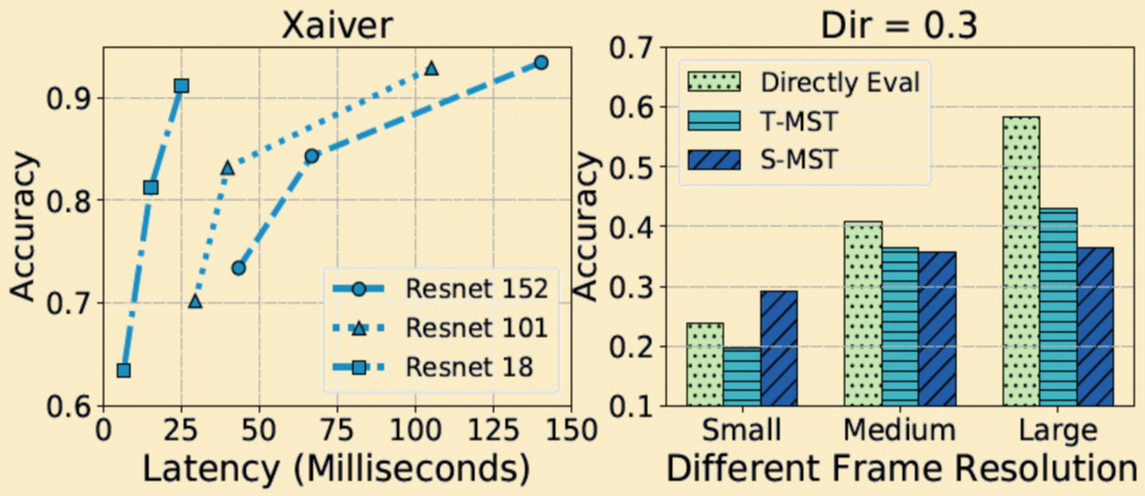
ArtFL: Exploiting Data Resolution in Federated Learning for Dynamic Runtime Inference via Multi-Scale Training

Siyang Jiang, Xian Shuai, Guoliang Xing
Department of Information Engineering, The Chinese University of Hong Kong

Introduction

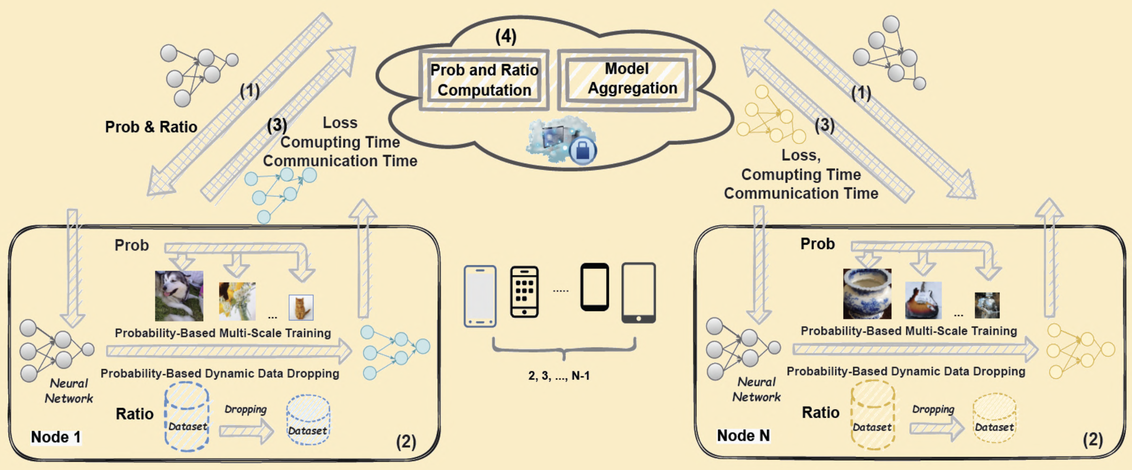
- Existing FL systems have not adequately addressed the dynamic real-time requirements of mission-critical applications such as autonomous driving and smart health due to stringent inference deadlines and resource limitations on edge devices.
- In this work, we propose, ArtFL, a novel federated learning system designed to support dynamic runtime inference through multi-scale training. In particular, we initially propose data-utility-based multi-scale training, allowing the trained model to process data of varying resolutions during inference.
- Next, we introduce an innovative strategy for frame resolution selection in inference, based on the similarity of adjacent frames. Then, leveraging latency-based dynamic data dropping, we propose a systematic scheme to reduce the overall training time by shortening the waiting time in FL.
- Lastly, we build two real-world FL testbeds for smart vehicles and healthcare applications, utilizing a heterogeneous edge platform for evaluation ArtFL.

