yang et al. Energy efficient federated learning over wireless communication networks. *IEEE Transactions on Wireless Communications*, 20(3):1935–1949, 2021.

[93] Jie Xu and Heqiang Wang. Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective. *IEEE Transactions on Wireless Communications*, 20(2):1188–1200, 2021.

[94] Amirhossein Reisizadeh, Isidoros Tziotis, Hamed Hassani, Aryan Mokhtari, and Ramtin Pedarsani. Straggler-resilient federated learning: Leveraging the interplay between statistical accuracy and system heterogeneity, 2020.

[95] Wenqi Shi, Sheng Zhou, Zhisheng Niu, Miao Jiang, and Lu Geng. Joint device scheduling and resource allocation for latency constrained wireless federated learning. *IEEE Transactions on Wireless Communications*, 20(1):453–467, 2021.

[96] Yangguang Cui, Kun Cao, Guitao Cao, Meikang Qiu, and Tongquan Wei. Client Scheduling and Resource Management for Efficient Training

in Heterogeneous IoT-Edge Federated Learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1, 2021.

[97] Van-Dinh Nguyen, Shree Krishna Sharma, Thang X. Vu, Symeon Chatzinotas, and Björn Ottersten. Efficient Federated Learning Algorithm for Resource Allocation in Wireless IoT Networks. *IEEE Internet of Things Journal*, 8(5):3394–3409, March 2021. Conference Name: IEEE Internet of Things Journal.

[98] Chit Wutyee Zaw, Shashi Raj Pandey, Kitae Kim, and Choong Seon Hong. Energy-aware resource management for federated learning in multi-access edge computing systems. *IEEE Access*, 9:34938–34950, 2021.

[99] Xiaopeng Mo and Jie Xu. Energy-Efficient Federated Edge Learning with Joint Communication and Computation Design. *Journal of Communications and Information Networks*, 6(2):110–124, June 2021. Conference Name: Journal of Communications and Information Networks.

[100] Wenyuan Xu, Weiwei Fang, Yi Ding, Meixia Zou, and Naixue Xiong. Accelerating Federated Learning for IoT in Big Data Analytics With Pruning, Quantization and Selective Updating. *IEEE Access*, 9:38457–38466, 2021. Conference Name: IEEE Access.

[101] Zirui Xu, Zhao Yang, Jinjun Xiong, Janlei Yang, and Xiang Chen. ELFISH: Resource-Aware Federated Learning on Heterogeneous Edge Devices. *Proceedings of Machine Learning Research*, 97:634–643, 2019.

[102] Nir Shlezinger, Mingzhe Chen, Yonina C. Eldar, H. Vincent Poor, and Shuguang Cui. Federated Learning with Quantization Constraints. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8851–8855, May 2020. ISSN: 2379-190X.

[103] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Vladimir Braverman, Joseph Gonzalez, Ion Stoica, and Raman Arora. COMMUNICATION-EFFICIENT FEDERATED LEARNING WITH SKETCHING. *Journal of Machine Learning Research*, page 28, 2020.

[104] Zaoxing Liu, Tian Li, Virginia Smith, and Vyas Sekar. Enhancing the Privacy of Federated Learning with Sketching. *arXiv:1911.01812 [cs, stat]*, November 2019. arXiv: 1911.01812.

[105] Liang Li, Dian Shi, Ronghui Hou, Hui Li, Miao Pan, and Zhu Han. To Talk or to Work: Flexible Communication Compression for Energy Efficient Federated Learning over Heterogeneous Mobile Edge Devices. In *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pages 1–10, May 2021. ISSN: 2641-9874.

[106] Zhenguo Ma, Yang Xu, Hongli Xu, Zeyu Meng, Liusheng Huang, and Yinxing Xue. Adaptive batch size for federated learning in resource-constrained edge computing. *IEEE Transactions on Mobile Computing*, pages 1–1, 2021.

[107] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous Federated Optimization. *arXiv:1903.03934 [cs]*, September 2019.

