University of St Andrews

PipeLearn: Pipeline Parallelism for Collaborative Machine Learning Systems

Zihan Zhang and Blesson Varghese

zz66@st-andrews.ac.uk

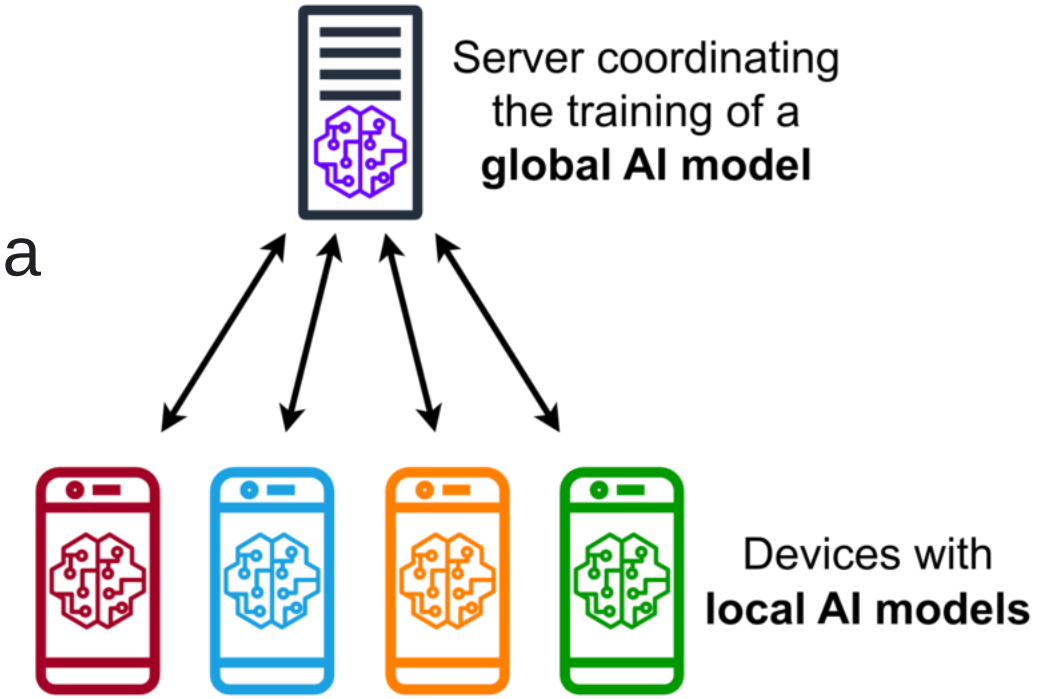
Seventh Annual UK System Research Challenges Workshop

Collaborative Machine Learning Systems

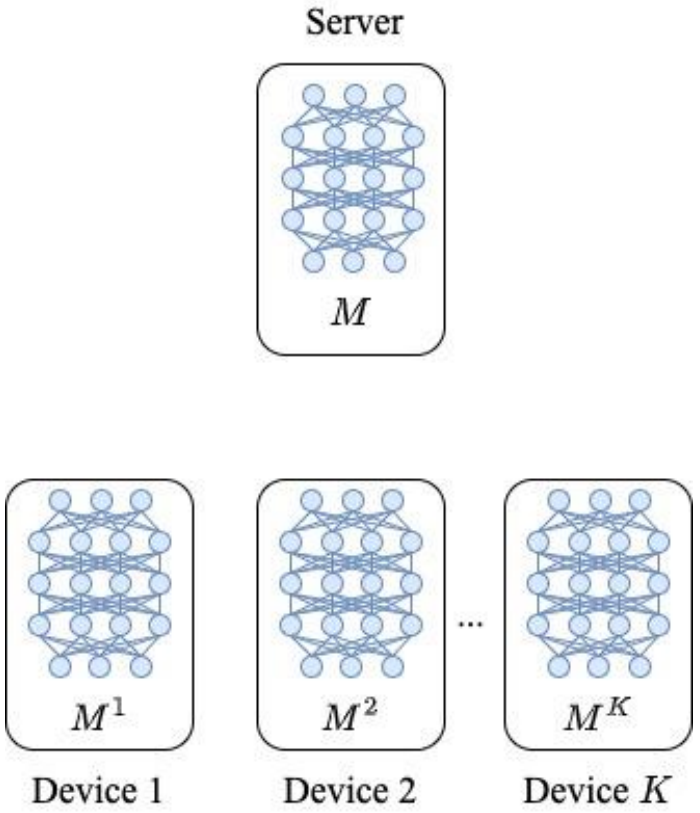
Collaborative machine learning (CML) systems were proposed to collaboratively train deep learning models using multiple devices and a server.

Three popular techniques:

- Federated Learning
- Split Learning
- Split Federated Learning



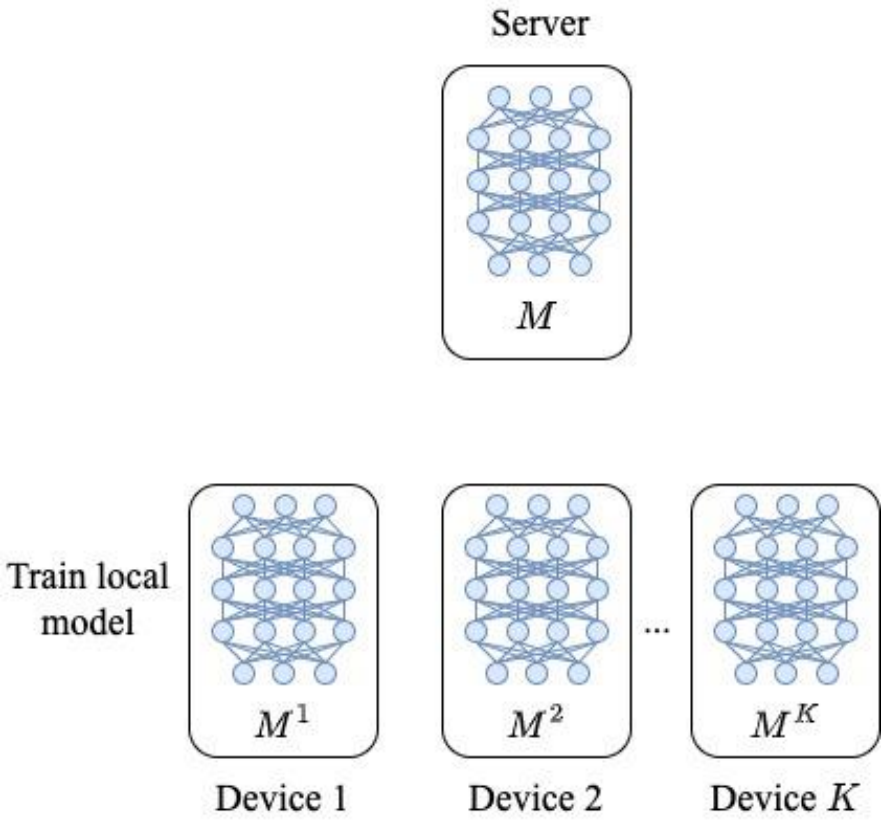
Federated Learning (FL)



- Problem:

The server resources are only employed when the local models are aggregated and remains idle for the remaining time.

