Exploring the Trade-offs between Energy and Performance of Federated Learning Algorithms

Mai Huong Do

July 16, 2024

This report presents my activity as part of my thesis, directed by **Georges Da Costa** and **Millian Poquet**, within the **SEPIA team**.

Contents

1 Introduction

1.1 Thesis context

There is an increasing interest in a new distributed Machine Learning (ML) paradigm called Federated Learning (FL), in which nodes compute their local gradients and communicate them to a central server. This centralized server then orchestrates training rounds over large data volumes created and stored locally at a huge number of nodes. This training procedure repeats until some criteria are met. This enables the participating nodes (e.g., IoT devices, mobile phones, etc) to protect their data and solve the data security and privacy issues imposed by law.

This growing demand for FL technology opens new challenges besides those that appear in traditional ML. These include (i) communications and energy costs between the nodes and the server, (ii) increased duration of the training period due to the correlation between databases, and (iii) unfair distribution of computation and communication between heterogeneous devices. In all these challenges, energy efficiency is totally absent in all new FL architectures such as cross-node FL, cross-Silo FL, federated transfer learning, personalized FL, etc.

The goal of ANR project [1]DELIGHT started early 2023 is to incorporate energy efficiency as one of the metrics of FL to push FL towards sustainability. In the context of this position, we will focus on the first challenge. This project will use the Flower framework for demonstration purposes.

1.2 Objective and Plan

Our team's contribution to the DELIGHT project is expected to be formulating mathematical energy consumption models, reducing energy consumption in hybrid architecture (CPU/GPU), and developing middleware for clusters/mesh/cloud.

The main objective of my PhD is to explore the tradeoffs between energy and performance of FL algorithms, develop new methods for estimating energy, formulate a mathematical energy consumption model, and provide software libraries for precise measurement of the energy consumption of FL at different blocks (local data, hardware, software, and network). For this purpose, we plan to leverage many parameters of the framework, such as the precision of the floating-point computations and how data is transferred between distributed actors. We will focus on the use case of automatic speech recognition or object detection.

Phases of the PhD:

- Set up an experimental environment on Grid'5000 (g5k) to gather performance and energy metrics.
  - Create a use-case for the Flower framework.
  - Build a reproducible and automated framework for obtaining metrics for this use case.
- Propose, model, and implement the different leverages.
- Explore the impact of the leverages on both energy and performance.

1.3 Flower - a FL framework

Searching Google Scholar quickly allows us to locate articles about the shortcomings of conventional ML. Since data is typically not constantly available at the server, sending a lot of local data to the center will incur transmission charges. Furthermore, security problems will occur when restrictions prevent some data from being exchanged.

FL can reverse this approach. It enables ML on distributed data by moving the training to the data instead of moving the data to the training. Simply, instead of having to move data, federated training moves the specialized model to the user and allows users to train themselves on their devices.

The five steps of FL will be presented below:

- Initialize global model: start by initializing the model on the server randomly or from a previously saved checkpoint, which is exactly the same as classic centralized learning.

[1] a**D**vancing f**E**derated **L**earn**I**ng while reducin**G** t**H**e carbon foo**T**print

- Send model to connected organizations/devices (client nodes): the parameters of the global model will be sent to the connected client nodes and ensure that each participating node starts its local training using the same model parameters.
- Train model locally on the data of each organization/device (client node): start the local training with the latest version of the global model parameters.
- Return model updates back to the server: after local training, each client node has a different version of the model parameters and then sends those model updates back to the server.
- Aggregate model updates into a new global model: after receiving model updates from the selected client nodes, the server has to combine all the model updates (aggregation).
- Repeat steps 1 to 4 until the model converges: the four steps above create a single round of FL, and the process has to continue until we get a fully trained model.

Theoretically, FL will be a major improvement on the disadvantages of traditional ML. However, federated evaluation and federated analytics require infrastructure to move ML models back and forth, train and evaluate them on local data, and then aggregate the updated models. Here, the Flower framework provides the infrastructure to do exactly that in an easy, scalable, and secure way. In short, Flower presents a unified approach to FL, analytics, and evaluation. It allows the user to federate any workload, any ML framework, and any programming language.

1.4 Grid'5000

G5k is a large-scale and flexible testbed for experiment-driven research in all areas of computer science, with a focus on parallel and distributed computing including Cloud, HPC, Big Data, and AI.

