# CS 332: Federated Learning

Peter Richtárik
https://richtarik.org


Spring 2023

# Contents I



**CS 332: Federated Learning**

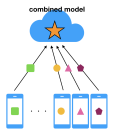
Peter Richtárik

KAUST
King Abdullah University of Science and Technology

# Part 1: Course Organization

# About Me

- **Name:** Peter Richtárik
- **Position:** Professor of Computer Science
  - also affiliated with AMCS and STAT (have students in CS, AMCS and STAT)
- **Office:** Library (AI Initiative Area)
- **Website:** `https://richtarik.org`
- **Email:** peter.richtarik@kaust.edu.sa
- **Brief Academic CV:**
  - 2019–now, Professor, KAUST
  - 2017–2019, Associate Professor, KAUST
  - 2009–2017, Assistant and later Associate Professor, University of Edinburgh, United Kingdom
  - 2007–2009, Postdoc, UC Louvain, Belgium
  - 2002–2007, PhD student, Cornell, USA
  - 1996–2001, Bc and Mgr student, Comenius University, Slovakia
- **Nationality:** Slovakia, European Union
- **Research:** randomized convex and nonconvex optimization, machine learning, optimization for machine learning, randomized linear algebra, distributed optimization and learning, federated learning.



# Logistics

- **Lecture Time:** Tuesdays, 9:00am–12:00pm
  - Some lectures will be cancelled (and most of these rescheduled) due to reasons such as i) key conference deadlines, ii) university holidays, iii) me being on annual leave:
    - February 21 (mid semester break) and February 28 (maybe will be rescheduled, maybe we'll do it via Zoom)
    - more will be announced later
- **Lecture Location:** + Room 4120 @ Bldg 9
- **Office Hours:** Immediately after each lecture, in the Zoom / lecture room. Additional office hours by email appointment.
- **Teaching Assistant:** No TA

# Brief Course Description

This is a PhD level course in a new branch of machine learning: **federated learning (FL).**

- In FL, (mostly supervised) machine learning models are trained in a massively distributed fashion on millions of mobile devices with an explicit effort to preserve the privacy of users' data.
- FL is a new field, with few theoretical results and early production systems (e.g., Tensor Flow Federated).

# Goals and Objectives

- Understand the **key concepts** behind FL
- Get familiar with **key theoretical results** of FL
- Become familiar with **key research papers** in the FL field
- Write a **report/paper with original applied or theoretical research** in the area of FL

# Knowledge Required

**Essential:** Good knowledge of

- multivariate calculus
- linear algebra
- probability theory
- algorithms
- optimization
- programming

**Ideal:** Essential knowledge +

- CS 331: Stochastic Gradient Descent Methods (ideal background)
- Mastery of some subfield of Machine Learning (e.g., ML systems, unsupervised learning)
- Prior exposure to TensorFlow, PyTorch

# Reference Text

- FL is a young field, originating in 2016/2017 through in the papers of McMahan et al [6], Konečný et al [4, 3] and McMahan et al [8].
  - there is no textbook on the topic
  - there is no university-level course on the topic anywhere (as far as I know)
- As there is no textbook on this topic, the course material will be based on **recent papers** selected by the instructor.
  - During the first week of the semester, read the short review article of Lin et al [5] and subsequently the long review article of Kairouz et al [2], and also watch the videos posted in the Resources section of Piazza.
- Read the lecture notes/slides - these will be written by you!
- Read a lot of FL papers on your own during the semester!

# Method of Evaluation

- **70% project**
  - results will be written up in the form of a NeurIPS research paper
- **30% in-class participation** for activities such as
  - asking questions
  - delivering talks on the progress of your project
  - delivering lectures/tutorials on topics selected by the instructor
  - (writing LaTeX lecture notes for delivered lectures)
- There are **no assignments** and **no exams**

# Final Project - I

- **Basic idea:** Conduct original research in FL during the course, write a paper about this research and submit it to the NeurIPS 2022 (Advances in Neural Information Processing Systems) conference:
  *https://neurips.cc*
- **Submission of paper to NeurIPS:** Submission to NeurIPS is encouraged (I am assuming this is also what you want!) but not compulsory. Whether or not you submit to NeurIPS does not affect grading in any way. If you decide to submit, do it via CMT3 or OpenReview – I do not know which of these platforms will NeurIPS decide to use this year. It was CMT3 for a long time until 2020, and OpenReview in 2021.