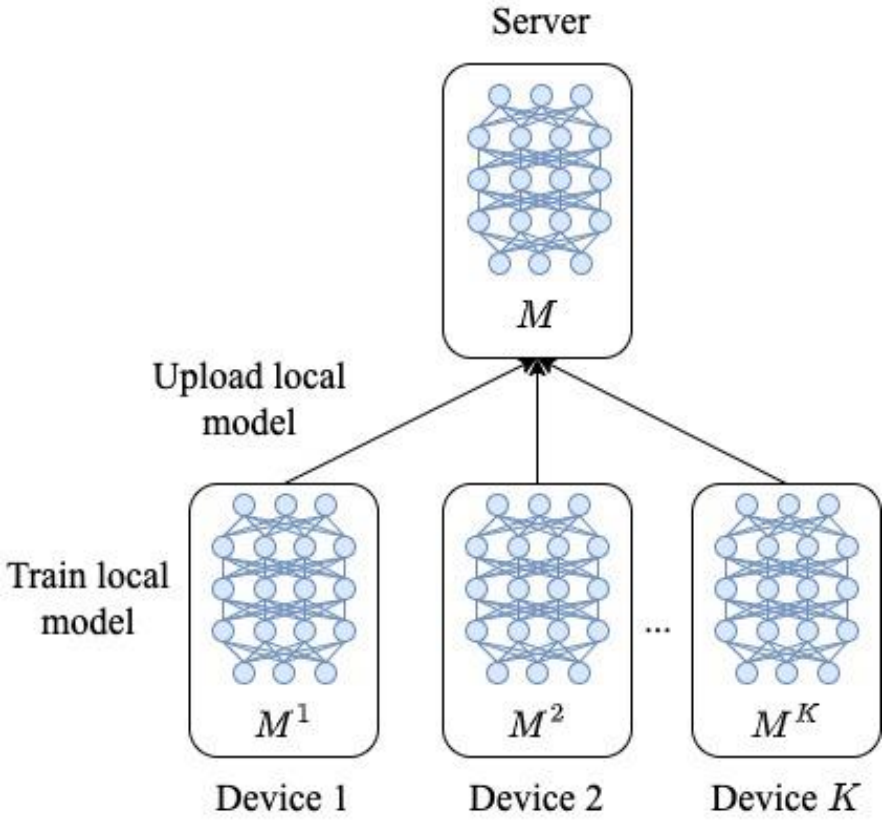
Federated Learning (FL)



- Problem:

The server resources are only employed when the local models are aggregated and remains idle for the remaining time.

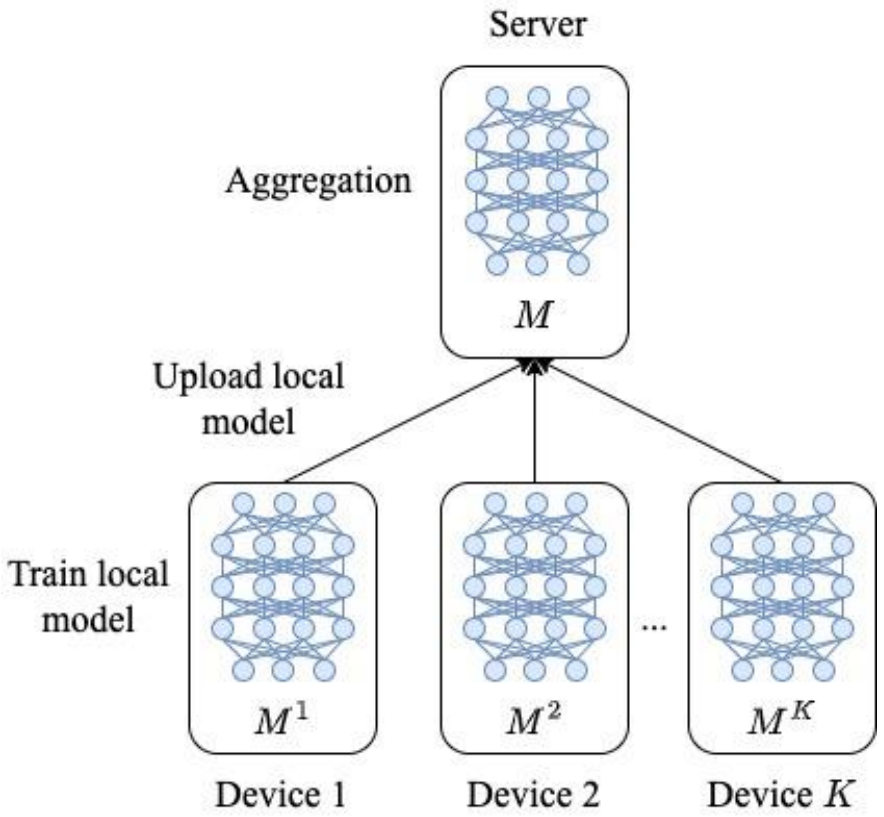
Federated Learning (FL)



- Problem:

The server resources are only employed when the local models are aggregated and remains idle for the remaining time.

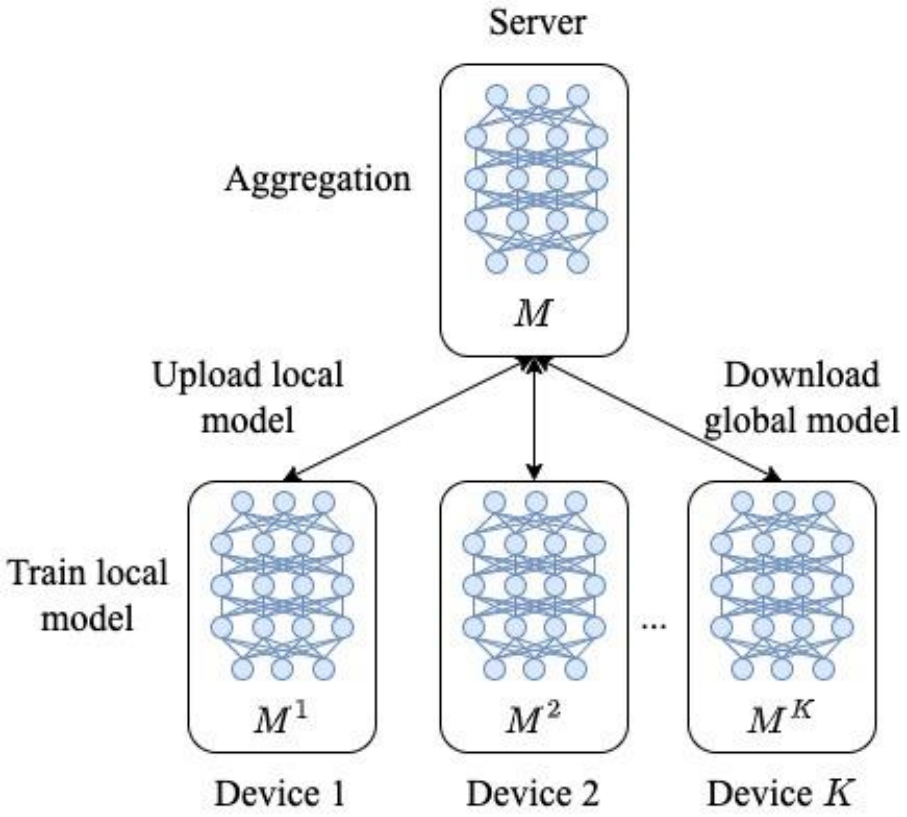
Federated Learning (FL)



- Problem:

The server resources are only employed when the local models are aggregated and remains idle for the remaining time.