- Provides access to a large amount of resources:
  - 15,000 cores
  - 800 compute-nodes grouped in homogeneous clusters
  - Features various technologies: PMEM, GPU, SSD, NVMe, 10G and 25G Ethernet, Infiniband, Omni-Path
- Highly reconfigurable and controllable:
  - Researchers can experiment with a fully customized software stack thanks to bare-metal deployment features.
  - Experiment isolation at the networking layer is possible.
- Advanced monitoring and measurement features:
  - Traces collection of networking and power consumption.
  - Provides a deep understanding of experiments.
- Designed to support Open Science and reproducible research:
  - Full traceability of infrastructure and software changes on the testbed.
- Vibrant community:
  - 500+ users supported by a solid technical team.

2 Progress

2.1 Survey of related works

The survey spans from 2019 to 2024, focusing on keywords such as energy consumption, federated learning, trade-offs, and performance. The main goal is to reproduce reusable works, ensuring the research is valuable and applicable in various contexts. Therefore, I prefer to focus on articles that provide a reproducible framework. Currently, the survey includes 31 articles, but this number is expected to increase over time as more relevant research is conducted and added. The continuous increase in articles has the benefit of providing a comprehensive overview of developments but also represents a major challenge for us in the research race.

Conclusion:

- Integrating energy awareness into the design and implementation of FL frameworks will be essential. This involves optimizing not only computational tasks but also communication methods and data analysis processes to minimize energy consumption.
- Most of them did not model the energy consumption of the server (aggregation).
- Time processing is one of the interesting points when evaluating the performance of FL.
- Not many articles care about carbon equivalent yet while it is a practical indicator and directly affects the real world.
- In optimization and resource allocation problems, memory and queue factors are often overlooked while they are an important factor that directly affects packet loss.

Please go to the next pages and take a look at the Table. 1 and Table. 2 to see the survey summary.

2.2 Flower implementation

2.2.1 Preparation

Includes: package requirements and datasets.

```
dependencies = [
    "flwr>=1.8.0,<2.0",
    "flwr-datasets[vision]>=0.0.2,<1.0.0",
    "tensorflow-cpu>=2.9.1, != 2.11.1 ; platform_machine == \"x86_64\""
]
```

```
# Load model and data (MobileNetV2, CIFAR-10)
model = tf.keras.applications.MobileNetV2((32, 32, 3), classes=10, weights=
    None)
model.compile("adam", "sparse_categorical_crossentropy", metrics=["accuracy"])
```

2.2.2 Main components

Includes: client, server, strategy.

```
# Define Flower client
class CifarClient(fl.client.NumPyClient):
    def get_parameters(self, config):
        return model.get_weights()

    def fit(self, parameters, config):
        model.set_weights(parameters)
        model.fit(x_train, y_train, epochs=1, batch_size=32)
        return model.get_weights(), len(x_train), {}

    def evaluate(self, parameters, config):
        model.set_weights(parameters)
```

Article	Year	Energy				FL performance			Etc.			
		Scope of E		Type of Energy		Acc.	Loss	Time processing	Carbon equivalent	Bandwidth	Computing resource	Memory resource
		Client	Server	Communication	Computation							
[TBZ+19b]	2019	x		x	x	x		x			x	
[SZG20]	2020	x		x			x					
[ZLG20]	2020	x		x	x		x	x			x	
[LWCX20]	2020	x		x	x			x			x	x
[JPC+20]	2020	x		x	x	x						
[SKRB21b]	2021	x	x	x	x		x		x			
[KW21]	2021	x		x	x	x				x	x	x
[LLW+21]	2021	x		x	x		x	x				
[YCS+21]	2021	x		x	x	x		x		x	x	
[ZPKH21]	2021	x		x	x		x	x		x	x	
[MX21b]	2021	x		x	x	x		x			x	
[DTN+21b]	2021	x		x	x	x		x			x	
[LSH+21]	2021	x		x	x					x	x	
[LZP21]	2021	x			x		x	x				x
[CYS+21b]	2022	x		x	x		x	x		x	x	
[JHD22]	2022			x	x	x						
[HCS+22]	2022	x		x	x		x					
[CLC+22]	2022	x		x	x		x	x		x	x	
[ZLWG22]	2022	x		x	x		x	x			x	
[AA22]	2022	x		x	x							
[YLS+24]	2023	x	x	x	x	x						
[SRKB23b]	2023	x	x	x	x				x			
[CLX+22]	2023	x		x	x					x	x	
[RYDF+23]	2023	x	x	x	x			x			x	
[CYLN23]	2023	x		x	x		x	x		x		
[BDLMB23]	2023	x	x	x	x		x	x		x	x	
[ASCF23b]	2023						x	x				
[Pil23b]	2023	x		x	x						x	
[XLL+24]	2024	x	x	x	x					x	x	
[KSMD24b]	2024	x		x	x	x		x				
[ZWX+24]	2024	x		x	x	x		x		x	x	

