Communication-Efficient Approaches to Federated Deep Neural Networks

Technische Universität München

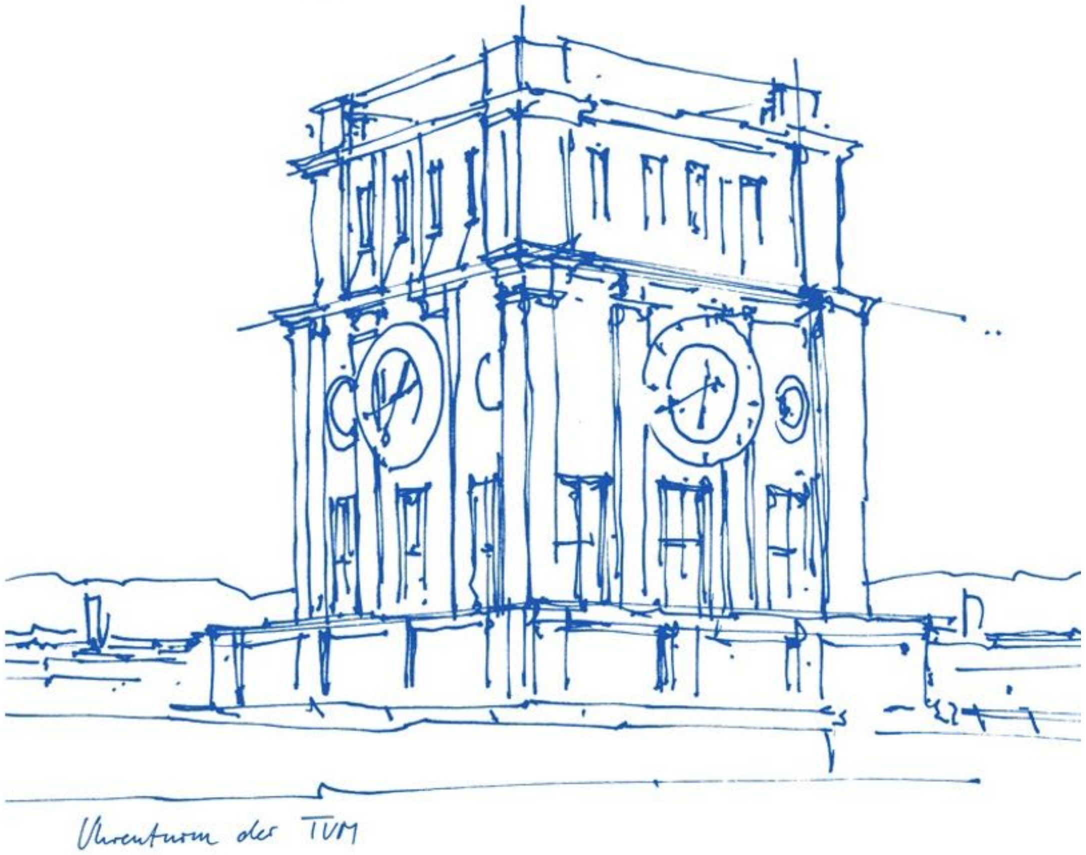
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Bachelor's Thesis

By: Adrian Edward Thomas Henkel
adrian.henkel@campus.lmu.de

Supervisor: Reza Nasirigerdeh
reza.nasirigerdeh@tum.de

Advisors: Prof. Dr. Jan Baumbach, and Dr. Josch Pauling
jan.baumbach@uni-hamburg.de
josch.pauling@wzw.tum.de



Agenda

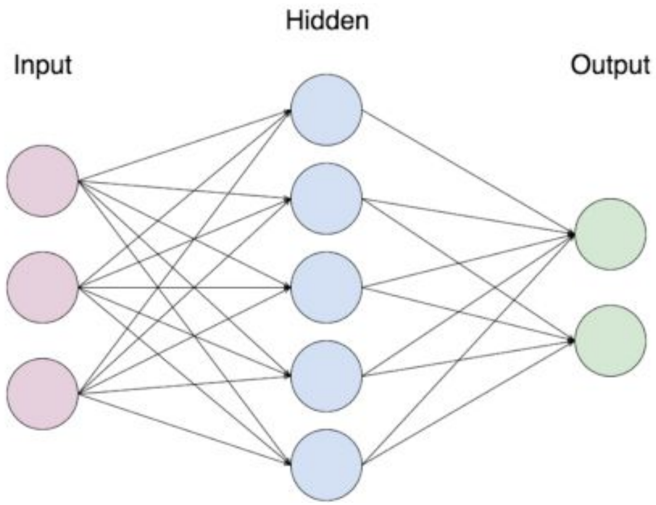
- Introduction
  - Deep neural networks
  - Federated learning
- Communication-efficient approaches
- Simulation framework
- Datasets and models
- Results
- Summary and outlook

A typical neural network

Technische Universität München

Architecture:

- One Input layer
- One or multiple hidden layers
- One output layer



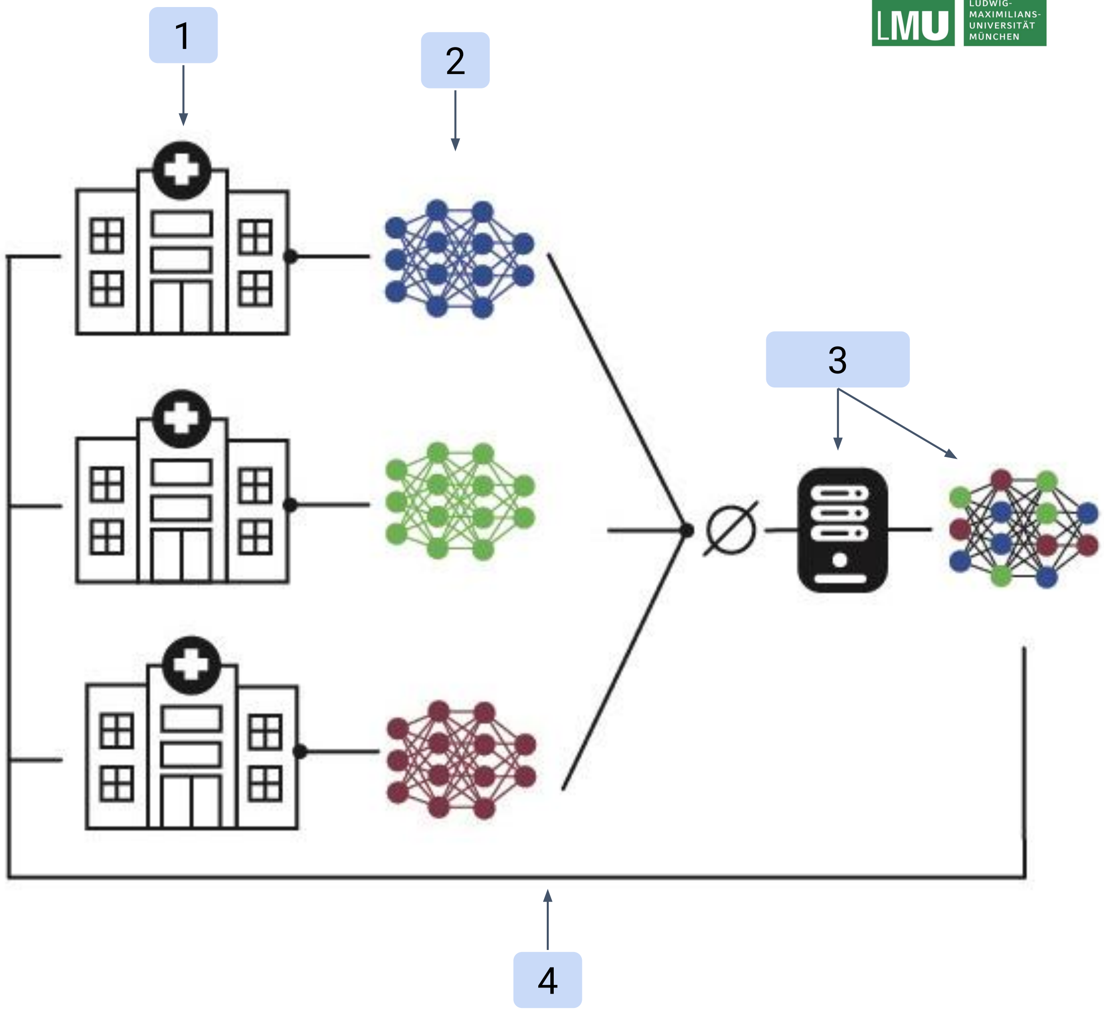
Training process:

- Initiate the model with random weights
- Iteratively update the network to minimize a loss function
- In each iteration:
  - Select a subset of the training data (batch)
  - Find updated weights that optimize the loss function of the batch (backpropagation)

Federated learning

- **Multiple clients** train a **global model** under the coordination of a **central server**

1. Each **client** trains the model on its local data
2. Each **client** shares the updated model with the server
3. The **server** computes the global model by taking the weighted average over the updated local models from the clients
4. The **server** sends the global model back to the clients
5. Repeat step 1-4 until the global model converges



FL - Challenges

- Privacy
  - Reconstruction of the private data from the model parameters is possible in FL
- Network communication
  - A huge amount of traffic might be exchanged over the network in FL
- Heterogeneous configurations
  - Clients with various computational and communication speeds
  - Non-IID (Independent and Identically Distributed) data across clients

- In this work, we **focus** on the **network communication challenge in deep neural networks**.

FL - Network bandwidth usage

K = number of clients
G = number of model parameters
L = size of each model parameter in bits
N = number of iterations (communication rounds)
π_C = network bandwidth usage

$$\longrightarrow \pi_C = 2 \cdot K \cdot G \cdot L \cdot N$$

- Communication-efficient approaches
  - Gradient quantification ($\downarrow$L)
  - Gradient sparsification ($\downarrow$G)
  - More local updates ($\downarrow$N)

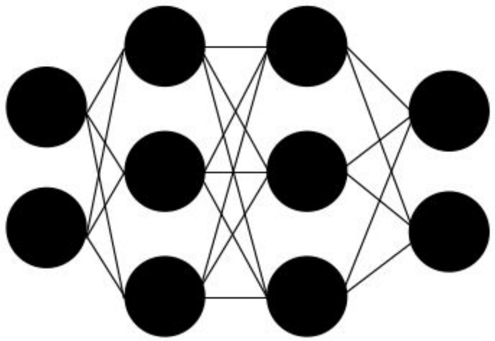
Example:
- K = 50
- G = 1.000.000
- L = 32
- N = 200

≈ 80GB

Gradient quantification (GQ)

- Default size of a parameter is 32 bits (L=32)
- Reduce the size of each parameter to 16 bits before sending the model
- The gradients are re-transformed into 32-bit representation before training

$$\pi_{GQ} = 2 \cdot K \cdot G \cdot \frac{L}{2} \cdot N = \frac{\pi_C}{2}$$

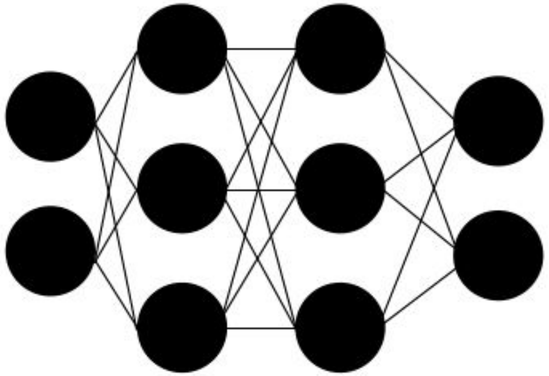


Gradient sparsification (GS)