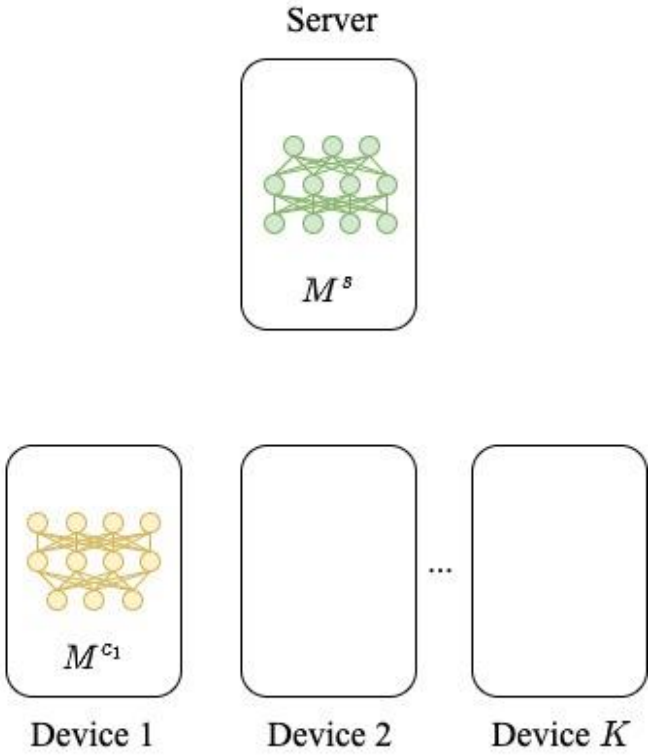
Federated Learning (FL)



- Problem:

The server resources are only employed when the local models are aggregated and remains idle for the remaining time.

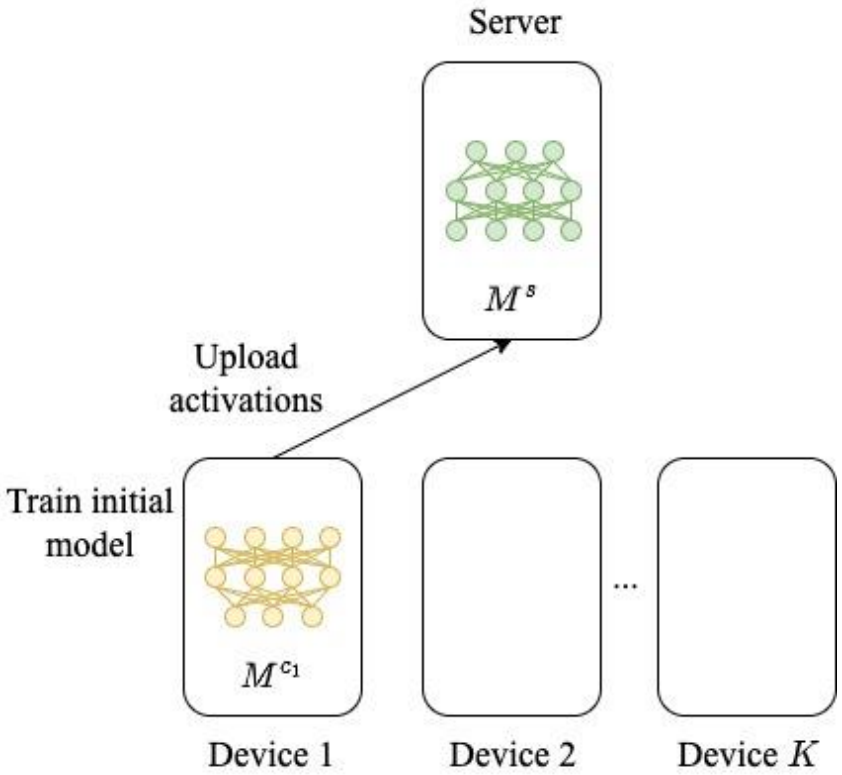
Split Learning (SL)



- Problem:

Only one device or the server will utilise its resources while the other devices or server are idle.

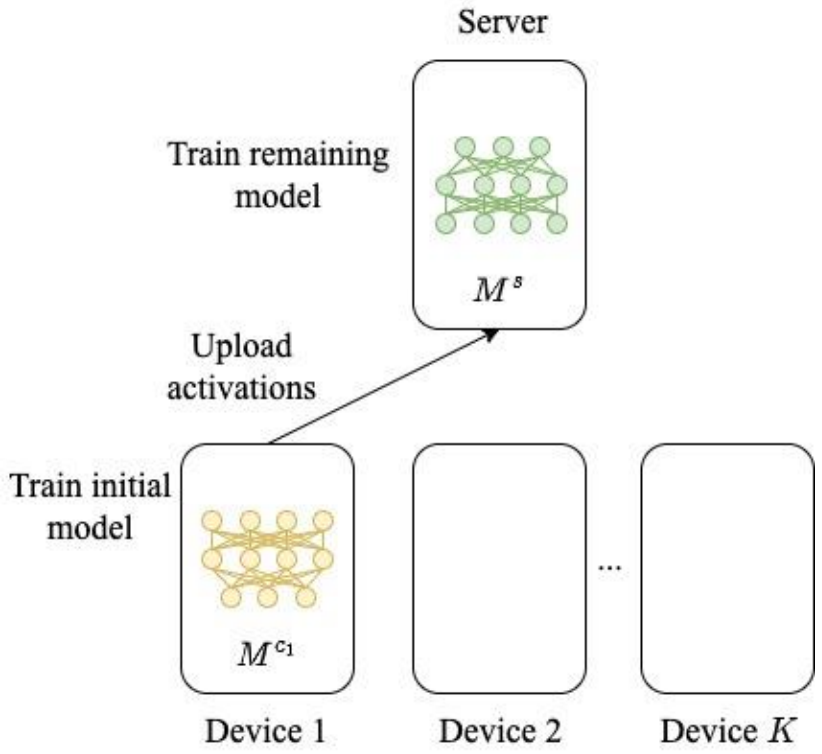
Split Learning (SL)



- Problem:

Only one device or the server will utilise its resources while the other devices or server are idle.