[108] Guomei Shi, Li Li, Jun Wang, Wenyan Chen, Kejiang Ye, and ChengZhong Xu. Hysync: Hybrid federated learning with effective synchronization. In *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 628–633, 2020.

[109] Jiangshan Hao, Yanchao Zhao, and Jiale Zhang. Time efficient federated learning with semi-asynchronous communication. In *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 156–163, 2020.

[110] Dongdong Ye, Rong Yu, Miao Pan, and Zhu Han. Federated Learning in Vehicular Edge Computing: A Selective Model Aggregation Approach. *IEEE Access*, 8:23920–23935, 2020. Conference Name: IEEE Access.

[111] Huizi Xiao, Jun Zhao, Qingqi Pei, Jie Feng, Lei Liu, and Weisong Shi. Vehicle Selection and Resource Optimization for Federated Learning in Vehicular Edge Computing. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–15, 2021. Conference Name: IEEE Transactions on Intelligent Transportation Systems.

[112] Shiva Raj Pokhrel and Jinho Choi. A decentralized federated learning approach for connected autonomous vehicles. In *2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6, 2020.

[113] Reza Shokri and Vitaly Shmatikov. Privacy-Preserving Deep Learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*, pages 1310–1321, Denver, Colorado, USA, 2015. ACM Press.

[114] Amine Abouaomar, Soumaya Cherkaoui, Zoubeir Mlika, and Abdellatif Kobbane. Resource Provisioning in Edge Computing for Latency-Sensitive Applications. *IEEE Internet of Things Journal*, 8(14):11088–11099, July 2021. Conference Name: IEEE Internet of Things Journal.

[115] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farokhi Farhad, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. Federated Learning with Differential Privacy: Algorithms and Performance Analysis. *arXiv:1911.00222 [cs]*, November 2019.

[116] Jingjing Wang, Chunxiao Jiang, Haijun Zhang, Yong Ren, Kwang-Cheng Chen, and Lajos Hanzo. Thirty Years of Machine Learning: The Road to Pareto-Optimal Wireless Networks. *IEEE Communications Surveys Tutorials*, pages 1–1, 2020.

[117] Zhiqiang Gong, Ping Zhong, and Weidong Hu. Diversity in Machine Learning. *IEEE Access*, 7:64323–64350, 2019. Conference Name: IEEE Access.

[118] Sadman Sakib, Mostafa M. Fouda, Zubair Md. Fadlullah, and Nidal Nasser. Migrating Intelligence from Cloud to Ultra-Edge Smart IoT Sensor Based on Deep Learning: An Arrhythmia Monitoring Use-Case. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 595–600, June 2020.

[119] Muhammad Habib ur Rehman, Chee Sun Liew, Teh Ying Wah, Muhammad Imran, Khaled Salah, Nidal Nasser, and Davor Svetinovic. Device-centric adaptive data stream management and offloading for analytics applications in future internet architectures. *Future Generation Computer Systems*, 114:155–168, January 2021.

[120] G. Zhu, Y. Wang, and K. Huang. Broadband Analog Aggregation for Low-Latency Federated Edge Learning. *IEEE Transactions on Wireless Communications*, 19(1):491–506, January 2020. Conference Name: IEEE Transactions on Wireless Communications.

[121] Nguyen H. Tran, Wei Bao, Albert Zomaya, Minh N. H. Nguyen, and Choong Seon Hong. Federated Learning over Wireless Networks: Opti-

mization Model Design and Analysis. In *IEEE INFOCOM 2019*, pages 1387–1395, April 2019.

[122] Kai Yang, Tao Jiang, Yuanming Shi, and Zhi Ding. Federated Learning via Over-the-Air Computation. *IEEE Transactions on Wireless Communications*, 19(3):2022–2035, March 2020. Conference Name: IEEE Transactions on Wireless Communications.

[123] Lei Shi and Yi-Dong Shen. Diversifying convex transductive experimental design for active learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 1997–2003, New York, New York, USA, July 2016. AAAI Press.