Table 1: Related works: classify by objectives

Article	**Proposal**	**Dataset**	**Language**	**Remark**
[TBZ+19b]	Non-convex → decomposing sub problems	MNIST	Python	[TBZ+19a]
[SZG20]	Lyapunov optimization			None
[ZLG20]	- use DRL, action based on experience - near-optimal solution			None
[LWCX20]	- propose MCFL, a multi-layer coordination framework - balances the training completion time, model accuracy and energy efficiency			None
[JPC+20]	adaptive client-selection algorithm			None
[SKRB21b]	framework for the analysis of energy and carbon footprints	MNIST FL radar	Python	[SKRB21a]
[KW21]	DRL (state, action, reward max is -E), update weight each round until convergence			None
[LLW+21]	low-cost sampling-based algorithm to learn the convergence related unknown parameters			None
[YCS+21]	- Subproblem: FL performance and resources - Iterative algorithm to find a feasible solution			None
[ZPKH21]	bandwidth & data offload→ Generalized Nash Equilibrium Problem			None
[MX21b]	transfer to convex → use Lagrange duality	MNIST	Matlab	[MX21a]
[DTN+21b]	Non-convex → decomposing sub problems	MNIST FENIST Synthetic	Python	[DTN+21a]
[LSH+21]	Flexible Topk Local Stochastic Gradient Descent with Dynamic Batch sizes (FT-LSGD-DB) algorithm & Compression Control Algorithm			None
[LZP21]	Joint Optimization Algorithm : - Local batch size - User scheduling - Combine 2 above			None
[CYS+21b]	closed-form expression for the expected convergence rate → the optimal power → optimize user selection and loss	MNIST	Matlab	[CYS+21a]
[JHD22]	stochastic sign-based algorithm Optimize by SignSGD Sub-problem: learning perf (with fix E) and energy requirement (with fix learn perf)			None
[HCS+22]	- Convergence rate: how to transmit power, the number of scheduled users and user associations affect the training loss - decompose the problem to sub-problems and solve			None
[CLC+22]	Iterative algorithm based on Lyapunov optimization: select client, allocate B (choosing different numbers of clients in different learning stages, later better)			None
[ZLWG22]	Use DRL, action based on experience (system condition → action with best reward) - near-optimal solution			Delete
[AA22]	Power-aware design improves the FL performance in battery-powered scenarios, reduced client dropouts and increased participation levels			None
[YLS+24]	Game theory, NASH equilibrium			None
[SRKB23b]	Framework for the analysis of energy and carbon footprints in distributed and FL	MNIST CIFAR	Python	[SRKB23a]
[CLX+22]	Quantized weight → obtained theoretical convergence bound → Efficient iterative algorithm for bandwidth allocation & weight quantization level			None
[RYDF+23]	Sub-problems: the computation & communication tasks Offline Greedy Energy-Efficient Client Scheduling Scheme → Online			None
[CYLN23]	Transfer to Lyapunov optimization Derive optimal bandwidth and power policies by convex optimization Polynomial-time algorithm → device scheduling			None
[BDLMB23]	Lyapunov stochastic optimization → customize for LMS estimation & training Deep CNN			None
[ASCF23b]	Intelligent Participant Selection (IPS): improve resource diversity Staleness-Aware Aggregation (SAA): improve resource efficiency	CIFAR10, OpenImage StackOverflow, Reddit, GG speech	Python	[ASCF23a]
[Pil23b]	Pseudo-polynomial optimal solution Multiple-Choice Minimum-Cost Maximal Knapsack Packing Problem 4 algorithms for scenarios	generate	Python	[Pil23a]
[XLL+24]	Online Learning Algorithm for the UE Selection Based on Bandits With Correlated Contexts An Approximation Algorithm With Selected UEs Online learning with bounded regret for Multi FL requests			None
[KSMD24b]	Pareto boundary for the convergence rate Nash bargaining solution and analyzing the derived convergence rate	MNIST	Python	[KSMD24a]
[ZWX+24]	Decomposition to 3 sub-problems Overall alternating optimization algorithm			None