# Final Project - II

Google Scholar

Top publications

Categories > Engineering & Computer Science > **Artificial Intelligence**

| | Publication | h5-index | h5-median |
|---|---|---|---|
| 1. | International Conference on Learning Representations | 203 | 359 |
| 2. | Neural Information Processing Systems | 198 | 377 |
| 3. | International Conference on Machine Learning (ICML) | 171 | 309 |
| 4. | AAAI Conference on Artificial Intelligence | 126 | 183 |
| 5. | Expert Systems with Applications | 111 | 152 |
| 6. | IEEE Transactions On Systems, Man And Cybernetics Part B, Cybernetics | 111 | 150 |
| 7. | IEEE Transactions on Neural Networks and Learning Systems | 107 | 146 |
| 8. | Neurocomputing | 100 | 143 |
| 9. | Applied Soft Computing | 96 | 123 |
| 10. | International Joint Conference on Artificial Intelligence (IJCAI) | 95 | 140 |
| 11. | IEEE Transactions on Fuzzy Systems | 87 | 117 |
| 12. | Knowledge-Based Systems | 85 | 121 |
| 13. | The Journal of Machine Learning Research | 82 | 153 |
| 14. | Neural Computing and Applications | 67 | 98 |
| 15. | Neural Networks | 64 | 92 |
| 16. | International Conference on Artificial Intelligence and Statistics | 57 | 89 |
| 17. | Engineering Applications of Artificial Intelligence | 57 | 78 |
| 18. | Robotics and Autonomous Systems | 56 | 87 |
| 19. | Conference on Learning Theory (COLT) | 54 | 80 |
| 20. | Journal of Intelligent & Fuzzy Systems | 50 | 79 |

Figure: AI conference rankings by Google Scholar Metrics. Retrieved on January 23, 2021.

# Final Project - III

- **Submission of paper for grading:** Send your final paper (in pdf format) to me by email: peter.richtarik@kaust.edu.sa
- **Paper format:** NeurIPS 2022 format:
  - use of the official NeurIPS 2022 style files,
  - adherence to all conventions when writing a NeurIPS paper (e.g., the use of a proper title, abstract, introduction, literature review, contributions, main findings, experimental evaluation, bibliography, appendix/supplementary file)
  - in short, your Final Project should exactly like a paper which one would be able and willing to submit to the conference, without it violating an rules and conventions
- **Deadline: last day of the semester**
  - The deadline for NeurIPS 2023 was not announced yet.
  - The Spring 2023 semester ends on May 11, 2023; the exam week is during May 14–17, 2023.
  - **Failure of submission by the deadline will lead to zero points for the project and a failure in the course. Hence, it is critical to start working on the project early and keep working hard throughout the duration of the course.**

# Final Project - IV

- **Authorship:** By default, the entire paper (including the idea, results, experiments and writing) needs to be each student's individual work. However, I may grant exceptions, such as:
  - I am happy to suggest a suitable research topic.
  - I may allow a small team of people (e.g., 2 students) to work on a single project if I believe this is a good idea benefiting both the students and the project. In such a case, each student will be marked on their contribution to the paper.
  - In rare circumstances, I may even allow the formation of a larger project team. Example of such a project from a course I taught in Spring 2019: Horváth et al [1].
  - I may accept being a collaborator in some cases (e.g., if I suggest a topic, offer ideas, contribute to theory, writing and so on), time permitting.
  - If you plan to submit to NeurIPS, and if you want to involve your supervisor in the research, it should be possible to make that happen. However, I would need to have a discussion with the supervisor before this is approved as I need to make sure your grade will be based on your contribution to the work.

**CS 332: Federated Learning**

Peter Richtárik

KAUST
King Abdullah University of Science and Technology

# Part 2: Introduction to Federated Learning

# FL: Overview

- **FL was introduced in 2016** by Google (McMahan and his team), Jakub Konečný and P.R. [4, 3, 8]
- Launched in 4/2017 via Gboard on Android devices: `https://ai.googleblog.com/2017/04/federated-learning-collaborative.html`
- The CEO of Google, Sundar Pichai, said:

  *"...we continue to set the pace in machine learning and AI research. We introduced a new technique for training deep neural networks on mobile devices called Federated Learning. This technique enables people to run a shared machine learning model, while keeping the underlying data stored locally on mobile phones."*

- Now **in use by billions of Android devices**
- Apple, Samsung, Tencent, Facebook, Sony, NVidia have FL efforts
- Several startups at ICML 2019 / NeurIPS 2019, and many more companies doing this since

# The Rise of Federated Learning - I

- Proliferation of **devices/entities,** such as
  - mobile phones and tablets
  - wearable devices (e.g., smart watches, smart glasses)
  - autonomous vehicles (e.g., cars, drones)
  - smart home devices (e.g., thermostats)
  - organizations/corporations (e.g., hospitals)