[124] X. You, R. Wang, and D. Tao. Diverse Expected Gradient Active Learning for Relative Attributes. *IEEE Transactions on Image Processing*, 23(7):3203–3217, July 2014. Conference Name: IEEE Transactions on Image Processing.

[125] Afaf Taïk and Soumaya Cherkaoui. Federated Edge Learning : Design Issues and Challenges. *arXiv:2009.00081 [cs]*, August 2020. arXiv: 2009.00081.

[126] Sadman Sakib, Mostafa Fouda, Zubair Fadullah, and N. Nasser. On covid-19 prediction using asynchronous federated learning-based agile radiograph screening booths. In *Accepted in IEEE International Conference on Communications (ICC)*, 2021.

[127] Sadman Sakib, Mostafa M. Fouda, Zubair Md. Fadlullah, Nidal Nasser, and Waleed Alasmary. A Proof-of-Concept of Ultra-Edge Smart IoT Sensor: A Continuous and Lightweight Arrhythmia Monitoring Approach. *IEEE Access*, 9:26093–26106, 2021. IEEE Access.

[128] Cheng Zhang, Hedvig Kjellstrom, and Stephan Mandt. Determinantal Point Processes for Mini-Batch Diversification. *arXiv:1705.00607 [cs, stat]*, August 2017. arXiv: 1705.00607.

[129] Yi Yang, Zhigang Ma, Feiping Nie, Xiaojun Chang, and Alexander G. Hauptmann. Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization. *International Journal of Computer Vision*, 113(2):113–127, June 2015.

[130] A. Holub, P. Perona, and M. C. Burl. Entropy-based active learning for object recognition. In *2008 IEEE Computer Society Conference on*

*Computer Vision and Pattern Recognition Workshops*, pages 1–8, June 2008. ISSN: 2160-7508.

[131] Lou Jost. Entropy and diversity. *Oikos*, 113(2):363–375, 2006.

[132] Alfredo Nespoli, Emanuele Ogliari, Silvia Pretto, Michele Gavazzeni, Sonia Vigani, and Franco Paccanelli. Data quality analysis in day-ahead load forecast by means of LSTM. In *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I CPS Europe)*, pages 1–5, June 2020.

[133] Clemens Arbesser, Florian Spechtenhauser, Thomas Mühlbacher, and Harald Piringer. Visplause: Visual Data Quality Assessment of Many Time Series Using Plausibility Checks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):641–650, January 2017. Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[134] Afaf Taïk and Soumaya Cherkaoui. Electrical Load Forecasting Using Edge Computing and Federated Learning. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, June 2020. ISSN: 1938-1883.

[135] Alfonso Delgado-Bonal and Alexander Marshak. Approximate Entropy and Sample Entropy: A Comprehensive Tutorial. *Entropy*, 21(6):541, June 2019. Number: 6 Publisher: Multidisciplinary Digital Publishing Institute.

[136] Jun Jiang, Pengjian Shang, Zuoquan Zhang, and Xuemei Li. Permutation entropy analysis based on Gini–Simpson index for financial time series. *Physica A: Statistical Mechanics and its Applications*, 486:273–283, November 2017.

[137] S. Monem Khorasani, G. A. Hodtani, and M. Molavi Kakhki. Investigation and comparison of ECG signal sparsity and features variations due to pre-processing steps. *Biomedical Signal Processing and Control*, 49:87–95, March 2019.

[138] Dongzhu Liu, Guangxu Zhu, Jun Zhang, and Kaibin Huang. Data-Importance Aware User Scheduling for Communication-Efficient Edge Machine Learning. *arXiv:1910.02214 [cs, math]*, October 2019. arXiv: 1910.02214.

[139] Peter M. Catt. Forecastability: Insights from Physics, Graphical Decomposition, and Information Theory. *Foresight: The International Journal of Applied Forecasting*, pages 24–33, 2009. Publisher: International Institute of Forecasters.

[140] Luis Montesinos, Rossana Castaldo, and Leandro Pecchia. On the use of approximate entropy and sample entropy with centre of pressure time-series. *Journal of NeuroEngineering and Rehabilitation*, 15(1):116, December 2018.