Table 2: Related works: proposals and open sources

```python
        loss, accuracy = model.evaluate(x_test, y_test)
        return loss, len(x_test), {"accuracy": accuracy}

# Start Flower client
if len(sys.argv) < 2:
    fl.client.start_numpy_client(server_address="127.0.0.1:8080", client=CifarClient())
else:
    fl.client.start_numpy_client(server_address=sys.argv[1], client=CifarClient())
```

```python
# Choose the strategy
strategy = tsc.newStrat()
# Start Flower server
fl.server.start_server(
  server_address="0.0.0.0:8080",
  config=fl.server.ServerConfig(num_rounds=1),
  strategy=strategy
)
```

```python
class newStrat(fl.server.strategy.FedAvg):
    def aggregate_fit(
        self,
        server_round: int,
        results: List[Tuple[ClientProxy, FitRes]],
        failures: List[Union[Tuple[ClientProxy, FitRes], BaseException]],
    ) -> Tuple[Optional[Parameters], Dict[str, Scalar]]:
        """Aggregate fit results using weighted average."""
        if not results:
            return None, {}
        # Do not aggregate if there are failures and failures are not accepted
        if not self.accept_failures and failures:
            return None, {}

        # Convert results
        weights_results = [
            (parameters_to_ndarrays(fit_res.parameters), fit_res.num_examples)
            for _, fit_res in results
        ]
        parameters_aggregated = ndarrays_to_parameters(aggregate_median(weights_results))

        # Aggregate custom metrics if aggregation fn was provided
        metrics_aggregated = {}
        if self.fit_metrics_aggregation_fn:
            fit_metrics = [(res.num_examples, res.metrics) for _, res in results]
            metrics_aggregated = self.fit_metrics_aggregation_fn(fit_metrics)
        elif server_round == 1:  # Only log this warning once
            log(WARNING, "No fit_metrics_aggregation_fn provided")

        np.savez(f"round-{server_round}-weight.npz", parameters_aggregated)
        np.savez(f"round-{server_round}-agg-metrics.npz", metrics_aggregated)

        return parameters_aggregated, metrics_aggregated
```

2.3 Energy consumption measurement

2.3.1 Measurement tools

Using Expetator - A tool for running HPC benchmarks using several types of leverages (DVFS) and low-level monitoring (hardware performance counters, RAPL), mostly on g5k. To install Expetator:

```
    pip3 install expetator
```

```
class DemoBench:

    def __init__(self, params=[1]):
        self.names = {"flower"}
        self.params = params

    def build(self, executor):

        executor.local(f"chmod +x {script_dir}")
        params = {"flower": self.params}
        return params

    def run(self, bench, param, executor):

        output = executor.local(f"{script_dir}")
        return output.strip(), "flower"

if __name__ == "__main__":
    experiment.run_experiment(
        "/tmp/flower", [DemoBench()],
        leverages=[], monitors=[Mojitos(sensor_set={'user','rxp','dram0'})],
            times=1
    )
```

2.3.2 Energy parameters

For CPU statistics, there are eight states are “user” (us), “system” (sy), “nice” (ni), “idle” (id), “iowait” (wa), “hardware interrupt” (hi), “software interrupt” (si), and “steal” (st).

Of these eight, “system”, “user”, and “idle” are the main states the CPU can be in.

- **System (sy)**: The “system” CPU state shows the amount of CPU time used by the kernel. The kernel handles low-level tasks such as interacting with hardware, memory allocation, managing file systems, and running device drivers. It also includes tasks like the CPU scheduler, which determines process access to the CPU.
- **User (us)**: The “user” CPU state shows CPU time used by user-space processes, which are higher-level processes like applications or database servers. Any CPU time used by processes outside the kernel is marked as “user”.
- **Nice (ni)**: The “nice” CPU state is a subset of the “user” state and shows CPU time used by processes with a positive niceness value, indicating lower priority than other tasks.
- **Idle (id)**: The “idle” CPU state shows CPU time that is not actively being used. It is calculated by tasks with the lowest possible priority.
- **Iowait (wa)**: “Iowait” marks time spent waiting for input or output operations, such as reading or writing to disk.
- **Other Statistics**: Besides “nice” in “user” and “iowait” in “idle”, there are more subcategories the main CPU states can be divided into:
  - “Hardware interrupt” (hi or irq) and “software interrupt” (si, or softirq) categories are time spent servicing interrupts.
  - “Steal” (st) marks time spent waiting for a virtual CPU in a virtual machine.