Motivation



- The data resolution provides models significant room for the trade-off between accuracy and real-time performance.
- Direct inference on different data resolutions under conventional training can drastically undermine accuracy due to resolution discrepancy between training and inference.

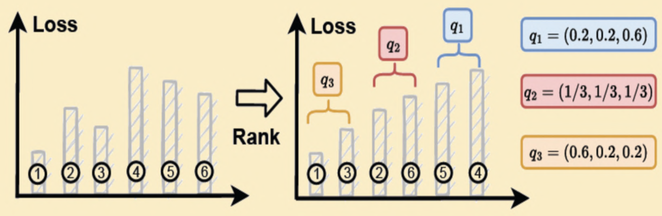
Workflow



- First, clients check in to the server and then receive the global model and the metadata from the server, including the probabilities on different resolutions and the dropping ratio for training data.
- Next, guided by the metadata, clients locally update the received global model using multi-scale training with the clipped dataset.
- Then, once the local training is finished, clients upload the well-trained model and report the training time consumption and loss.
- Last, the server aggregates the model and calculates the metadata for each client for the following FL round. Each communication round consists of the above steps until convergence.

Methodology

- **Data-Utility-Based Multi-Scale Training**: The key idea of is that we want to enable the flexibility of dynamic requirements in inference time. Therefore, we consider the training and inference as two closely coupled stages.
- **Optimal Frame Resolution Selection**: To choose the right frame for rescale during inference stage, we propose optimal frame resolution selection to choose the optimal resolution for each frame according to the similarity of two adjacent frames.

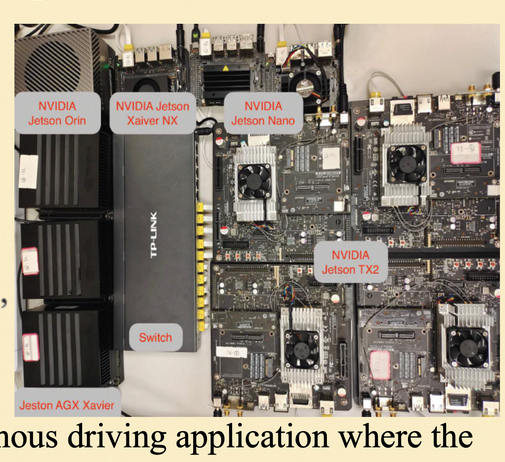


$$\max_{s_i \in S} \sum_{i}^{J} a_i s_i, \quad s.t. \begin{cases} \sum_i^J t_i s_i \le T \\ \sum_i^J s_i = 1 \end{cases}$$

- **Latency-Based Dynamic Data Dropping**: since the waiting time is prolonged in the training stage due to the multi-scale training, we propose latency-based dynamic data dropping aims to reduce the training duration.

$$T_{E\_Train}^{(i,t)} = \frac{\mathbf{p}^{(i,t)} \otimes \mathbf{I}}{\mathbf{p}^{(i,t-1)} \otimes \mathbf{I}} \cdot \frac{1}{Dr^{(i,t-1)}} \cdot T_{Train}^{(i,t-1)},$$

- **Implementation**



- **Smart Vehicle Testbed**: we simulate an autonomous driving application where the task is to recognize the traffic signs in real time (≥ 30 fps).