[141] PyTorch [online] https://www.pytorch.org. Accessed : 2020-10-26.

[142] Optimization and root finding (scipy.optimize) — SciPy v1.5.3 Reference Guide. [Online] https://docs.scipy.org/doc/scipy/reference/optimize.html , Accessed: 2020-10-26.

[143] Neel Guha, Ameet Talwalkar, and Virginia Smith. One-Shot Federated Learning. *arXiv:1902.11175 [cs, stat]*, March 2019. arXiv: 1902.11175.

[144] Afaf Tak and Soumaya Cherkaoui. Federated Edge Learning: Design Issues and Challenges. *IEEE Network*, 35(2):252–258, March 2021.

[145] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards Federated Learning at Scale: System Design. *arXiv:1902.01046 [cs, stat]*, March 2019. arXiv: 1902.01046.

[146] Hajar Moudoud, Soumaya Cherkaoui, and Lyes Khoukhi. An IoT Blockchain Architecture Using Oracles and Smart Contracts: the Use-Case of a Food Supply Chain. In *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–6, September 2019.

[147] Takayuki Nishio and Ryo Yonetani. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. *ICC 2019*, pages 1–7, May 2019.

[148] Qunsong Zeng, Yuqing Du, Kin K. Leung, and Kaibin Huang. Energy-Efficient Radio Resource Allocation for Federated Edge Learning. *arXiv:1907.06040 [cs, math]*, July 2019.

[149] Nguyen H. Tran, Wei Bao, Albert Zomaya, Minh N. H. Nguyen, and Choong Seon Hong. Federated Learning over Wireless Networks: Optimization Model Design and Analysis. In *IEEE INFOCOM 2019*, pages 1387–1395, April 2019.

[150] Jack Goetz, Kshitiz Malik, Duc Bui, Seungwhan Moon, Honglei Liu, and Anuj Kumar. Active Federated Learning. *arXiv:1909.12641 [cs, stat]*, September 2019.

[151] Afaf Taïk, Zoubeir Mlika, and Soumaya Cherkaoui. Data-Aware Device Scheduling for Federated Edge Learning. *IEEE Transactions on Cognitive Communications and Networking*, pages 1–1, 2021.

[152] Muhammad Habib ur Rehman, Khaled Salah, Ernesto Damiani, and Davor Svetinovic. Towards Blockchain-Based Reputation-Aware Federated Learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 183–188, July 2020.

[153] Wenchao Xia, Wanli Wen, Kai-Kit Wong, Tony Q.S. Quek, Jun Zhang, and Hongbo Zhu. Federated-Learning-Based Client Scheduling for Low-Latency Wireless Communications. *IEEE Wireless Communications*, 28(2):32–38, April 2021.

[154] Xiang Ma, Haijian Sun, and Rose Qingyang Hu. Scheduling Policy and Power Allocation for Federated Learning in NOMA Based MEC. *arXiv:2006.13044 [cs, eess, stat]*, June 2020.

[155] Wenchao Xia, Wanli Wen, Kai-Kit Wong, Tony Q.S. Quek, Jun Zhang, and Hongbo Zhu. Federated-Learning-Based Client Scheduling for Low-Latency Wireless Communications. *IEEE Wireless Communications*, 28(2):32–38, April 2021. IEEE Wireless Communications.

[156] Muhammad Shayan, Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. Biscotti: A Ledger for Private and Secure Peer-to-Peer Machine Learning. *arXiv:1811.09904 [cs, stat]*, December 2019. arXiv: 1811.09904.

[157] Yang Zhao, Jun Zhao, Linshan Jiang, Rui Tan, Dusit Niyato, Zengxiang Li, Lingjuan Lyu, and Yingbo Liu. Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices. *arXiv:1906.10893 [cs, eess]*, February 2021. arXiv: 1906.10893.

[158] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang. Incentive Mechanism for Reliable Federated Learning: A Joint Optimization Approach to Combining Reputation and Contract Theory. *IEEE Internet of Things Journal*, 6(6):10700–10714, December 2019. IEEE Internet of Things Journal.