2.4 Experiment on g5k

After working independently and fluently with the Flower framework and Expetator, I will combine them to measure resource usage when running the Flower framework. Then, I also have to deploy them into the g5k system.

The idea is as follows:

- Step 1: Write a script to run the Flower framework automatically in g5k:
  - Create a .json file: information of clients and server host, dataset using in Flower, g5k wall time, etc.
  - Call the .json and extract the needed information (such as server IP address and number of clients, which dataset used,...).
  - Run the server.py in and client.py on the identified hosts.
- Step 2: Make our benchmark for Flower in Expetator.
- Step 3: Collect the data: Flower performance & Energy consumption.

The quick start of g5k can be found in [2]here. The example of testing results are shown on Fig. 1 and Fig. 2.

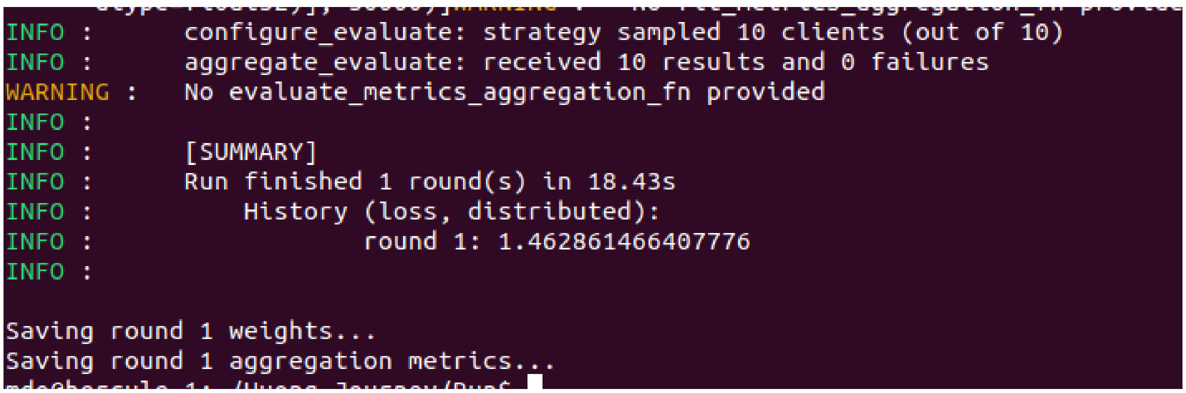

Figure 1: Flower framework results with 10 clients & 1 server

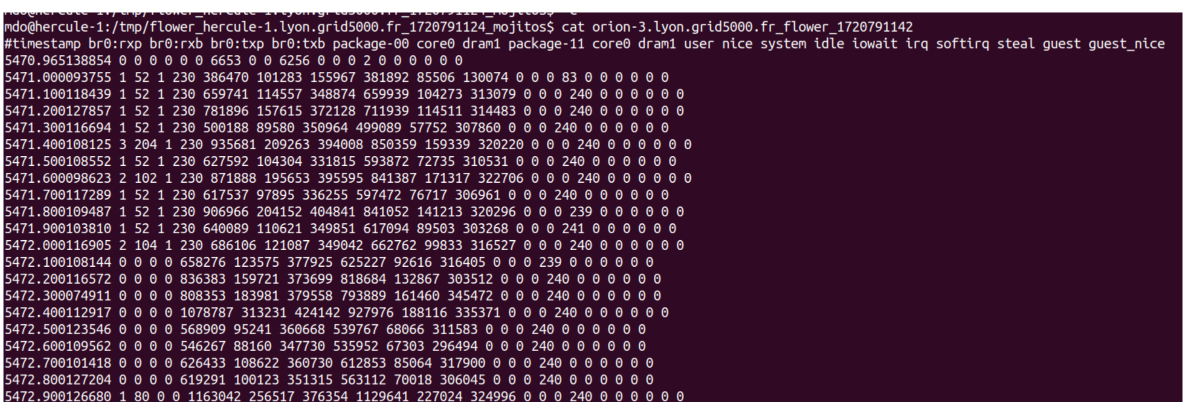

Figure 2: Energy consumption in node **orion-3 (lyon)**, measurement by Expetator