Technische Universität München

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

- A model might be over-parameterized
- Calculate the difference between the global model and the updated model
- Eliminate all parameters under a Percentile P and determine their positions from a binary matrix
- Send the sparse model and the binary matrix



$$\pi_{GS}(P) = \left(\frac{\sum_{i=1}^{N} \frac{F_i^{server}}{100}}{2N} + \frac{\sum_{j=1}^{K} \sum_{i=1}^{N} \frac{F_i^{j}}{100}}{2K \cdot N} + \frac{1}{L} \right) \cdot \pi_C$$

Parameter after training

Parameter over the percentile

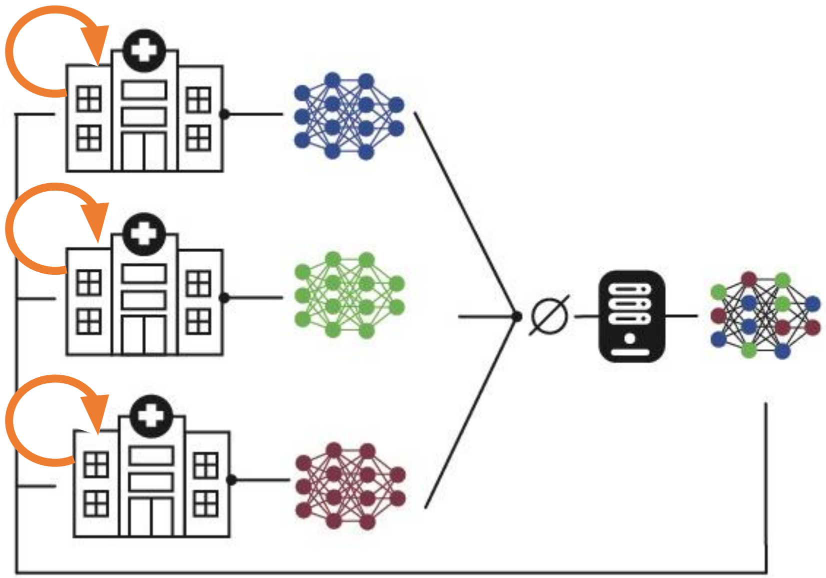
Old parameters of the global model

More local updates (MU)

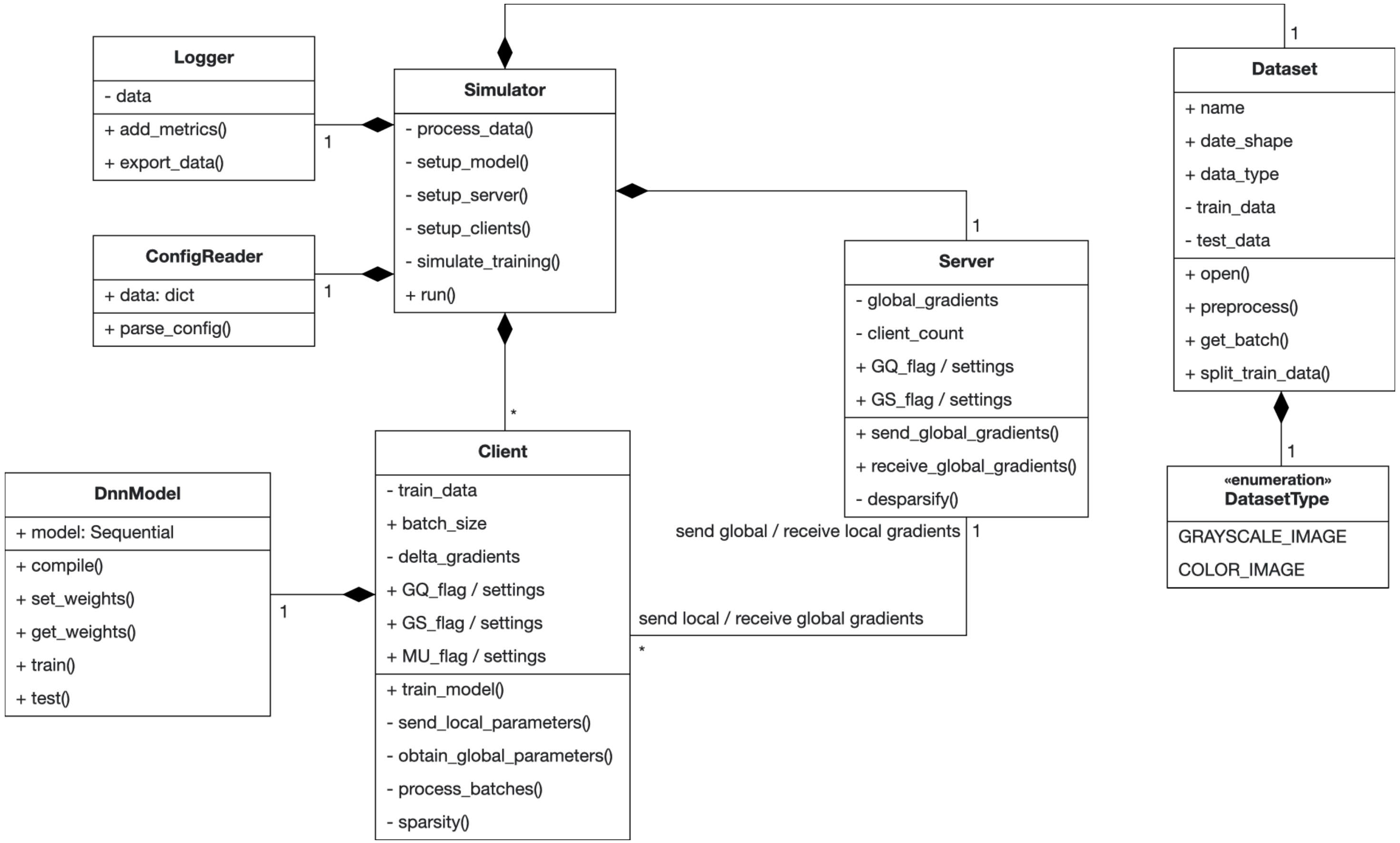
- Typically the number of local updates (E) is one
- Increase of the local updates in one iteration
- Faster convergence of the global model