[159] Hajar Moudoud, Lyes Khoukhi, and Soumaya Cherkaoui. Prediction and Detection of FDIA and DDoS Attacks in 5G Enabled IoT. *IEEE Network*, 35(2):194–201, March 2021.

[160] Di Cao, Shan Chang, Zhijian Lin, Guohua Liu, and Donghong Sun. Understanding Distributed Poisoning Attack in Federated Learning. In *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 233–239, December 2019.

[161] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How To Backdoor Federated Learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR, June 2020.

[162] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, July 2009.

[163] Shiqi Shen, Shruti Tople, and Prateek Saxena. Auror: defending against poisoning attacks in collaborative deep learning systems. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, ACSAC '16, pages 508–519, New York, NY, USA, December 2016. Association for Computing Machinery.

[164] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE Access*, 8:58443–58469, 2020. IEEE Access.

[165] Lóránt Szabó, László Lindenmaier, and Viktor Tihanyi. Smartphone Based HD Map Building for Autonomous Vehicles. In *2019 IEEE 17th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 365–370, January 2019.

[166] Seong-Woo Kim and Wei Liu. Cooperative Autonomous Driving: A Mirror Neuron Inspired Intention Awareness and Cooperative Perception Approach. *IEEE Intelligent Transportation Systems Magazine*, 8(3):23–32, 2016.

[167] Bo Yang, Xuelin Cao, Kai Xiong, Chau Yuen, Yong Liang Guan, Supeng Leng, Lijun Qian, and Zhu Han. Edge Intelligence for Autonomous Driving in 6G Wireless System: Design Challenges and Solutions. *IEEE Wireless Communications*, 28(2):40–47, April 2021. IEEE Wireless Communications.

[168] Tengchan Zeng, Omid Semiari, Mingzhe Chen, Walid Saad, and Mehdi Bennis. Federated Learning on the Road: Autonomous Controller Design for Connected and Autonomous Vehicles. *arXiv:2102.03401 [cs, eess]*, February 2021. arXiv: 2102.03401.

[169] Zhengxin Yu, Jia Hu, Geyong Min, Zhiwei Zhao, Wang Miao, and M. Shamim Hossain. Mobility-Aware Proactive Edge Caching for Connected Vehicles Using Federated Learning. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–11, 2020.

[170] Yuanshao Zhu, Shuyu Zhang, Yi Liu, Dusit Niyato, and James J.Q. Yu. Robust federated learning approach for travel mode identification from non-iid gps trajectories. In *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 585–592, 2020.

[171] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated Learning in Mobile Edge Networks: A Comprehensive Survey. *IEEE Communications Surveys Tutorials*, 22(3):2031–2063, 2020. IEEE Communications Surveys Tutorials.

[172] Mohammed Aledhari, Rehma Razzak, Reza M. Parizi, and Fahad Saeed. Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Access*, 8:140699–140725, 2020. IEEE Access.

[173] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M. Hadi Amini. A Survey on Federated Learning for Resource-Constrained IoT Devices. *IEEE Internet of Things Journal*, pages 1–1, 2021. IEEE Internet of Things Journal.

[174] Issam Jabri, Tesnim Mekki, Abderrezak Rachedi, and Maher Ben Jemaa. Vehicular fog gateways selection on the internet of vehicles: A fuzzy logic with ant colony optimization based approach. *Ad Hoc Networks*, 91:101879, August 2019.

[175] Ismaeel Al Ridhawi, Moayad Aloqaily, Burak Kantarci, Yaser Jararweh, and Hussein T. Mouftah. A continuous diversified vehicular cloud service availability framework for smart cities. *Computer Networks*, 145:207–218, November 2018.

[176] Irina Tal and Gabriel-Miro Muntean. Towards Smarter Cities and Roads: A Survey of Clustering Algorithms in VANETs, 2014. ISBN: 9781466659780 Pages: 16-50 Publisher: IGI Global.