[2]https://synergies.univ-tlse3.fr/service/extension/drive/link/QRE7RL545W2AVUNFYKWA6ZUTBLOCJGM

2.5 DELIGHT meeting

After the DELIGHT project meeting with professors and students from Avignon University/LIA and CNRS LAAS that took place on May 6th - 7th, we had an initial direction for each person's work. We then arranged a meeting on July 5th among the students to discuss the progress of our work. The meeting content is summarized in the following table:

Name	**Team/ Status**	**Done works**	**On going/Open issues**	**Remarks**
Mai H	IRIT/ PhD	- Finish a simple framework combining Flower + Energy measurement - Deploy and test in Grid5000	- Try to change clients per round (toward clients selection) - Stability needs attention. - Need to find a solution to the problem of waiting time and synchronization when running multiple clients.	- Violation when exceed the day/night boundary of Grid'5000 –>Solved
Khaoula	LIA/ Master	- Working on client selection	None	- Not yet evaluating energy consumption - Will leave after 1 month
Ahmad	LIA/ PhD	- Implementing and testing FL schemes in the literature: FedAvg, FedBN, FedDrop, FedPMT and other variants - Suggesting and testing new schemes	None	- Not yet evaluating energy consumption
Oumayma	LAAS/ PhD	- Reducing the size of training data - Study the effect of data size on energy consumption	- How to mesure the energy - Which type of energy should we consider - Violation when exceeding the limit of Grid'5000	- Open issues were solved during the meeting

2.6 Future works

In the next steps, I will continue to build a reproducible and automated framework and focus on improving stability when using Expetator. I also plan to propose, model, and implement the different leverages to explore their impact on energy and performance. I will test not only the current Flower framework but also another FL framework or an updated version of the current Flower. In addition, I plan to read deeply and fully understand other works on energy modeling for FL (which I have surveyed) as a basis for my proposal.

3 Working situation

3.1 Difficulties encountered

Up to now I have not encountered too many difficulties, this is because it has not been too long since I started work and I am still in the process of studying to consolidate my knowledge.

However, in the future I may encounter difficulties because there will come a time when I need to propose my ideas. I need to continue surveying, find out the interesting points of related works, as well as their disadvantages and of course, I will have to brainstorm.

3.2 Publications

I have not had any publications with SEPIA until now. But in the past, I already had one journal and three conference articles. I currently have a manuscript being revised and will submit it soon, but it also belongs to my previous institute (in Korea).

- Do, H. M., Tran, T. P., & Yoo, M. (2023). Deep Reinforcement Learning-Based Task Offloading and Resource Allocation for Industrial IoT in MEC Federation System. *IEEE Access*.
- Do, H. M., & Yoo, M. (2023). Delay Optimization for Augmented Reality Service using Mobile Edge Computing Federation system. In *2023 14th International Conference on Information and Communication Technology Convergence (ICTC)*

- Do, H. M., & Yoo, M. (2023). Energy Consumption Optimization in Mobile Edge Computing Federation based Deep Reinforcement Learning. In *Korean Society of Communication Studies Conference Proceedings*, 1834-1835.
- Do, H. M., & Yoo, M. (2022). Delay Optimization in Mobile Edge Computing Federation using Task Offloading and Resource Allocation. In *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 767-770). IEEE.

3.3 Teaching

I have no previous experience teaching at a university. However, I do plan to teach in the near future, maybe next semester, as long as those classes are in English.

3.4 Training

ASR day - IRIT: I attended the doctoral students' day organized by the ASR department, during which I presented my work, which is equivalent to 6 hours of training for EDMITT.

MOOC 1: I finished MOOC 1 (Reproducible research: methodological principles for transparent science) and am waiting to update my certificate to Adum.fr.

MOOC 2: I am participating in MOOC 2 and I have not finish yet (Reproducible Research II: Practices and tools for managing computations and data).

References

[AA22] Amna Arouj and Ahmed M. Abdelmoniem. Towards energy-aware federated learning on battery-powered clients. In *Proceedings of the 1st ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network*, pages 7–12. ACM, 2022.

[ASCF23a] Ahmed M. Abdelmoniem, Atal Narayan Sahu, Marco Canini, and Suhaib A. Fahmy, 2023. [Accessed 11-07-2024].

[ASCF23b] Ahmed M. Abdelmoniem, Atal Narayan Sahu, Marco Canini, and Suhaib A. Fahmy. REFL: Resource-efficient federated learning. In *Proceedings of the Eighteenth European Conference on Computer Systems*, pages 215–232. ACM, 2023.