Figure: IoT devices

# The Rise of Federated Learning - II

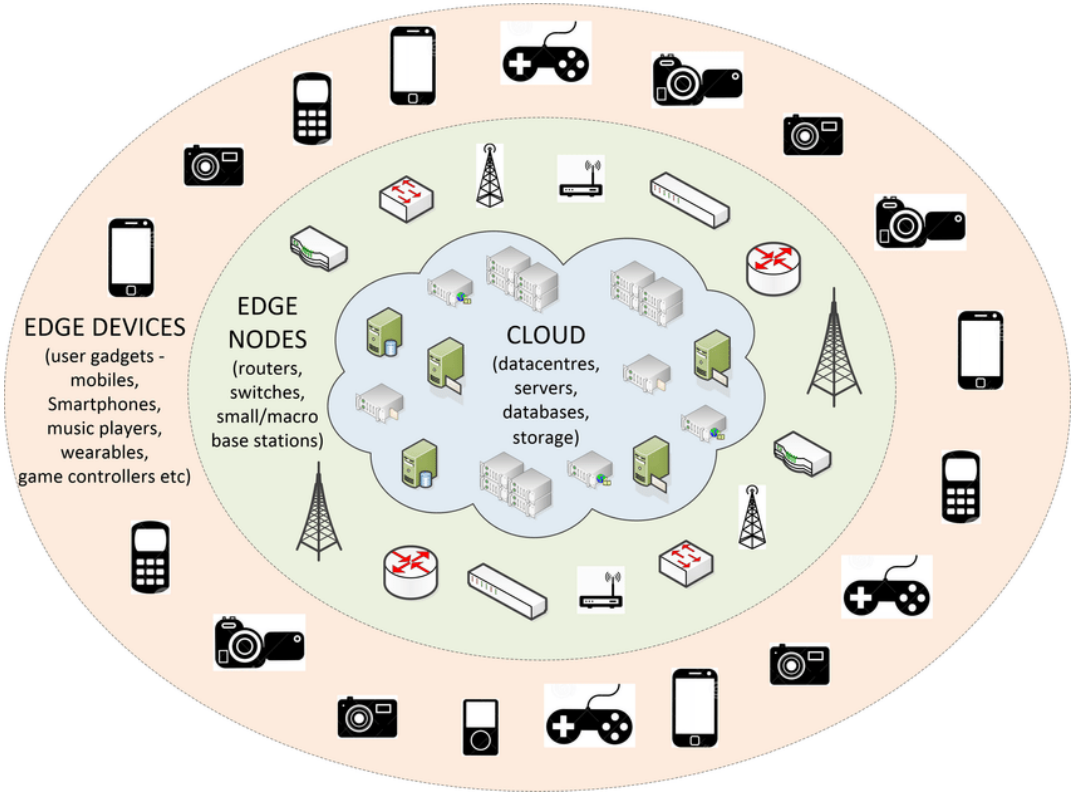



Figure: Edge devices

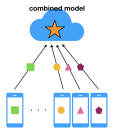
- These **devices are becoming more powerful & smarter**
  - more storage
  - better & more processors
  - better software

# The Rise of Federated Learning - III

- Increasing amounts of **data is captured and stored on mobile devices**
  - photos
  - videos
  - messages
  - emails
  - device settings
  - device usage patterns
- Moving data to a centralized location for processing is problematic:
  - it is **energy inefficient** (it takes time and bandwidth to transfer lots of data)
  - many users are reluctant to part with their data due to **privacy** concerns
  - could be **expensive** to users (paying for cloud storage)

# Traditional Centralized Approach to Machine Leaning

- Move all data to a centralized (proprietary) location (data warehouse)
- Run ML algorithms on the centralized data to build a model
- Push the learned model (perhaps in a compressed/modified form) to devices
- Users use the model



# Federated Learning: "Decentralized" Approach to Machine Learning

**Key idea behind FL:** View the network of mobile devices as a distributed-memory computer and train models that way.

- Keep the data on the devices (one step towards protecting the data)
- Run ML algorithms on the network of mobile devices *as if* this was a large distributed computer
- Take further steps to ensure certain level of protection / privacy of suers' data
- Push the learned model (perhaps in a compressed/modified form) to devices
- Users use the model

# Example Application: Next-Word Prediction


Figure: An example application of federated learning for the task of next-word prediction on mobile phones. To preserve the privacy of the text data and to reduce strain on the network, we seek to train a predictor in a distributed fashion, rather than sending the raw data to a central server. In this setup, remote devices communicate with a central server periodically to learn a global model. At each communication round, a subset of selected phones performs local training on their non-identically-distributed user data, and sends these local updates to the server. After incorporating the updates, the server then sends back the new global model to another subset of devices. This iterative training process continues across the network until convergence is reached or some stopping criterion is met.