[177] Craig Cooper, Daniel Franklin, Montserrat Ros, Farzad Safaei, and Mehran Abolhasan. A Comparative Survey of VANET Clustering Techniques. *IEEE Communications Surveys Tutorials*, 19(1):657–681, 2017. IEEE Communications Surveys Tutorials.

[178] Dinesh Singh, Ranvijay, and R. S. Yadav. NWCA: A New Weighted Clustering Algorithm to form Stable Cluster in VANET. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, ICTCS '16, pages 1–6, New York, NY, USA, March 2016. Association for Computing Machinery.

[179] Ameneh Daeinabi, Akbar Ghaffar Pour Rahbar, and Ahmad Khademzadeh. VWCA: An efficient clustering algorithm in vehicular ad hoc networks. *Journal of Network and Computer Applications*, 34(1):207–222, January 2011.

[180] Yeongwoo Kim, Ezeddin Al Hakim, Johan Haraldson, Henrik Eriksson, José Mairton B. da Silva Jr., and Carlo Fischione. Dynamic Clustering in Federated Learning. *arXiv:2012.03788 [cs]*, December 2020. arXiv: 2012.03788.

[181] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An Efficient Framework for Clustered Federated Learning. *arXiv:2006.04088 [cs, stat]*, June 2021. arXiv: 2006.04088.

[182] Peter Kairouz et al. Advances and Open Problems in Federated Learning. *arXiv:1912.04977 [cs, stat]*, March 2021. arXiv: 1912.04977.

[183] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three Approaches for Personalization with Applications to Federated Learning. *arXiv:2002.10619 [cs, stat]*, July 2020. arXiv: 2002.10619.

[184] Zheyi Chen, Pu Tian, Weixian Liao, and Wei Yu. Zero Knowledge Clustering Based Adversarial Mitigation in Heterogeneous Federated

Learning. *IEEE Transactions on Network Science and Engineering*, 8(2):1070–1083, April 2021. IEEE Transactions on Network Science and Engineering.

[185] Afaf Taïk, Hajar Moudoud, and Soumaya Cherkaoui. Data-Quality Based Scheduling for Federated Edge Learning. In *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, pages 17–23, October 2021. ISSN: 0742-1303.

[186] Lumin Liu, Jun Zhang, S.H. Song, and Khaled B. Letaief. Client-Edge-Cloud Hierarchical Federated Learning. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, June 2020. ISSN: 1938-1883.

[187] Haoye Chai, Supeng Leng, Yijin Chen, and Ke Zhang. A Hierarchical Blockchain-Enabled Federated Learning Algorithm for Knowledge Sharing in Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):3975–3986, July 2021. IEEE Transactions on Intelligent Transportation Systems.

[188] Mengying Ren, Jun Zhang, Lyes Khoukhi, Houda Labiod, and Véronique Vèque. A Unified Framework of Clustering Approach in Vehicular Ad Hoc Networks. *IEEE Transactions on Intelligent Transportation Systems*, 19(5):1401–1414, May 2018. IEEE Transactions on Intelligent Transportation Systems.

[189] Weiwei Li, Ali Tizghadam, and Alberto Leon-Garcia. Robust clustering for connected vehicles using local network criticality. In *2012 IEEE International Conference on Communications (ICC)*, pages 7157–7161, June 2012. ISSN: 1938-1883.

[190] Mengying Ren, Lyes Khoukhi, Houda Labiod, Jun Zhang, and Véronique Vèque. A mobility-based scheme for dynamic clustering in vehicular ad-hoc networks (VANETs). *Vehicular Communications*, 9:233–241, 2017.

[191] David Pisinger. Where are the hard knapsack problems? *Computers & Operations Research*, 32(9):2271–2284, September 2005.

[192] Cheng Chen, Sean Chester, Venkatesh Srinivasan, Kui Wu, and Alex Thomo. Group-Aware Weighted Bipartite B-Matching. In *Proceedings of the 25th ACM International on Conference on Information and*

*Knowledge Management*, CIKM '16, pages 459–468, New York, NY, USA, October 2016. Association for Computing Machinery.

[193] Faez Ahmed, John P. Dickerson, and Mark Fuge. Diverse weighted bipartite b-matching. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Aug 2017.