[BDLMB23] Claudio Battiloro, Paolo Di Lorenzo, Mattia Merluzzi, and Sergio Barbarossa. Lyapunov-based optimization of edge resources for energy-efficient adaptive federated learning. *IEEE Transactions on Green Communications and Networking*, 7(1):265–280, 2023.

[CLC+22] Xu Chu, Xiaoyang Liu, Qimei Chen, Yunfei Xiong, Juanjuan Wang, Han Yu, and Xiang Hu. Joint accuracy and resource allocation for green federated learning networks. In Meikang Qiu, Keke Gai, and Han Qiu, editors, *Smart Computing and Communication*, volume 13202, pages 154–163. Springer International Publishing, 2022. Series Title: Lecture Notes in Computer Science.

[CLX+22] Rui Chen, Liang Li, Kaiping Xue, Chi Zhang, Miao Pan, and Yuguang Fang. Energy efficient federated learning over heterogeneous mobile devices via joint design of weight quantization and wireless transmission. *IEEE Transactions on Mobile Computing*, pages 1–13, 2022.

[CYLN23] Zhixiong Chen, Wenqiang Yi, Yuanwei Liu, and Arumugam Nallanathan. Knowledge-aided federated learning for energy-limited wireless networks. *IEEE Transactions on Communications*, 71(6):3368–3386, 2023.

[CYS+21a] Mingzhe Chen, Zhaohui Yang, Walid Saad, Changchuan Yin, H. Vincent Poor, and Shuguang Cui. https://github.com/mzchen0/Wireless-FL, 2021. [Accessed 11-07-2024].

[CYS+21b] Mingzhe Chen, Zhaohui Yang, Walid Saad, Changchuan Yin, H. Vincent Poor, and Shuguang Cui. A joint learning and communications framework for federated learning over wireless networks. *IEEE Transactions on Wireless Communications*, 20(1):269–283, 2021.

[DTN+21a] Canh T. Dinh, Nguyen H. Tran, Minh N. H. Nguyen, Choong Seon Hong, Wei Bao, Albert Y. Zomaya, and Vincent Gramoli. https://github.com/CharlieDinh/FEDL, 2021. [Accessed 11-07-2024].

[DTN+21b] Canh T. Dinh, Nguyen H. Tran, Minh N. H. Nguyen, Choong Seon Hong, Wei Bao, Albert Y. Zomaya, and Vincent Gramoli. Federated learning over wireless networks: Convergence analysis and resource allocation. *IEEE/ACM Transactions on Networking*, 29(1):398–409, 2021.

[HCS+22] Rami Hamdi, Mingzhe Chen, Ahmed Ben Said, Marwa Qaraqe, and H. Vincent Poor. Federated learning over energy harvesting wireless networks. *IEEE Internet of Things Journal*, 9(1):92–103, 2022.

[JHD22] Richeng Jin, Xiaofan He, and Huaiyu Dai. Communication efficient federated learning with energy awareness over wireless networks. *IEEE Transactions on Wireless Communications*, 21(7):5204–5219, 2022.

[JPC+20] Joohyung Jeon, Soohyun Park, Minseok Choi, Joongheon Kim, Young-Bin Kwon, and Sungrae Cho. Optimal user selection for high-performance and stabilized energy-efficient feder2020ated learning platforms. *Electronics*, 9(9):1359, 2020.

[KSMD24a] Minsu Kim, Walid Saad, Mohammad Mozaffari, and Mérouane Debbah. https://github.com/news-vt/Green-Quantized-FL-over-Wireless-Networks-An-Energy-Efficient-Design, 2024. [Accessed 11-07-2024].

[KSMD24b] Minsu Kim, Walid Saad, Mohammad Mozaffari, and Mérouane Debbah. Green, quantized federated learning over wireless networks: An energy-efficient design. *IEEE Transactions on Wireless Communications*, 23(2):1386–1402, 2024.

[KW21] Young Geun Kim and Carole-Jean Wu. AutoFL: Enabling heterogeneity-aware energy efficient federated learning. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 183–198. ACM, 2021.

[LLW+21] Bing Luo, Xiang Li, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. Cost-effective federated learning design. In *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[LSH+21] 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. IEEE, 2021.

[LWCX20] Li Li, Jun Wang, Xu Chen, and Cheng-Zhong Xu. Multi-layer coordination for high-performance energy-efficient federated learning. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2020.

