FedConv: A Learning-on-Model Paradigm for Heterogeneous Federated Clients

Leming Shen¹, Qiang Yang¹.², Kaiyan Cui¹.³, Yuanqing Zheng¹, Xiao-Yong Wei⁴.¹, Jianwei Liu⁵, Jinsong Han⁵

¹The Hong Kong Polytechnic University, ²University of Cambridge,
³Nanjing University of Posts and Telecommunications, ⁴Sichuan University, ⁵Zhejiang University
Federated Learning (FL)

- Collaboratively train a global model
- Without transmitting private data
Model Heterogeneity in FL

• Mobile devices have *diverse system resources*.
• Smallest affordable model → performance ↓
Existing Solution: Parameter Sharing
Existing Solution: Parameter Sharing

- **Imbalanced Training** (Fixed sharing portion)
  - Larger models miss the information from other clients.
Existing Solution: Parameter Sharing

- **Imbalanced Training** *(Fixed sharing portion)*
  - Larger models miss the information from other clients.
Existing Solution: Parameter Sharing

- **Imbalanced Training** (Fixed sharing portion)
  - Larger models miss the information from other clients.

- Unshared
  - Smaller models perform better
  - The global model exhibits instability and even performs worse
Existing Solutions: Model Pruning

Channel-Level Pruning\(^1\)
- Remove entire channels
- Less input data

Filter-Level Pruning\(^2\)
- Remove entire filters
- Less output feature maps

---

Existing Solutions: Model Pruning

- Information Loss & Extra Overhead
  - Remove entire channels or filters
  - Pruning performed by the client
Existing Solutions: Model Pruning

- Information Loss & Extra Overhead
  - Remove entire channels or filters
  - Pruning performed by the client
Ideally for Sub-model Generation…

1. Minimize the information loss
2. Retain the performance
3. No extra overhead on clients
Ideally for Sub-model Generation...

1. Minimize the information loss
2. Retain the performance
3. No extra overhead on clients

Convolution
Insight

• Convolution can extract effective features from input images

• We can also use it to **extract crucial parameter information**
Convolutional Compression

Global model

Conv1

Input
Convolutional Compression

Global model
Conv1

Input

16
Convolutional Compression

Input

Global model

Conv1

16

Global model Conv2
(kernels 32x16@3x3)
Convolutional Compression

Input → Conv1 → Global model Conv2 (kernels 32x16@3x3) → 32 feature maps → Deer
Convolutional Compression

Shrinkage Ratio = 0.75
Convolutional Compression

Input

Global model
Conv1

Sub-model
Conv1

Global model Conv2
(kernels 32x16@3x3)

32 feature maps

Deer

Shrinkage Ratio = 0.75
Convolutional Compression

Input

Sub-model

Conv1

Global model

Conv1

Global model Conv2 (kernels 32x16@3x3)

Sub-model Conv2 (kernels 24x12@3x3)

Deer

Deer

Shrinkage Ratio = 0.75
Convolutional Compression

Input

Convolution

Global model Conv1

Sub-model Conv1

Shrinkage Ratio = 0.75

Global model Conv2 (kernels 32x16@3x3)

Sub-model Conv2 (kernels 24x12@3x3)

32 feature maps

24 feature maps

Deer

Deer
Convolutional Compression

Input

Convolution

Conv1

Global model Conv1

Shrinkage Ratio = 0.75

Global model Conv2
(kernels 32x16@3x3)

Sub-model Conv2
(kernels 24x12@3x3)

Deer

32 feature maps

24 feature maps
Convolutional Compression

Global model

Conv1

Sub-model

Conv1

Convolution Operation 2

Global model Conv2 (kernels 32x16@3x3)

Sub-model Conv2 (kernels 24x12@3x3)

Shrinkage Ratio = 0.75

Input

feature maps

Deer

feature maps

Deer
Convolutional Compression

Input

Global model

Conv1

Convolution

16

Sub-model

Conv1

Shrinkage Ratio = 0.75

Global model Conv2
(kernels 32x16@3x3)

Convolution Operation 2

Sub-model Conv2
(kernels 24x12@3x3)

24 feature maps

Deer

Accuracy (%)

Mutual Information (I(X, Z))

85
80
75
70

85
80
75
70

3.6
3.2
2.8
2.4

Accuracy (I)

Pretrain
Channel Different
Filter
Conv

I(X, Z)

Deer

THE HONG KONG
POLYTECHNICAL UNIVERSITY
香港理工大學
Convolutional Compression

• How to determine the size of the compressed model?
• Shrinkage Ratio = 0.75
Convolutional Compression (Cont.)