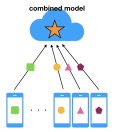
Credits: Application due to Google (2017) [7]; image and text from [5].

# (Supervised) FL as an Optimization Problem I

- $n$ **participating devices:** $1, 2, \ldots, n$
- Device $i$ contains **training data** $\mathcal{D}_i = \{(a_{ij}, b_{ij}) \ : \ j = 1, \ldots, n_i\}$.
  - $a_{ij} \in \mathcal{A}$: input
  - $b_{ij} \in \mathcal{B}$: output (label)
- Want to learn a **single global model** $M_x$ parameterized by vector $x \in \mathbb{R}^d$ (e.g., weights of a neural network) such that

$$M_x(a_{ij}) \approx b_{ij} \qquad \text{for all} \qquad i, j.$$

- We choose a **loss function** $\mathcal{L} : \mathcal{B} \times \mathcal{B} \to \mathbb{R}$ which is supposed to be small if $M_x(a_{ij})$ is a good prediction of $b_{ij} \in \mathcal{B}$



# Approach 1: All Data Points are Equally Important - I

**Goal: Find model $x \in \mathbb{R}^d$ minimizing the total loss across all training data:**

$$\min_{x \in \mathbb{R}^d} \underbrace{\frac{1}{\sum_{i=1}^n n_i}}_{\text{normalization factor}} \times \underbrace{\sum_{i=1}^n \sum_{j=1}^{n_i} \mathcal{L}(M_x(a_{ij}), b_{ij})}_{\text{total loss}} \tag{1}$$

- Problem (3) is a distributed version of **Empirical Risk Minimization.**
- We can rewrite the aggregate loss function as

$$\frac{1}{\sum_{i=1}^n n_i} \sum_{i=1}^n \sum_{j=1}^{n_i} \mathcal{L}(M_x(a_{ij}), b_{ij}) = \sum_{i=1}^n \underbrace{\frac{n_i}{\sum_{t=1}^n n_t}}_{\stackrel{\text{def}}{=} p_i} \underbrace{\frac{1}{n_i} \sum_{j=1}^{n_i} \underbrace{\mathcal{L}(M_x(a_{ij}), b_{ij})}_{\stackrel{\text{def}}{=} f_{ij}(x)}}_{\stackrel{\text{def}}{=} f_i(x)}$$

$$= \sum_{i=1}^n p_i f_i(x).$$

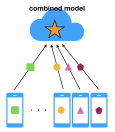
# Approach 1: All Data Points are Equally Important - II

- So, (3) can be written in the form

$$\boxed{\min_{x \in \mathbb{R}^d} \sum_{i=1}^{n} p_i f_i(x)} \tag{2}$$

where

$$p_i = \frac{n_i}{\sum_{t=1}^{n} n_t}, \qquad f_i(x) = \frac{1}{n_i} \sum_{j=1}^{n_i} f_{ij}(x).$$



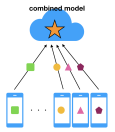
# Approach 2: All Devices are Equally Important - I

**Goal: Find model $x \in \mathbb{R}^d$ minimizing the total average loss across all devices:**

$$\min_{x \in \mathbb{R}^d} \underbrace{\frac{1}{n}}_{\text{normalizing factor}} \times \underbrace{\sum_{i=1}^{n} \underbrace{\frac{1}{n_i} \sum_{j=1}^{n_i} \mathcal{L}(M_x(a_{ij}), b_{ij})}_{\text{average loss on device } i}}_{\text{total average loss}} \tag{3}$$

- We can rewrite the aggregate loss function as

$$\frac{1}{n} \sum_{i=1}^{n} \underbrace{\frac{1}{n_i} \sum_{j=1}^{n_i} \mathcal{L}(M_x(a_{ij}), b_{ij})}_{\stackrel{\text{def}}{=} f_i(x)} = \frac{1}{n} \sum_{i=1}^{n} f_i(x).$$



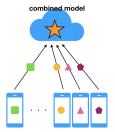
# Approach 2: All Devices are Equally Important - II

- So, (3) can be written in the form

$$\boxed{\min_{x \in \mathbb{R}^d} \sum_{i=1}^{n} p_i f_i(x)} \tag{4}$$

where

$$p_i = \frac{1}{n}, \qquad f_i(x) = \frac{1}{n_i} \sum_{j=1}^{n_i} f_{ij}(x).$$



# Challenges of FL - I

FL differs significantly from traditional approaches to distributed machine learning.