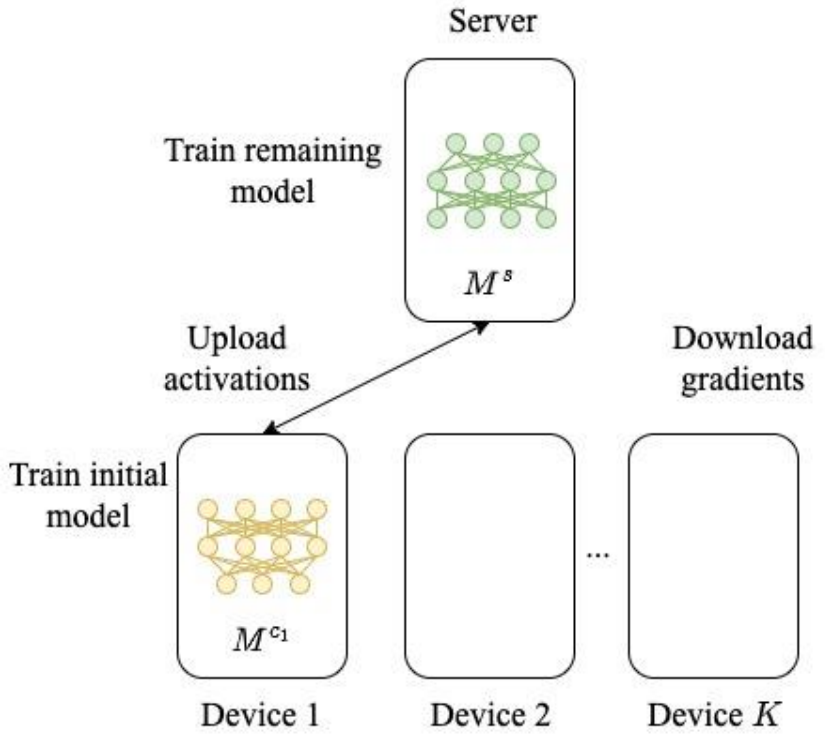
Split Learning (SL)



- Problem:

Only one device or the server will utilise its resources while the other devices or server are idle.

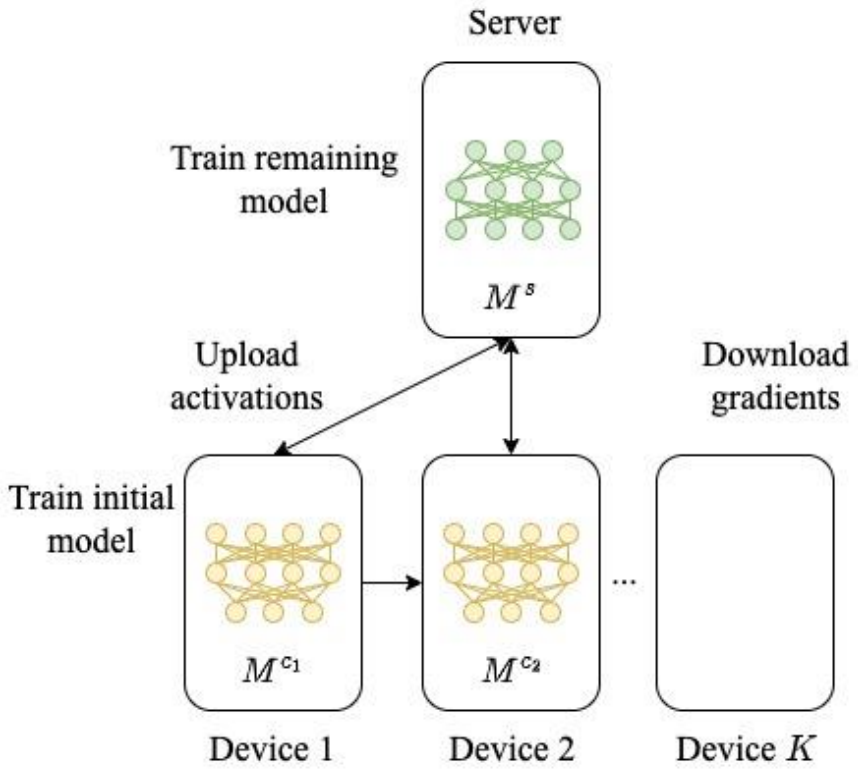
Split Learning (SL)



- Problem:

Only one device or the server will utilise its resources while the other devices or server are idle.

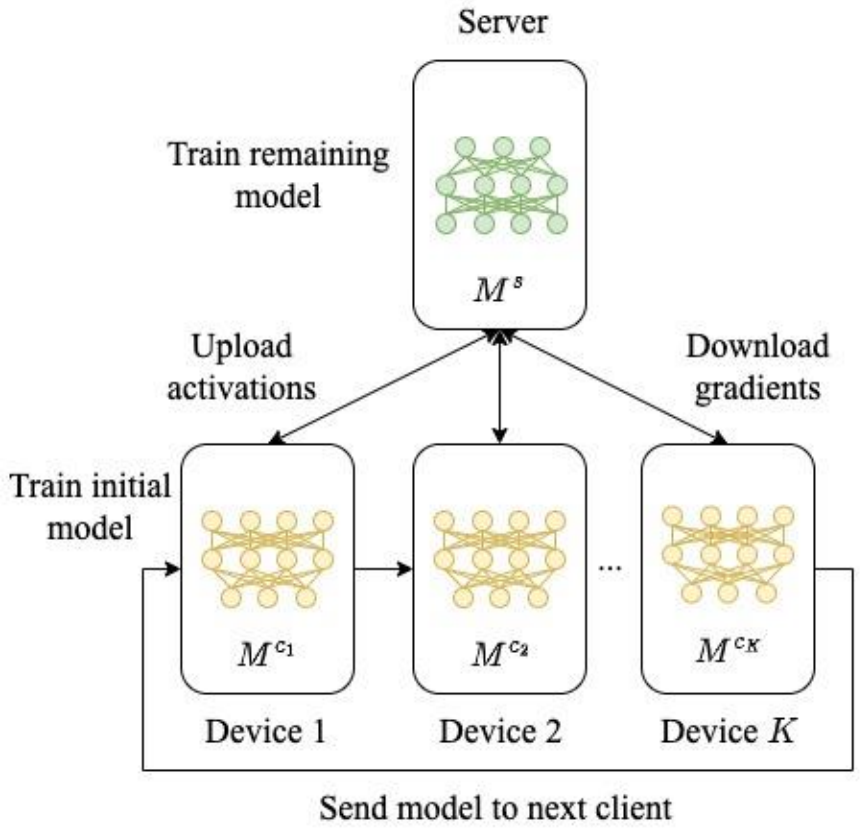
Split Learning (SL)



- Problem:

Only one device or the server will utilise its resources while the other devices or server are idle.