• How to retain performance?
• A learning-on-model paradigm

- Learning-on-data: raw data as input
- Learning-on-model: model parameters as input

Perform at the server
System Overview – FedConv

Central Server
① Initialization
Global Model

Heterogeneous Clients

THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大学
System Overview – FedConv

Central Server

1. Initialization

Global Model

Convolutional Compression

2. Compression

Conv Parameters

Sub-models

Shrinkage Ratios

Heterogeneous Clients
System Overview – FedConv

Central Server

1. **Initialization**
   - Global Model

2. **Compression**
   - Conv Parameters
   - Sub-models

3. **Sending**

Convolutional Compression

Shrinkage Ratios

Heterogeneous Clients

- Graphics card
- Tablet
- Mobile phone
- Smartwatch
System Overview – FedConv

Central Server

① Initialization

Global Model

Convolutional Compression

② Compression

Conv Parameters

Sub-models

Shrinkage Ratios

③ Sending

Heterogeneous Clients

④ Local Training

Global Model

Conv Parameters

Sub-models

Heterogeneous Clients

Local Training
System Overview – FedConv

Central Server

1. Initialization

Global Model

Convolutional Compression

2. Compression

Conv Parameters

Sub-models

Shrinkage Ratios

3. Sending

Heterogeneous Clients

4. Local Training

5. Uploading
System Overview – FedConv

Central Server

1. Initialization
   - Global Model

Convolutional Compression

2. Compression
   - Conv Parameters
   - Sub-models

Shrinkage Ratios

3. Sending

Transposed Convolutional Dilation

4. Local Training

Heterogeneous Clients

5. Uploading

6. Dilation
   - TC Parameters
   - Sub-models

Dilated Models
System Overview – FedConv

1. **Initialization**
   - Global Model

2. **Compression**
   - Conv Parameters
   - Shrinkage Ratios
   - Convolutional Compression
   - Sub-models

3. **Sending**
   - Sending

4. **Local Training**
   - Uploading
   - Heterogeneous Clients
   - Dilated Models
   - Weight Vectors
   - Weighted Average Aggregation

5. **Dilation**
   - TC Parameters
   - Transposed Convolutional Dilation
   - Sub-models

6. **Aggregation**
   - Global Model
   - Dilated Models
   - Weight Vectors

7. **Weighted Average Aggregation**
   - Central Server

---

THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學
Experiment Setup

• Hardware

<table>
<thead>
<tr>
<th>Type</th>
<th>Device Name</th>
<th>Number</th>
<th>CPU</th>
<th>RAM</th>
<th>GPU</th>
<th>GDDR</th>
<th>Network</th>
<th>SR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>ASUS W790-ACE Server</td>
<td>1</td>
<td>Intel Xeon Gold 6248R, 3.0GHz</td>
<td>640GB</td>
<td>NVIDIA A100</td>
<td>40GB</td>
<td>Ethernet</td>
<td>-</td>
</tr>
<tr>
<td>Router</td>
<td>Mi Router AX3000</td>
<td>1</td>
<td>Qualcomm IPQ5000 A53, 1.0GHz</td>
<td>256MB</td>
<td>-</td>
<td>-</td>
<td>Ethernet</td>
<td>-</td>
</tr>
<tr>
<td>PC</td>
<td>Supermicro X11SCA-F</td>
<td>2</td>
<td>Intel Xeon E-2236, 3.4GHz</td>
<td>32GB</td>
<td>NVIDIA RTX A4000</td>
<td>16GB</td>
<td>Ethernet</td>
<td>1.0</td>
</tr>
<tr>
<td>PC</td>
<td>Supermicro SYS-5038A-I</td>
<td>2</td>
<td>Intel Xeon E5-2620 v4, 2.10GHz</td>
<td>64GB</td>
<td>NVIDIA GeForce GTX 1080 Ti</td>
<td>12GB * 2</td>
<td>Wi-Fi</td>
<td>1.0</td>
</tr>
<tr>
<td>PC</td>
<td>ThinkPad P52s Laptop</td>
<td>4</td>
<td>Intel i5-8350U, 1.70GHz</td>
<td>32GB</td>
<td>NVIDIA Quadro P500</td>
<td>2GB</td>
<td>Wi-Fi</td>
<td>0.75</td>
</tr>
<tr>
<td>Board</td>
<td>NVIDIA Jetson TX2</td>
<td>4</td>
<td>Dual-Core NVIDIA Denver 2, 2GHz</td>
<td>8GB</td>
<td>256-core NVIDIA Pascal GPU</td>
<td>4GB</td>
<td>Wi-Fi</td>
<td>0.75</td>
</tr>
<tr>
<td>Board</td>
<td>NVIDIA Jetson Nano</td>
<td>4</td>
<td>ARM Cortex-A57 MPCore, 1.5 GHz</td>
<td>4GB</td>
<td>NVIDIA Maxwell architecture GPU</td>
<td>2GB</td>
<td>Wi-Fi</td>
<td>0.5</td>
</tr>
<tr>
<td>Board</td>
<td>Raspberry Pi 4</td>
<td>4</td>
<td>Quad core Cortex-A72, 1.8GHz</td>
<td>8GB</td>
<td>-</td>
<td>-</td>
<td>Wi-Fi</td>
<td>0.25</td>
</tr>
</tbody>
</table>