  -> less iterations

$$\pi_{MU} = 2 \cdot K \cdot G \cdot L \cdot N' = \frac{N'}{N} \cdot \pi_C$$

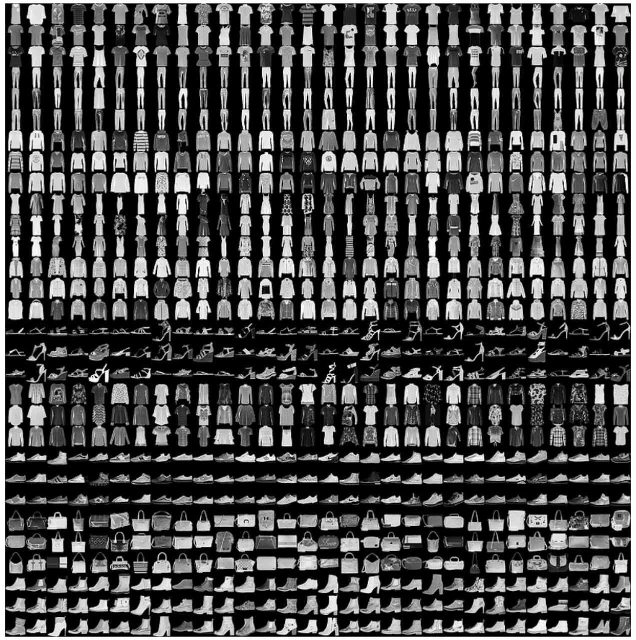


Simulation framework



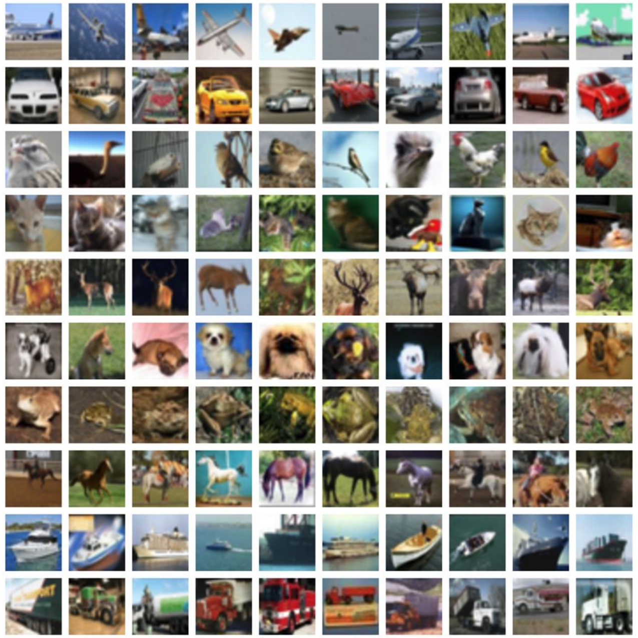
Repository: https://gitlab.lrz.de/000000000149C8EB/com-eff

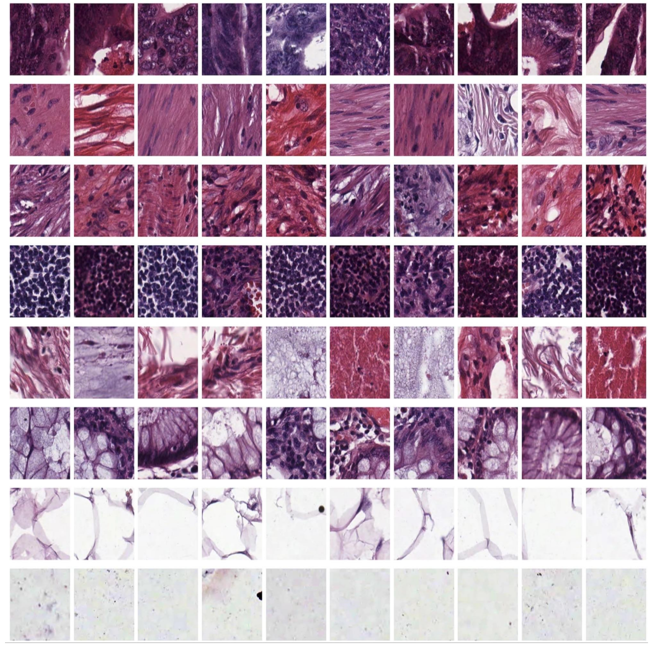
Datasets



Fashion-MNIST:
- Created 2017
- Zalando assortment
- 70000 samples
- 28 x 28 px. grayscale
- 10 classes

CIFAR-10
- Created 2009
- Subset of tiny image DB
- 60000 samples
- 32 x 32 px. RGB
- 10 classes