[194] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data Poisoning Attacks Against Federated Learning Systems. *arXiv:2007.08432 [cs, stat]*, August 2020. arXiv: 2007.08432.

[195] Le Liang, Geoffrey Ye Li, and Wei Xu. Resource Allocation for D2D-Enabled Vehicular Communications. *IEEE Transactions on Communications*, 65(7):3186–3197, July 2017. note: IEEE Transactions on Communications.

[196] Mubashir Husain Rehmani, Martin Reisslein, Abderrezak Rachedi, Melike Erol-Kantarci, and Milena Radenkovic. Integrating renewable energy resources into the smart grid: Recent developments in information and communication technologies. *IEEE Transactions on Industrial Informatics*, 14(7):2814–2825, 2018.

[197] Moayad Aloqaily, Azzedine Boukerche, Ouns Bouachir, Fariea Khalid, and Sobia Jangsher. An energy trade framework using smart contracts: Overview and challenges. *IEEE Network*, 34(4):119–125, 2020.

[198] GSMA. Smart Grid 5G Network Slicing. `www.gsma.com/futurenetworks/wiki/smart-grid-5g-network-slicing/`, Mar 2020. Accessed : 03-02-2021.

[199] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[200] Afaf Tak and Soumaya Cherkaoui. Federated Edge Learning: Design Issues and Challenges. *IEEE Network*, 35(2):252–258, 2021.

[201] Afaf Taïk and Soumaya Cherkaoui. Electrical load forecasting using edge computing and federated learning. In *IEEE International Conference on Communications*, pages 1–6. IEEE, 2020.

[202] Eunice Espe, Vidyasagar Potdar, and Elizabeth Chang. Prosumer communities and relationships in smart grids: A literature review, evolution and future directions. *Energies*, 11(10):2528, 2018.

[203] Yuris Mulya Saputra, Dinh Thai Hoang, Diep N Nguyen, Eryk Dutkiewicz, Markus Dominik Mueck, and Srikathyayani Srikanteswara. Energy demand prediction with federated learning for electric vehicle networks. In *IEEE Global Communications Conference*, pages 1–6. IEEE, 2019.

[204] N. Gensollen, V. Gauthier, M. Becker, and M. Marot. Stability and Performance of Coalitions of Prosumers Through Diversification in the Smart Grid. *IEEE Transactions on Smart Grid*, 9(2):963–970, March 2018. IEEE Transactions on Smart Grid.

[205] A. J. Dinusha Rathnayaka, Vidyasagar M. Potdar, Tharam S. Dillon, Omar K. Hussain, and Elizabeth Chang. A Methodology to Find Influential Prosumers in Prosumer Community Groups. *IEEE Transactions on Industrial Informatics*, 10(1):706–713, 2014.

[206] Faizan Safdar Ali, Ouns Bouachir, Öznur Özkasap, and Moayad Aloqaily. SynergyChain: Blockchain-Assisted Adaptive Cyber-Physical P2P Energy Trading. *IEEE Transactions on Industrial Informatics*, 17(8):5769–5778, August 2021.

[207] A. Triantafyllou, J. A. P. Jimenez, A. D. R. Torres, T. Lagkas, K. Rantos, and P. Sarigiannidis. The Challenges of Privacy and Access Control as Key Perspectives for the Future Electric Smart Grid. *IEEE Open Journal of the Communications Society*, 1:1934–1960, 2020.

[208] Kaixuan Chen, Jin Lin, and Yonghua Song. Trading strategy optimization for a prosumer in continuous double auction-based peer-to-peer market: A prediction-integration model. *Applied Energy*, 242:1121–1133, May 2019.

[209] Jiechen Wu, Junjie Hu, Xin Ai, Zhan Zhang, and Huanyu Hu. Multi-time scale energy management of electric vehicle model-based prosumers by using virtual battery model. *Applied Energy*, 251:113312, October 2019.

[210] Pecan Street Inc. Dataport 2021 [Online] https://dataport.pecanstreet.org/. Accessed : 2021-02-09.