• Software

- NN framework: PyTorch (we modify its package to enable back-propagation of the gradient to update convolution parameters)
- FL framework: Flower
Experiment Setup (Cont.)

• Datasets & Models
  • Image Classification
    • MNIST: handwritten digits ---- CNN
    • CIFAR10: color images ---- ResNet18
    • CINIC10: color images ---- GoogLeNet
  
  • Human Activity Recognition (HAR) ---- CNN
    • WiAR: WIFI CSI data
    • Depth camera dataset: gray-scale depth images
    • HARBox: 9-axis IMU data
Experiment Setup (Cont.)

- Baselines
  - Serveralone: trains one model with only server-side data
  - Standalone: each client separately trains their local models
  - FedAvg: averages the model parameters
  - FedMD: a knowledge distillation-based method
  - LotterFL: uses Lottery Ticket hypothesis to generate heterogeneous models
  - Hermes: applies channel-level pruning
  - TailorFL: applies filter-level pruning
  - HeteroFL: static parameter sharing scheme
  - FedRolex: dynamic parameter sharing scheme
Evaluation – Metrics

• Training Performance
  • Inference accuracy
    • Generalization: global model accuracy on global dataset
    • Personalization: client model accuracy on client dataset
  • Communication cost

• Runtime Performance
  • Memory footprint: CPU + GPU memory usage
  • Wall-clock time: total execution time of each client
Evaluation – Overall Performance

• Global model & client model performance
Evaluation – Overall Performance

- Global model & client model performance

The superior generalization performance of FedConv

The personalization performance of FedConv
Evaluation – Overall Performance

• Global model & client model performance (Cont.)

![Graphs showing global and client model performance](image)

**Figure 10:** The inference accuracy of aggregated global models and client models on different datasets.
Evaluation – Overall Performance

• Global model & client model performance (Cont.)

- FedConv is more robust to heterogeneous data distribution.
- The performance gain of FedConv becomes more significant with more heterogeneous data distribution.
- We can further enhance FedConv with personalization methods (e.g., adding task-specific layers)

Figure 10: The inference accuracy of aggregated global models and client models on different datasets.
## System Overhead