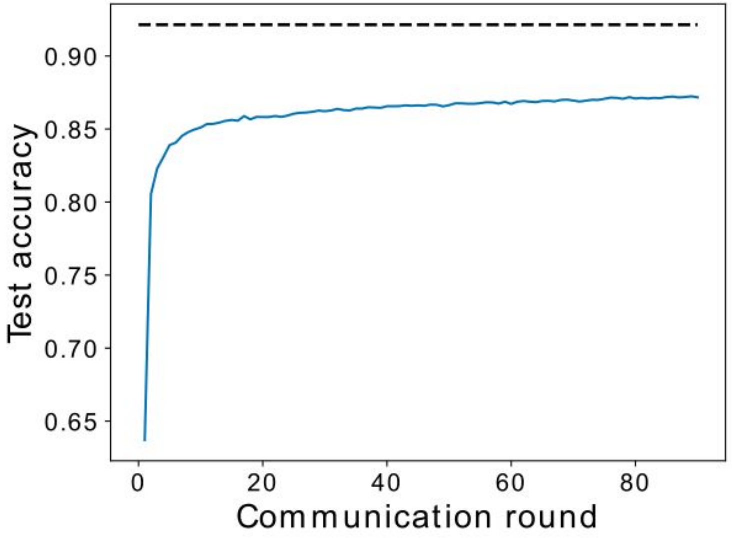
Colorectal Histology Dataset:
- Created 2009
- 4000 samples
- 150 x 150 px. RGB
- 8 classes

Models

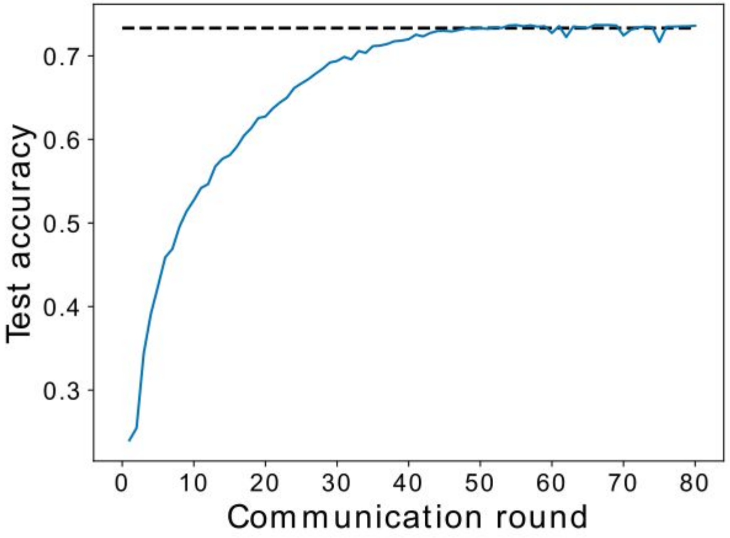
- **2CFNN**
  - Two convolutional layers
  - Two Max-Pooling layers
  - Dense layer with 512 neurons as penultimate layer
  - Total of 1,633,370 trainable parameters
- **3CFNN**
  - Three convolutional layers
  - Two Max-Pooling layers
  - Dense layer with 1024 neurons as penultimate layer
  - Total of 9,878,794 trainable parameters
- **VGG16**
  - Five blocks of two or three convolutional layers
  - Convolutional layers are ranging from 64 to 512 filters
  - Total of 65,087,304