[LZP21] Xueting Luo, Zhongyuan Zhao, and Mugen Peng. Tradeoff between model accuracy and cost for federated learning in the mobile edge computing systems. In *2021 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6. IEEE, 2021.

[MX21a] Xiaopeng Mo and Jie Xu. https://github.com/Bellypoly/On_simulating_energy_consumption_of_federated_learning_systems, 2021. [Accessed 11-07-2024].

[MX21b] 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, 2021.

[Pil23a] Laercio Lima Pilla. https://github.com/llpilla/energy-optimal-federated-learning, 2023. [Accessed 11-07-2024].

[Pil23b] Laercio Lima Pilla. Scheduling algorithms for federated learning with minimal energy consumption. *IEEE Transactions on Parallel and Distributed Systems*, 34(4):1215–1226, 2023.

[RYDF+23] Rukhsana Ruby, Hailiang Yang, Felipe A. P. De Figueiredo, Thien Huynh-The, and Kaishun Wu. Energy-efficient multiprocessor-based computation and communication resource allocation in two-tier federated learning networks. *IEEE Internet of Things Journal*, 10(7):5689–5703, 2023.

[SKRB21a] Stefano Savazzi, Sanaz Kianoush, Vittorio Rampa, and Mehdi Bennis. https://github.com/labRadioVision/federated_learning_carbon_footprint, 2021. [Accessed 11-07-2024].

[SKRB21b] Stefano Savazzi, Sanaz Kianoush, Vittorio Rampa, and Mehdi Bennis. A framework for energy and carbon footprint analysis of distributed and federated edge learning. In *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1564–1569. IEEE, 2021.

[SRKB23a] Stefano Savazzi, Vittorio Rampa, Sanaz Kianoush, and Mehdi Bennis. https://github.com/labRadioVision/federated, 2023. [Accessed 11-07-2024].

[SRKB23b] Stefano Savazzi, Vittorio Rampa, Sanaz Kianoush, and Mehdi Bennis. An energy and carbon footprint analysis of distributed and federated learning. *IEEE Transactions on Green Communications and Networking*, 7(1):248–264, 2023.

[SZG20] Yuxuan Sun, Sheng Zhou, and Deniz Gunduz. Energy-aware analog aggregation for federated learning with redundant data. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2020.

[TBZ+19a] Nguyen H. Tran, Wei Bao, Albert Zomaya, Minh N. H. Nguyen, and Choong Seon Hong. https://github.com/nhatminh/FEDL-INFOCOM, 2019. [Accessed 11-07-2024].

[TBZ+19b] 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 - IEEE Conference on Computer Communications*, pages 1387–1395. IEEE, 2019.

[XLL+24] Zichuan Xu, Dongrui Li, Weifa Liang, Wenzheng Xu, Qiufen Xia, Pan Zhou, Omer F. Rana, and Hao Li. Energy or accuracy? near-optimal user selection and aggregator placement for federated learning in MEC. *IEEE Transactions on Mobile Computing*, 23(3):2470–2485, 2024.

[YCS+21] Zhaohui Yang, Mingzhe Chen, Walid Saad, Choong Seon Hong, and Mohammad Shikh-Bahaei. Energy efficient federated learning over wireless communication networks. *IEEE Transactions on Wireless Communications*, 20(3):1935–1949, 2021.

[YLS+24] Lihua Yin, Sixin Lin, Zhe Sun, Ran Li, Yuanyuan He, and Zhiqiang Hao. A game-theoretic approach for federated learning: A trade-off among privacy, accuracy and energy. *Digital Communications and Networks*, 10(2):389–403, 2024.

[ZLG20] Yufeng Zhan, Peng Li, and Song Guo. Experience-driven computational resource allocation of federated learning by deep reinforcement learning. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 234–243. IEEE, 2020.

[ZLWG22] Yufeng Zhan, Peng Li, Leijie Wu, and Song Guo. L4l: Experience-driven computational resource control in federated learning. *IEEE Transactions on Computers*, 71(4):971–983, 2022.

[ZPKH21] 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.

[ZWX+24] Long Zhang, Suiyuan Wu, Haitao Xu, Qilie Liu, Choong Seon Hong, and Zhu Han. Optimizing tradeoff between learning speed and cost for federated-learning-enabled industrial IoT. *IEEE Internet of Things Journal*, 11(7):11663–11678, 2024.