### Table 2: System resource overhead.

<table>
<thead>
<tr>
<th>Metric</th>
<th>System</th>
<th>Heterogeneous Data ($\alpha = 0.05$)</th>
<th>Homogeneous Data ($\alpha = 10000$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>MNIST</td>
<td>CIFAR10</td>
</tr>
<tr>
<td>Memory Footprint (CPU + GPU (GB))</td>
<td>Standalone</td>
<td>2.14</td>
<td>3.51</td>
</tr>
<tr>
<td></td>
<td>FedAvg</td>
<td>1.90</td>
<td>2.40</td>
</tr>
<tr>
<td></td>
<td>FedMD</td>
<td>2.71</td>
<td>3.65</td>
</tr>
<tr>
<td></td>
<td>LotteryFL</td>
<td>2.62</td>
<td>3.51</td>
</tr>
<tr>
<td></td>
<td>Hermes</td>
<td>2.64</td>
<td>3.45</td>
</tr>
<tr>
<td></td>
<td>TailorFL</td>
<td>2.75</td>
<td>3.61</td>
</tr>
<tr>
<td></td>
<td>HeteroFL</td>
<td>2.63</td>
<td>3.31</td>
</tr>
<tr>
<td></td>
<td>FedRlbnk</td>
<td>2.63</td>
<td>3.21</td>
</tr>
<tr>
<td><strong>FedConv</strong></td>
<td></td>
<td>2.52</td>
<td>3.21</td>
</tr>
<tr>
<td>Wall-clock Time (s)</td>
<td>Standalone</td>
<td>3.87</td>
<td>24.65</td>
</tr>
<tr>
<td></td>
<td>FedMD</td>
<td>44.34</td>
<td>437.14</td>
</tr>
<tr>
<td></td>
<td>LotteryFL</td>
<td>9.18</td>
<td>147.98</td>
</tr>
<tr>
<td></td>
<td>Hermes</td>
<td>43.22</td>
<td>714.00</td>
</tr>
<tr>
<td></td>
<td>TailorFL</td>
<td>6.98</td>
<td>62.89</td>
</tr>
<tr>
<td></td>
<td>HeteroFL</td>
<td>6.96</td>
<td>42.56</td>
</tr>
<tr>
<td></td>
<td>FedRlbnk</td>
<td>6.92</td>
<td>45.98</td>
</tr>
<tr>
<td><strong>FedConv</strong></td>
<td></td>
<td>5.96</td>
<td>40.68</td>
</tr>
</tbody>
</table>
Evaluation – Overall Performance (Cont.)

• System Overhead – Communication Cost

<table>
<thead>
<tr>
<th>System</th>
<th>MNIST</th>
<th>CIFAR10</th>
<th>CINIC10</th>
<th>WiAR</th>
<th>DCD</th>
<th>HARBox</th>
</tr>
</thead>
<tbody>
<tr>
<td>FedAvg</td>
<td>14.80</td>
<td>4815.84</td>
<td>2697.85</td>
<td>28.24</td>
<td>13.45</td>
<td>8.87</td>
</tr>
<tr>
<td>FedMD</td>
<td>19.99</td>
<td>5126.46</td>
<td>2859.79</td>
<td>40.91</td>
<td>19.94</td>
<td>16.24</td>
</tr>
<tr>
<td>LotteryFL</td>
<td>11.11</td>
<td>4713.91</td>
<td>2623.93</td>
<td>23.01</td>
<td>10.05</td>
<td>8.55</td>
</tr>
<tr>
<td>Hermes</td>
<td>16.34</td>
<td>7099.66</td>
<td>2848.83</td>
<td>36.63</td>
<td>15.02</td>
<td>12.95</td>
</tr>
<tr>
<td>TailorFL</td>
<td>11.40</td>
<td>4787.18</td>
<td>2686.15</td>
<td>24.30</td>
<td>10.32</td>
<td>8.82</td>
</tr>
<tr>
<td>HeteroFL</td>
<td>11.11</td>
<td>4713.91</td>
<td>2623.93</td>
<td>23.01</td>
<td>10.05</td>
<td>8.55</td>
</tr>
<tr>
<td>FedRolex</td>
<td>11.11</td>
<td>4713.91</td>
<td>2623.93</td>
<td>23.01</td>
<td>10.05</td>
<td>8.55</td>
</tr>
<tr>
<td><strong>FedConv</strong></td>
<td><strong>11.11</strong></td>
<td><strong>4713.91</strong></td>
<td><strong>2623.93</strong></td>
<td><strong>23.01</strong></td>
<td><strong>10.05</strong></td>
<td><strong>8.55</strong></td>
</tr>
</tbody>
</table>

Table 3: Communication overhead comparison (GB).
Conclusion

• We propose FedConv, a client-friendly federated learning framework for heterogeneous clients, aiming to minimize the system overhead on resource-constrained mobile devices.

• FedConv features three key technical modules: convolutional compression, TC dilation, and weighted average aggregation.

• We believe the proposed learning-on-model paradigm is worthy of further exploration (e.g., configuration optimization).
Thanks for Listening!

- FedConv: A Learning-on-Model Paradigm for Heterogeneous Federated Clients

- Leming Shen, Qiang Yang, Kaiyan Cui, Yuanqing Zheng, Xiao-Yong Wei, Jianwei Liu, Jinsong Han