Results - Accuracy

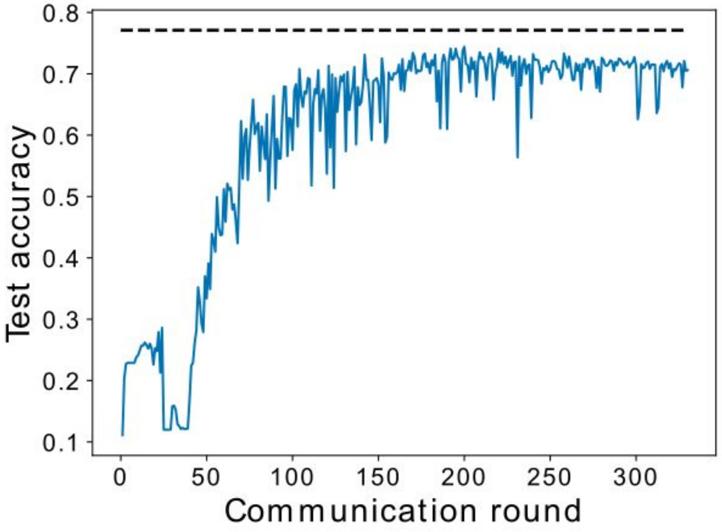
Gradient quantification



2CFNN - FMNIST



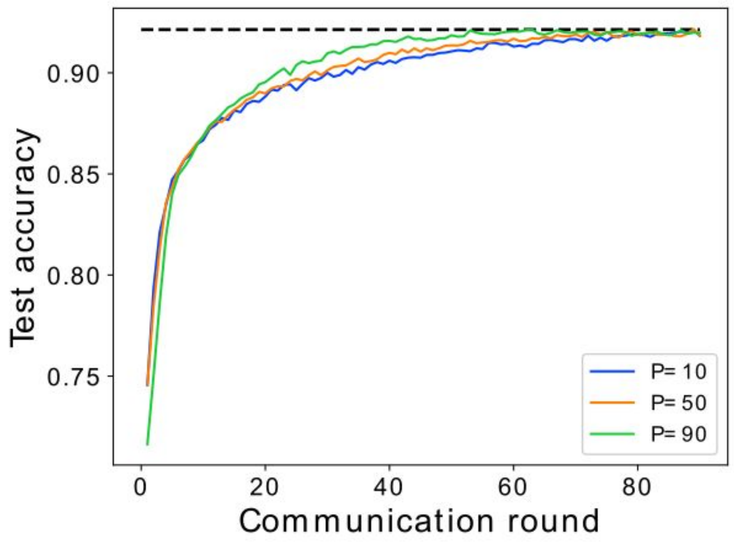
3CFNN - CIFAR-10



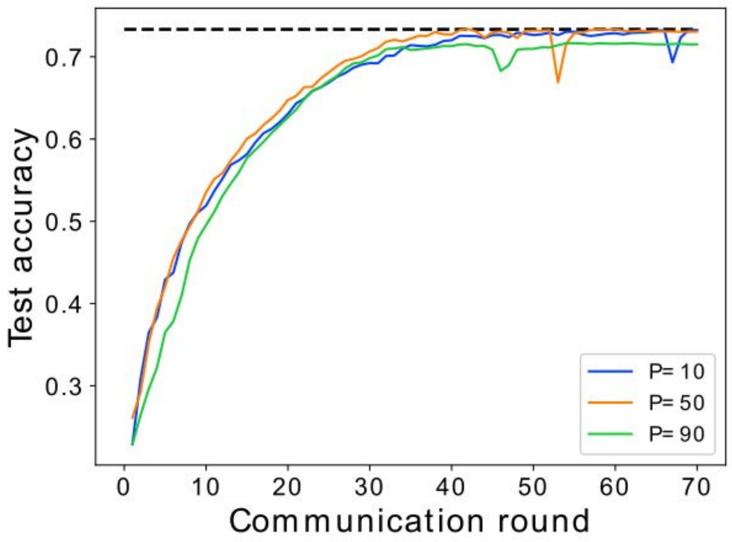
VGG16 - CCH

Results - Accuracy

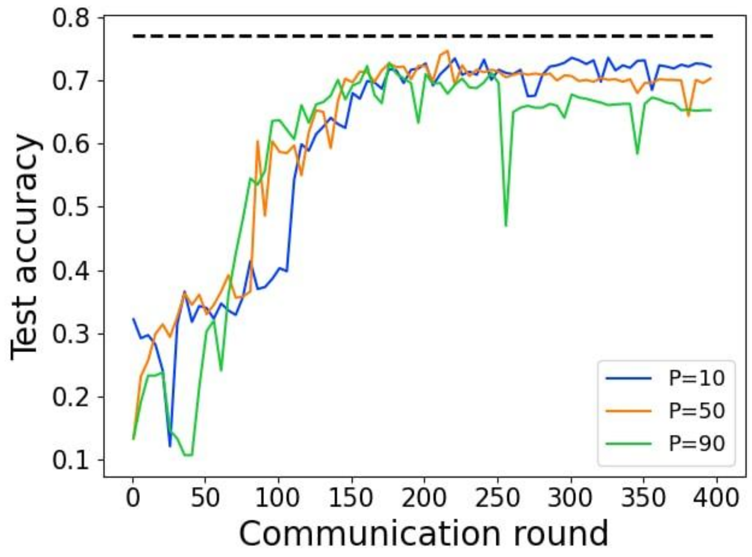
Gradient sparsification



2CFNN - FMNIST



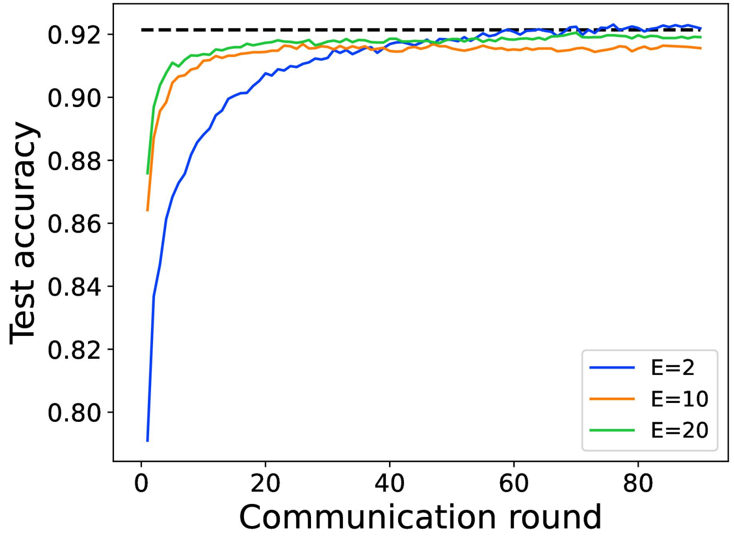
3CFNN - CIFAR-10



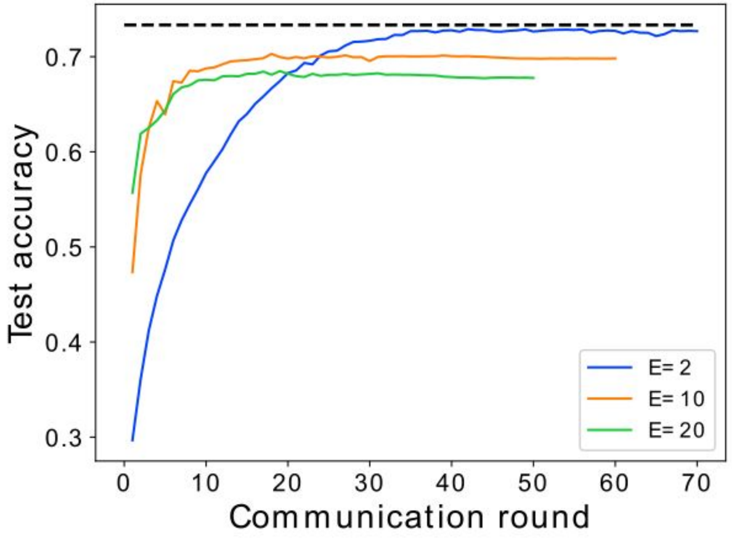
VGG16 - CCH

Results - Accuracy

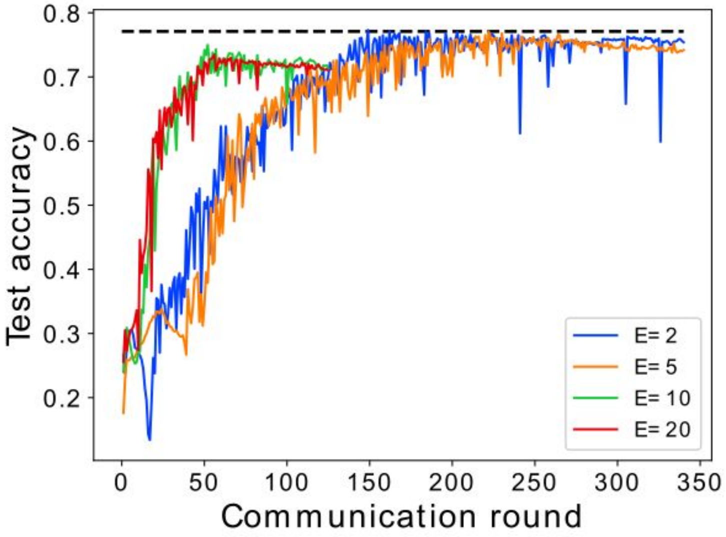
More local updates



2CFNN - FMNIST



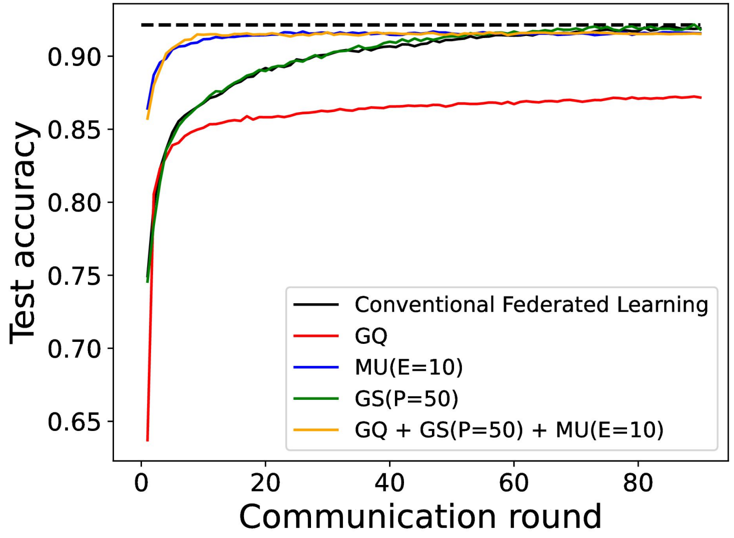
3CFNN - CIFAR-10



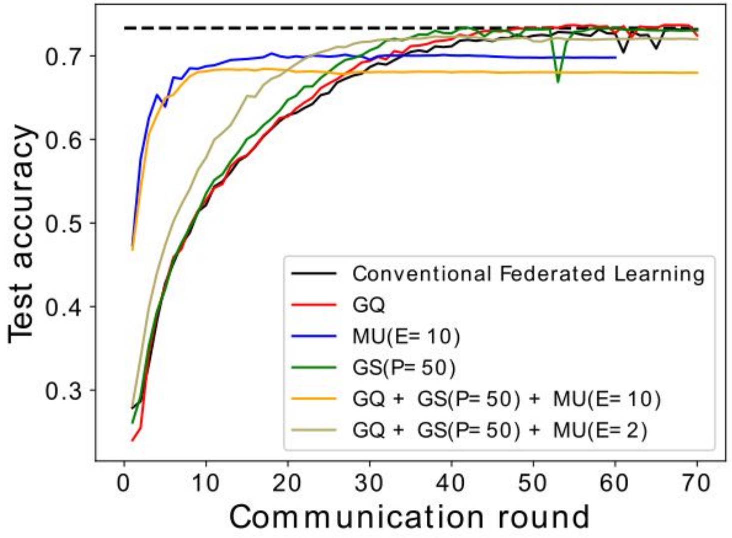
VGG16 - CCH

Results - Accuracy

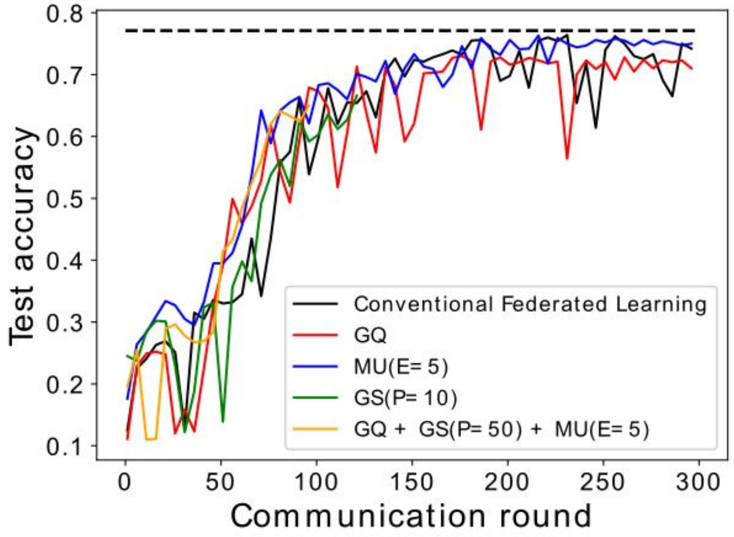
Combinations



2CFNN - FMNIST



3CFNN - CIFAR-10



VGG16 - CCH

Results - Bandwidth usage

Approach \ Target accuracy	0.84	0.86	0.88	0.90	0.92
P=10	665.35 | 642.48 | 3.44	1064.56 | 1156.46 | -8.63	1996.04 | 1927.43 | 3.44	3725.95 | 4240.35 | -13.81	11177.85 | 11179.10 | -0.01
P=50	665.35 | 509.41 | 23.44	1064.56 | 815.05 | 23.44	1996.04 | 1528.22 | 23.44	3725.95 | 2852.68 | 23.44	11177.85 | 7437.35 | 33.46
P=90	665.35 | 376.34 | 43.44	1064.56 | 677.41 | 36.37	1996.04 | 1053.75 | 47.21	3725.95 | 1655.89 | 55.56	11177.85 | 3989.18 | 64.31
E=2	665.35 | 399.21 | 40.00	1064.56 | 532.28 | 50.00	1996.04 | 1064.56 | 46.67	3725.95 | 1996.04 | 46.43	11177.85 | 7318.83 | 34.52
E=10	665.35 | 133.07 | 80.00	1064.56 | 133.07 | 87.50	1996.04 | 266.14 | 86.67	3725.95 | 665.35 | 82.14	-
E=20	665.35 | 133.07 | 80.00	1064.56 | 133.07 | 87.50	1996.04 | 266.14 | 86.67	3725.95 | 399.21 | 89.29	11177.85 | 9181.80 | 17.86
GQ	665.35 | 399.21 | 40.00	1064.56 | 1663.37 | -56.25	-	-	-
GQ + GS(P=50) + MU(E=10)	665.35 | 50.01 | 92.48	1064.56 | 100.01 | 90.61	1996.04 | 150.02 | 92.48	3725.95 | 200.02 | 94.63	-