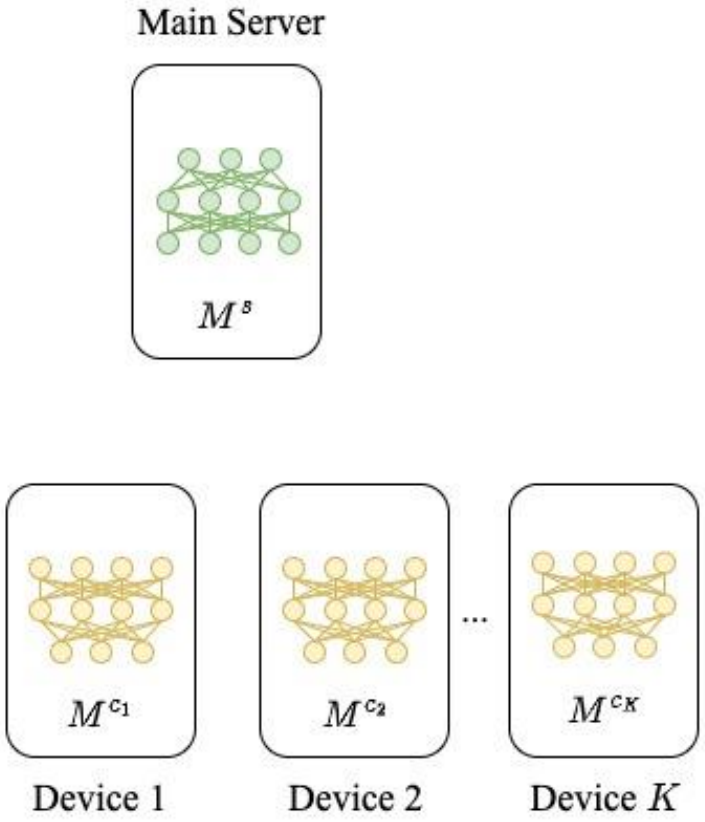
Split Learning (SL)



- Problem:

Only one device or the server will utilise its resources while the other devices or server are idle.

Split Federated Learning (SFL)

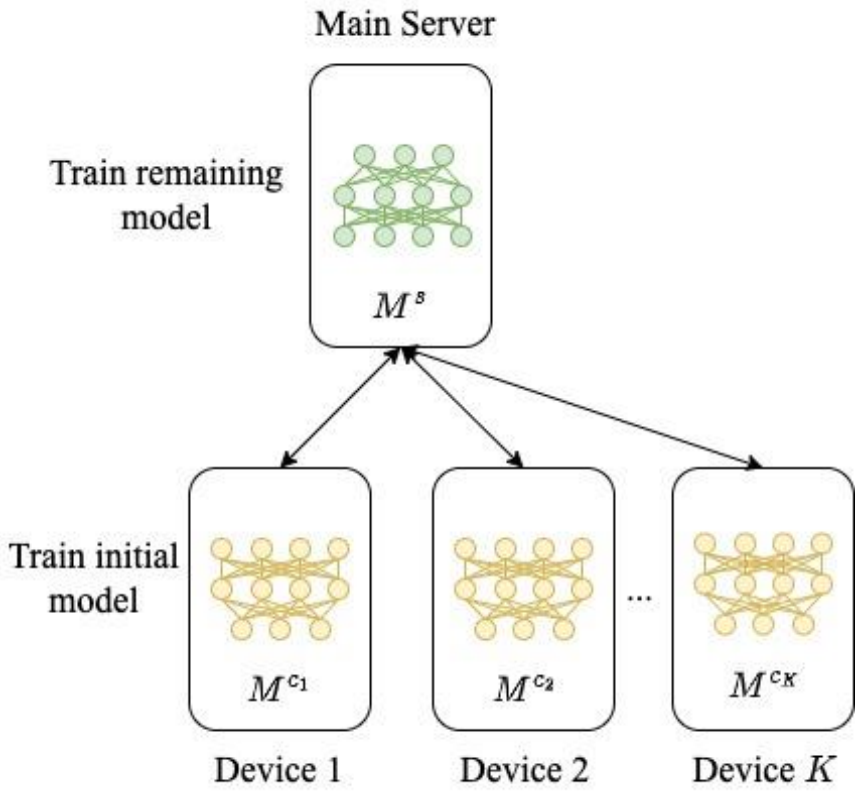


A hybrid of FL and SL.

- Problem:

The server is required to wait while the devices train the model and transfer data, and vice versa.

Split Federated Learning (SFL)

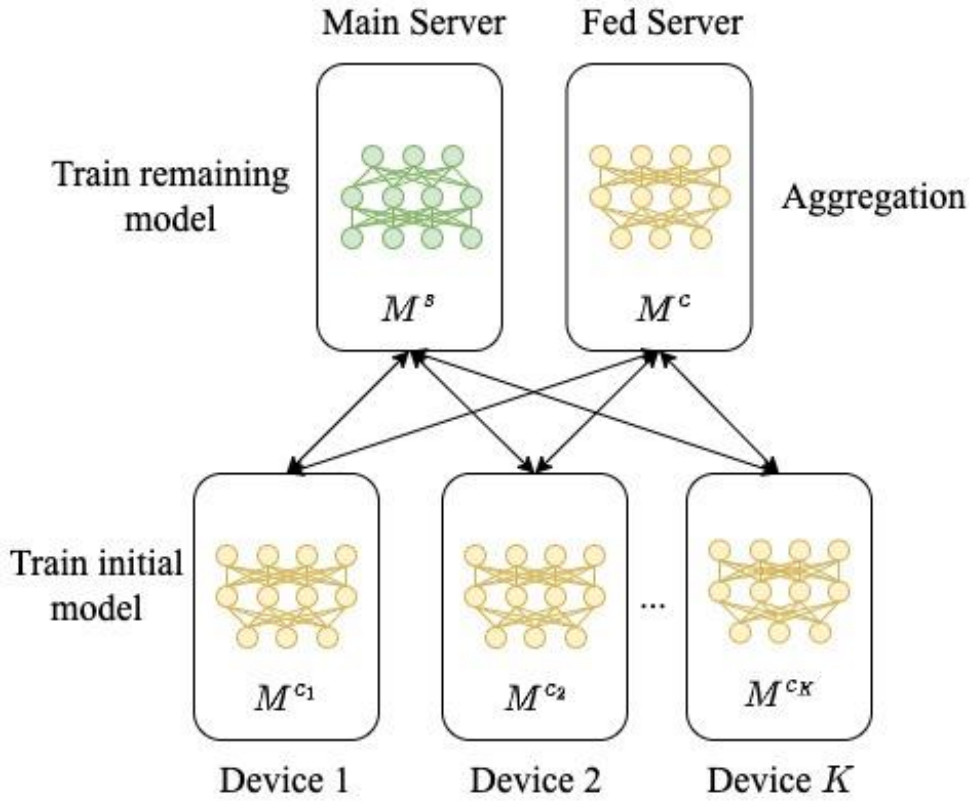


A hybrid of FL and SL.

- Problem:

The server is required to wait while the devices train the model and transfer data, and vice versa.

Split Federated Learning (SFL)



A hybrid of FL and SL.