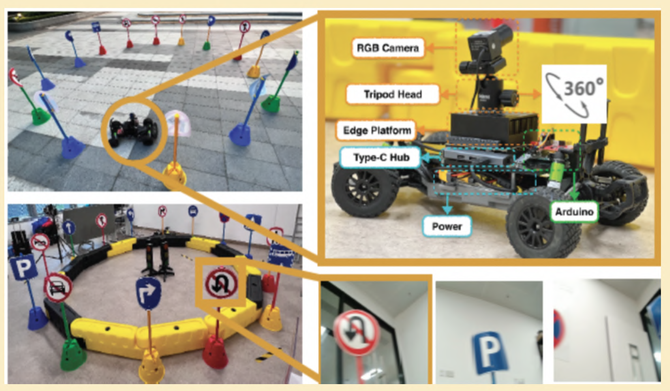


Table 1: A summary of two platforms.

Platform	Details
Edge Device Platform	Jetson Orin * 1. Cores of CPU/GPU: 12/1792. DRAM: 32G
	AGX Xavier * 3. Cores of CPU/GPU: 8/512, DRAM: 16G
	Xaiver NX * 18. Cores of CPU/GPU: 6/384. DRAM: 8G.
	Jetson TX2 * 6. Cores of CPU/GPU: 6/256. DRAM: 8G.
	Jetson Nano * 2. Cores of CPU/GPU: 4/128. DRAM: 4G.
Cloud Sever	CPU: i9-9820X. GPU: NVIDIA A100 * 4. DRAM: 512G.

- **Smart Health Testbed**: This testbed simulates the scenario of patient monitoring of two places: the hospital (sick room) and home (bedroom), where the key difference between the two places is the camera's positions and views.

Experimental Results

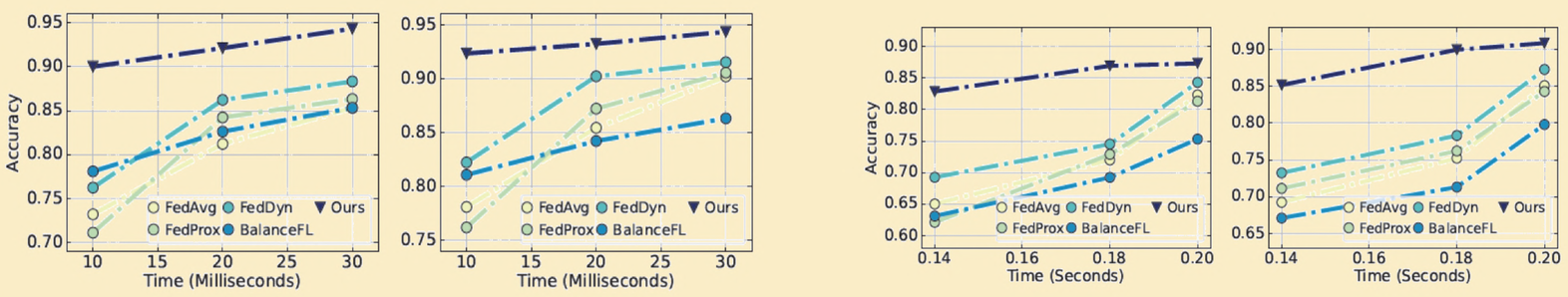


Table 2: Accuracy Comparison under Real-Time Constraint.

Testbeds	Runtime Status	FlexDNN[12]	RANet[57]	Ours
Smart Vehicle	Peak	0.43	0.44	**0.93**
Smart Health		0.39	0.35	**0.81**
Smart Vehicle	Spare	0.58	0.51	**0.95**
Smart Health		0.47	0.43	**0.87**

- ArtFL performs well in both peak and spare time. When the time bound is tight, baseline performance degrades considerably, while ArtFL maintains relatively consistent performance thanks to the multi-scale training.
- As shown in Table 2, ArtFL outperforms both baselines by a large margin under the time constraints, The main reason is that the data in the real world is not as clean as in public datasets, with many hard examples, which is not friendly to this early exit