Table 5.1: *2CFNN-FMNIST*: Bandwidth usage by the conventional federated training (in MB) | Bandwidth usage by the approach (in MB) | Bandwidth saving of the approach (in percent) to reach the target accuracy

Results - Bandwidth usage

Approach \ Target accuracy	0.5	0.6	0.68	0.7	0.72	0.733
P=10	7112.73 | 6868.26 | 3.44	13435.16 | 12973.36 | 3.44	23709.11 | 20604.73 | 13.09	26870.32 | 24420.42 | 9.12	36353.96 | 30525.52 | 16.03	60063.07 | 54182.78 | 9.79
P=50	7112.73 | 5445.69 | 23.44	13435.16 | 9076.15 | 32.44	23709.11 | 15126.91 | 36.20	26870.32 | 17547.22 | 34.70	36353.96 | 19967.52 | 45.07	60063.07 | 25413.21 | 57.69
P=90	7112.73 | 6655.84 | 6.42	13435.16 | 10891.38 | 18.93	23709.11 | 16337.06 | 31.09	26870.32 | 18757.37 | 30.19	-	-
E=2	7112.73 | 4741.82 | 33.33	13435.16 | 9483.64 | 29.41	23709.11 | 15806.07 | 33.33	26870.32 | 18967.28 | 29.41	36353.96 | 26080.02 | 28.26	-
E=10	7112.73 | 1580.61 | 77.78	13435.16 | 2370.91 | 82.35	23709.11 | 6322.43 | 73.33	26870.32 | 14225.46 | 47.06	-	-
E=20	7112.73 | 790.30 | 88.89	13435.16 | 1580.61 | 88.24	23709.11 | 11854.55 | 50.00	-	-	-
GQ	7112.73 | 3556.37 | 50.00	13435.16 | 6717.58 | 50.00	23709.11 | 11064.25 | 53.33	26870.32 | 13040.01 | 51.47	36353.96 | 16201.22 | 55.43	60063.07 | 19757.59 | 67.11
GQ + GS(P=50) + MU(E=2)	7112.73 | 1781.89 | 74.95	13435.16 | 3563.78 | 73.47	23709.11 | 5939.63 | 74.95	26870.32 | 6830.57 | 74.58	36353.96 | 10097.37 | 72.22	-

Table 5.2: *3CFNN-CIFAR-10*: Bandwidth usage by the conventional federated training (in MB) | Bandwidth usage by the approach (in MB) | Bandwidth saving of the approach (in percent) to reach the target accuracy

Results - Bandwidth usage

Technische Universität München