- Problem:

The server is required to wait while the devices train the model and transfer data, and vice versa.

Resources Under-Utilisation Challenges

1. The server and devices cannot perform tasks simultaneously as they depend on each other.

2. There is a large communication overhead between the server and devices to enable collaboration.

3. The presence of stragglers when heterogeneous devices participate in training results in more wait times for faster devices.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.

(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.

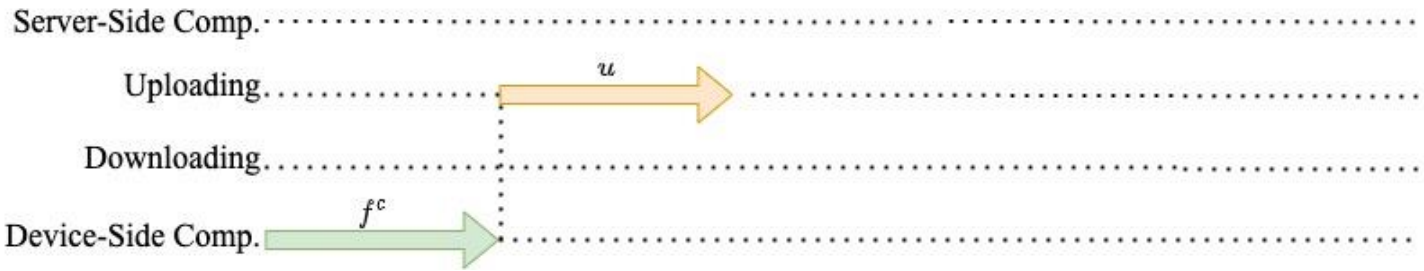
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



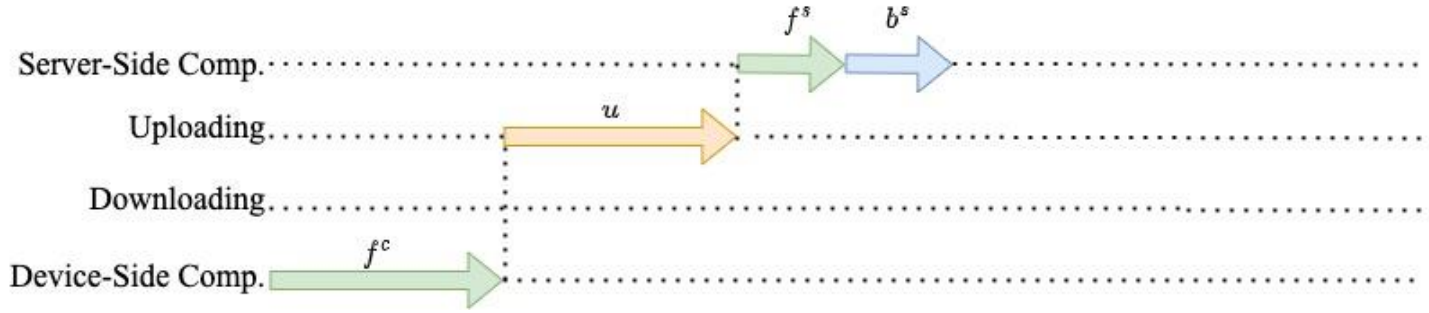
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



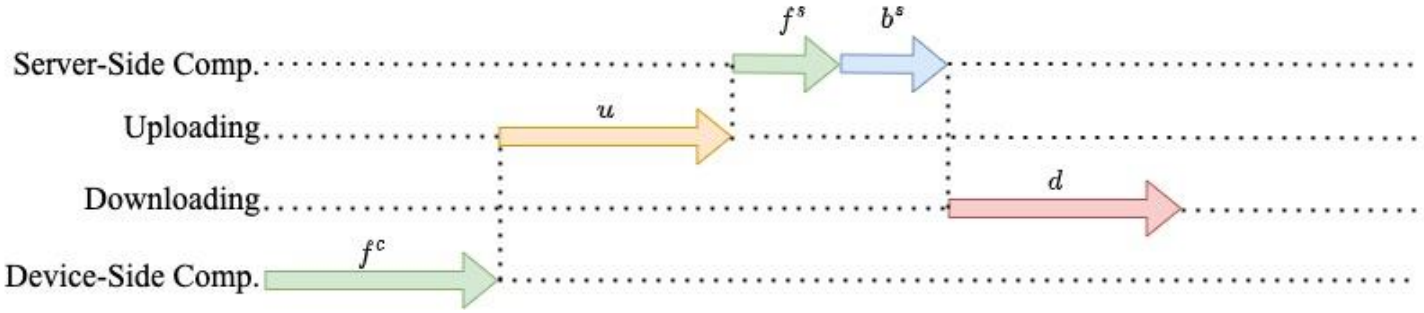
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



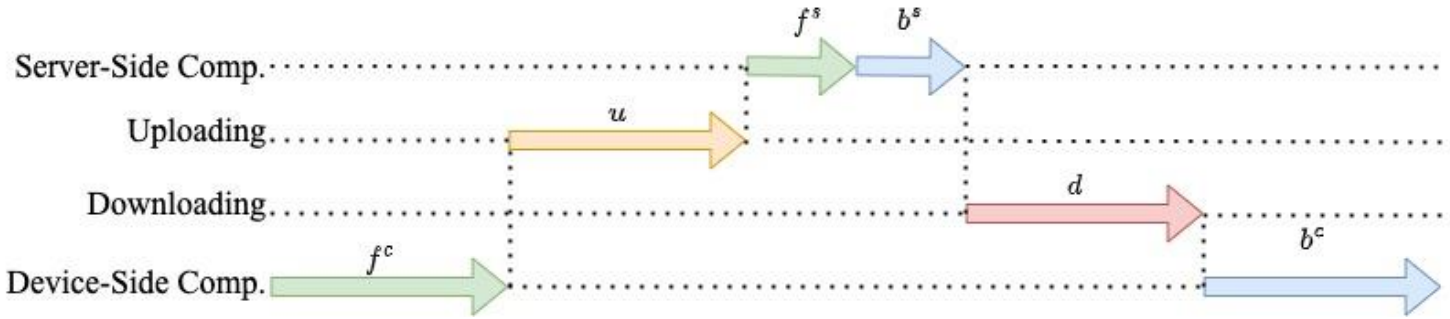
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