**Some of the Key Characteristic Challenges of FL are:**

1. **Expensive Communication**
   - Communication is typically more expensive than computation by orders of magnitude
   - **Desire:** Reduce the number of communication rounds
   - **Desire:** Reduce the number of bits sent in each communication
   - **Tool:** Communication compression
   - **Tool:** Variance reduction / error compensation
   - **Tool:** Do more work before communication (e.g., do more gradient steps on each device: "local GD/local SGD")
   - **Tool:** Decentralized training (on a general network rather than on a star network)
2. **System Heterogeneity**
   - Devices differ in many ways:
     - storage (capacity, read/write speed)
     - processing power (CPU, GPU, memory)

# Challenges of FL - II

- communication abilities (3G / 4G / 5G network)
- power (battery level vs plugged in)
- availability (e.g., busy with other work)

- **Desire:** Tolerate heterogeneous devices
- **Desire:** In each round, should be able to work with a subset of devices which are available ("partial participation")
- **Tool:** Asynchronous communication
- **Tool:** Sampling, minibatching

3. **System Size**
   - The number $n$ of devices can be very large (millions-hundreds of millions)
   - Does inclusion of additional devices help reducing the training time?
   - **Desire:** Be able to train with many devices
   - **Tool:** Variance reduction, error-compensation
4. **Data Heterogeneity**
   - The empirical data distributions $\mathcal{D}_1, \ldots, \mathcal{D}_n$ may be widely different and/or could be sampled from very different ground truth distributions
   - The devices may posses widely different numbers ($n_i$) of data points

# Challenges of FL - III

- **Desire:** Training with heterogeneous data is hard. Construct training methods able to train with heterogeneous data.
- **Desire:** A single global model may be too generic to be useful. Personalize the global model to each user.
- **Tool:** Design new optimization objectives for FL (e.g., "local-global model", use meta-learning)
- **Tool:** Sample more important data/device more often

5. **Data Volume**
   - the total volume of the training data sets $\{\mathcal{D}_i\}$ can be very large
   - **Desire:** Tolerate large training data sizes
   - **Tool:** Subsampling
6. **Model Size**
   - Many state-of-the-art models are over-parameterized, meaning that the number of parameters $d$ is larger (often by an order of magnitude) than the number of data points $\sum_{i=1}^{n} n_i$.
   - The number of devices $n$ can be very large, and the total number of data points can be much larger
   - Over-parameterized models could be too large to fit into device memory

# Challenges of FL - IV

- **Desire:** Be able to train useful under-parameterized models
- **Desire:** Be able to compress the model (trimming its size) after training without much loss in its predictive power
- **Tool:** Model compression

7. **Privacy Concerns**
   - While the training data is stored on devices, some information is communicated outside of the devices by the learning algorithm
   - If we limit the communication too much, we may not be able to learn a useful model
   - **Desire:** Strike a trade-off between being able to learn a useful model, and protection of users' data
   - **Desire:** Avoid communicating raw data points; communicate model updates (e.g., gradients), or randomly obfuscated model updates
   - **Tool:** Secure multi-party computation
   - **Tool:** Differential privacy
   - **Tool:** Homomorphic encryption

# Bibliography I


[1] Samuel Horváth, Chen-Yu Ho, Ľudovít Horváth, Atal Narayan Sahu, Marco Canini, and Peter Richtárik.

Natural compression for distributed deep learning.

*arXiv preprint arXiv:1905.10988*, 2019.

[2] Peter Kairouz and co authors.

Advances and open problems in federated learning.

*arXiv preprint arXiv:1912.04977*, 2019.

[3] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik.

Federated optimization: distributed machine learning for on-device intelligence.

*arXiv:1610.02527*, 2016.

[4] Jakub Konečný, H. Brendan McMahan, Felix Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon.

Federated learning: strategies for improving communication efficiency.

In *NIPS Private Multi-Party Machine Learning Workshop*, 2016.

[5] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith.

Federated learning: challenges, methods, and future directions.

*arXiv preprint arXiv:1908.07873*, 2019.


# Bibliography II


[6] Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas.

Federated learning of deep networks using model averaging.

*arXiv preprint arXiv:1602.05629*, 2016.

[7] Brendan McMahan and Daniel Ramage.

Federated learning: Collaborative machine learning without centralized training data.

GoogleAIBlog, April 2017.

[8] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas.

Communication-efficient learning of deep networks from decentralized data.

In *Proceedings of the 20 th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.