Approach \ Target accuracy	0.55	0.65	0.7	0.72	0.76	0.771
P=10	205.68 | 202.92 | 1.34	236.92 | 262.94 | -10.98	-	-	-	-
P=90	205.68 | 164.93 | 19.81	-	-	-	-	-
E=2	205.68 | 145.80 | 29.11	236.92 | 221.30 | 6.59	343.66 | 27597 | 19.70	367.09 | 328.04 | 10.64	466.03 | 377506 36 | 18.99	643.06 | 387.92 | 39.68
E=5	205.68 | 166.62 | 18.99	236.92 | 221.30 | 6.59	343.66 | 294.20 | 14.39	367.09 | 333.25 | 9.22	466.03 | 557.15 | -19.55	643.06 | 577.98 | 10.12
E=10	205.68 | 46.86 | 77.22	236.92 | 78.10 | 67.03	343.66 | 91.12 | 73.48	367.09 | 124.97 | 65.96	-	-
E=20	205.68 | 41.66 | 79.75	236.92 | 67.70 | 71.43	343.66 | 119.76 | 65.15	367.09 | 130.17 | 64.54	-	-
GQ	205.68 | 91.12 | 55.70	236.92 | 100.23 | 57.69	343.66 | 135.38 | 60.61	367.09 | 184.85 | 49.65	-	-
GQ + GS(P=50) + MU(E=5)	205.68 | 76.06 | 63.02	236.92 | 102.53 | 56.72	-	-	-	-

Table 5.3: *VGG16-CCH*: Bandwidth usage by the conventional federated training (in GB) | Bandwidth usage by the approach (in GB) | Bandwidth saving of the approach (in percent) to reach the target accuracy

Summary

- **All three approaches** can **significantly save** the network **bandwidth** but also affect the accuracy of the models
- The **More local update** approach is the **most suitable** choice taking communication **efficiency and accuracy** into account
- The **More local update** approach is the **most contributing** approach in the **combination**
- **Combining** the approaches **provides further bandwidth savings** in comparison with each individual approach

Outlook

- Gradient sparsification
  - Multi-threaded implementation (one thread per layer) to improve the sparsification speed
  - Server-side sparsification
- Communication-efficient approaches in
  - Non-IID label distributions
  - Imbalanced sample size distributions