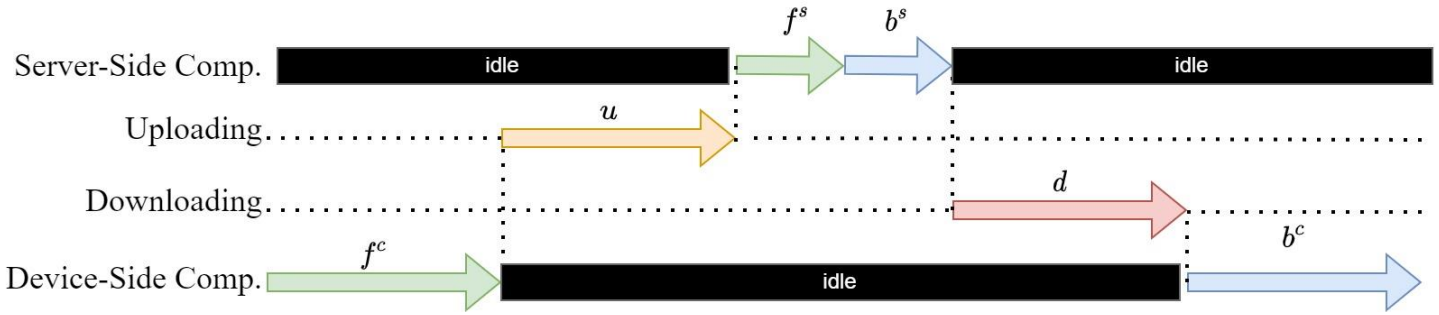
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



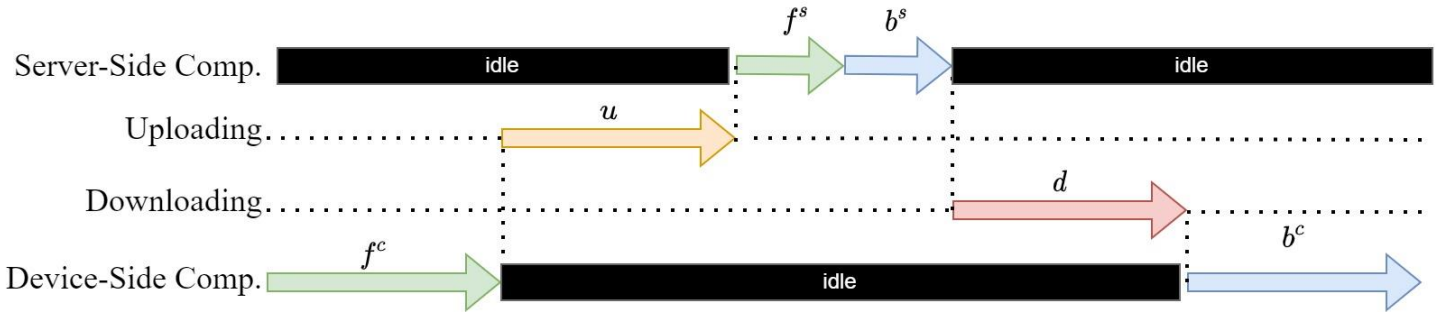
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



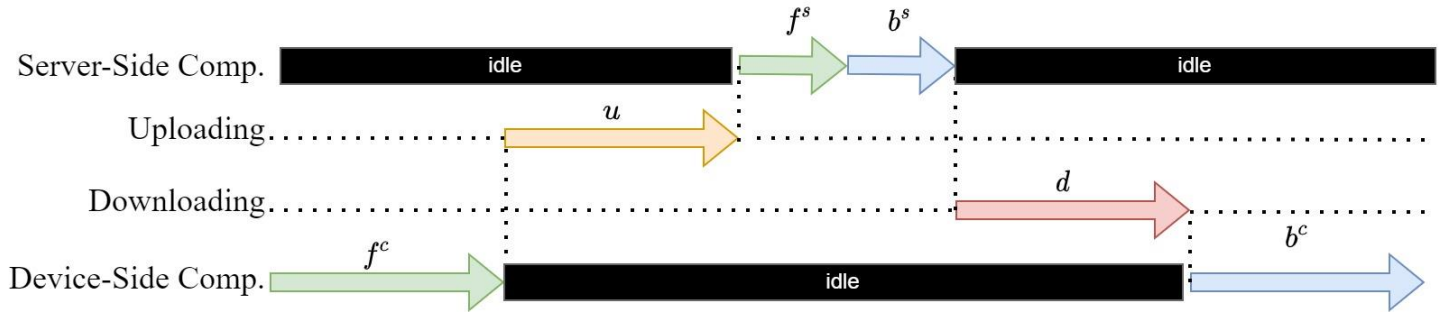
(a) Split Federated Learning

(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



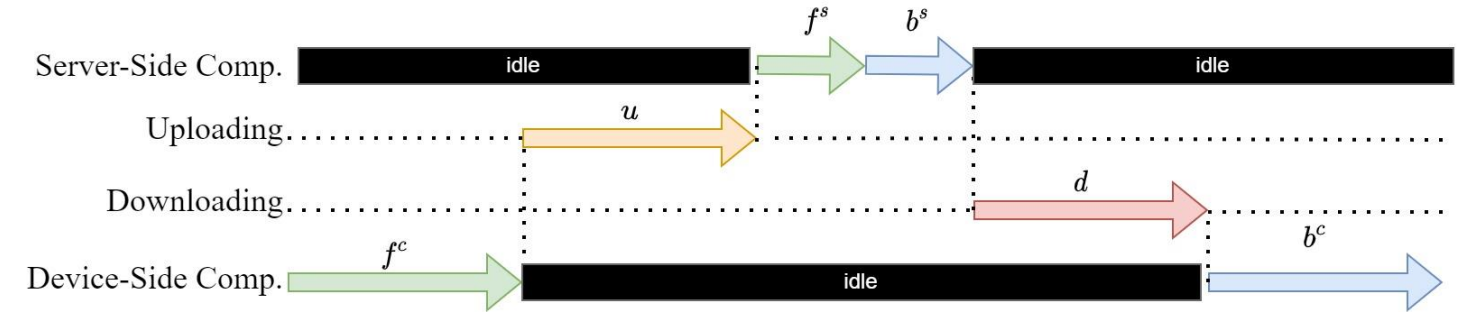
(a) Split Federated Learning

(b) PipeLearn

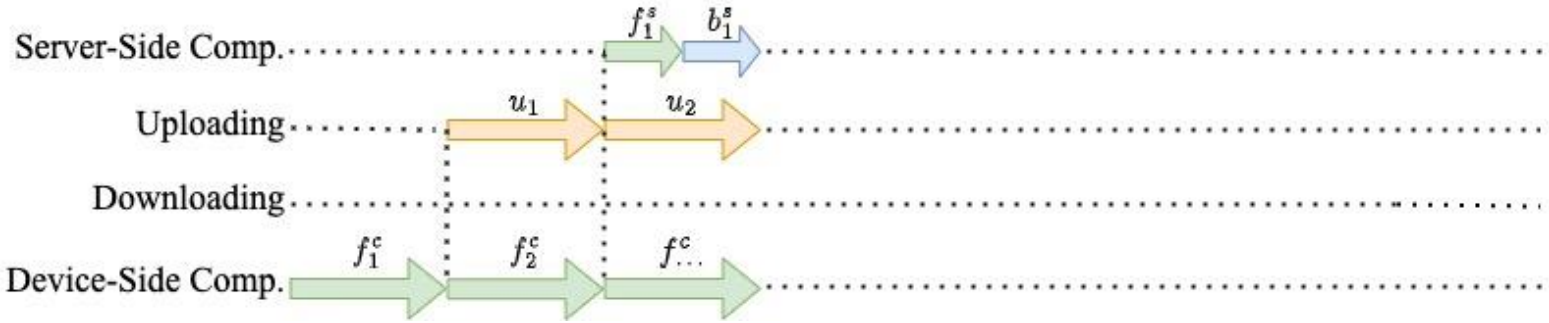
Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



(a) Split Federated Learning

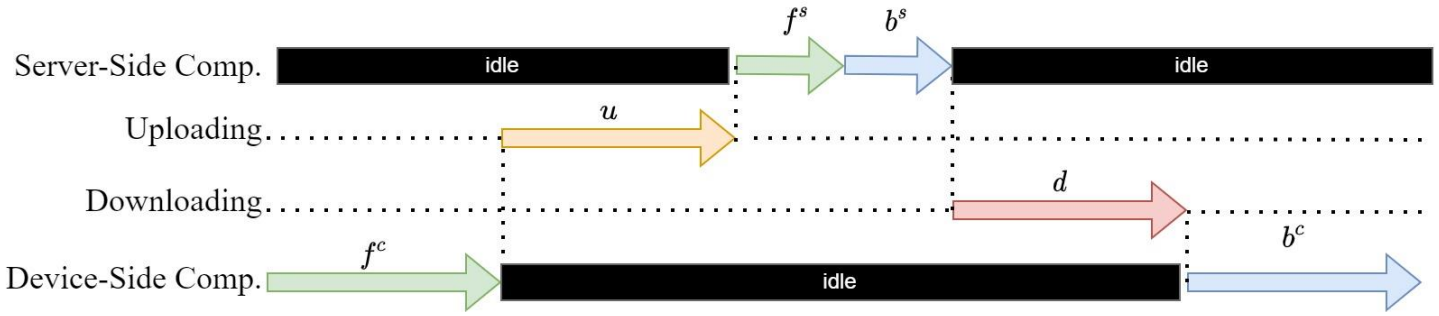


(b) PipeLearn

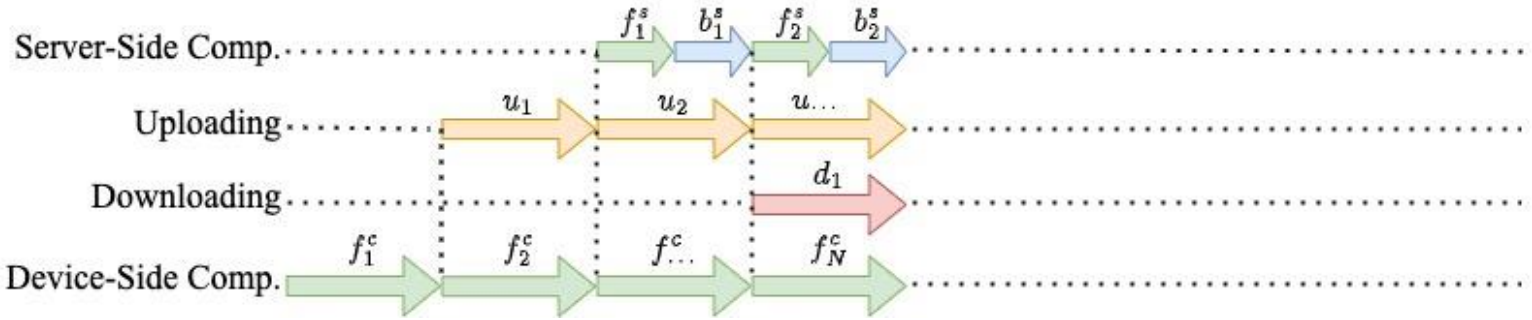
Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



(a) Split Federated Learning

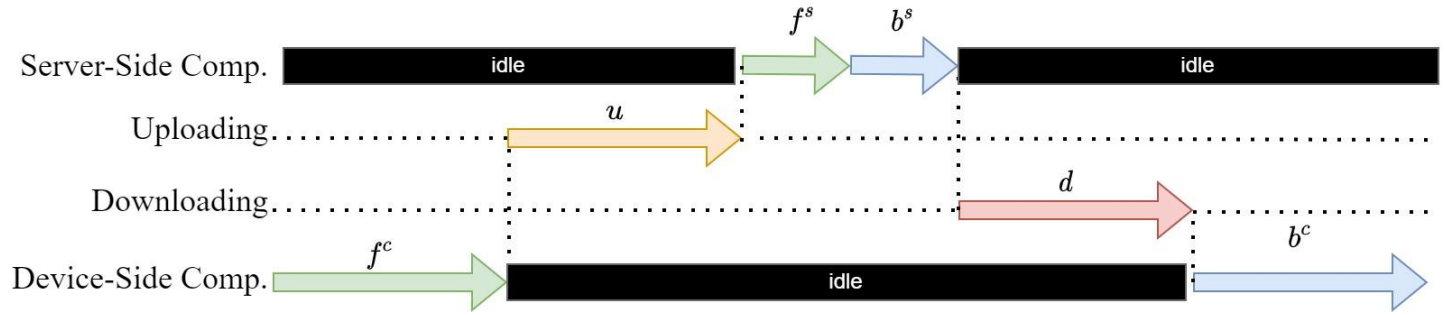


(b) PipeLearn

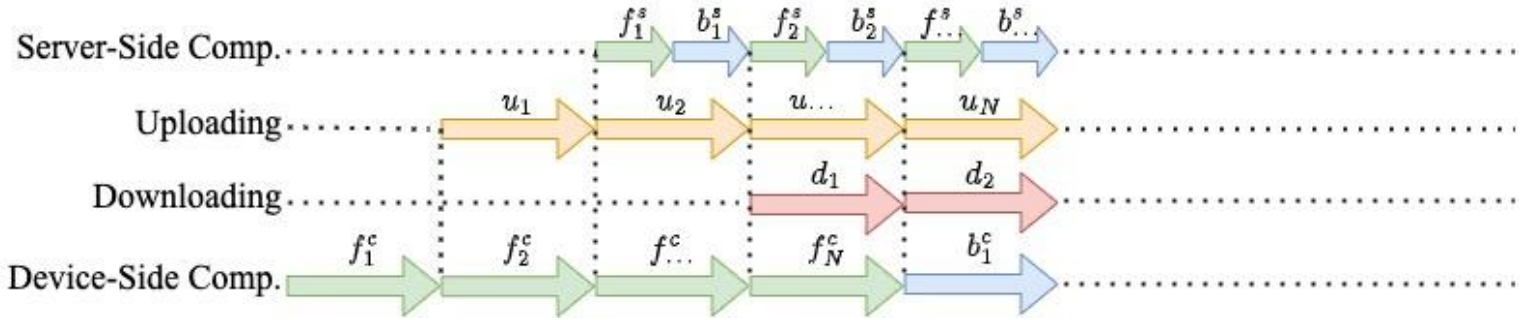
Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.



(a) Split Federated Learning



(b) PipeLearn

Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.

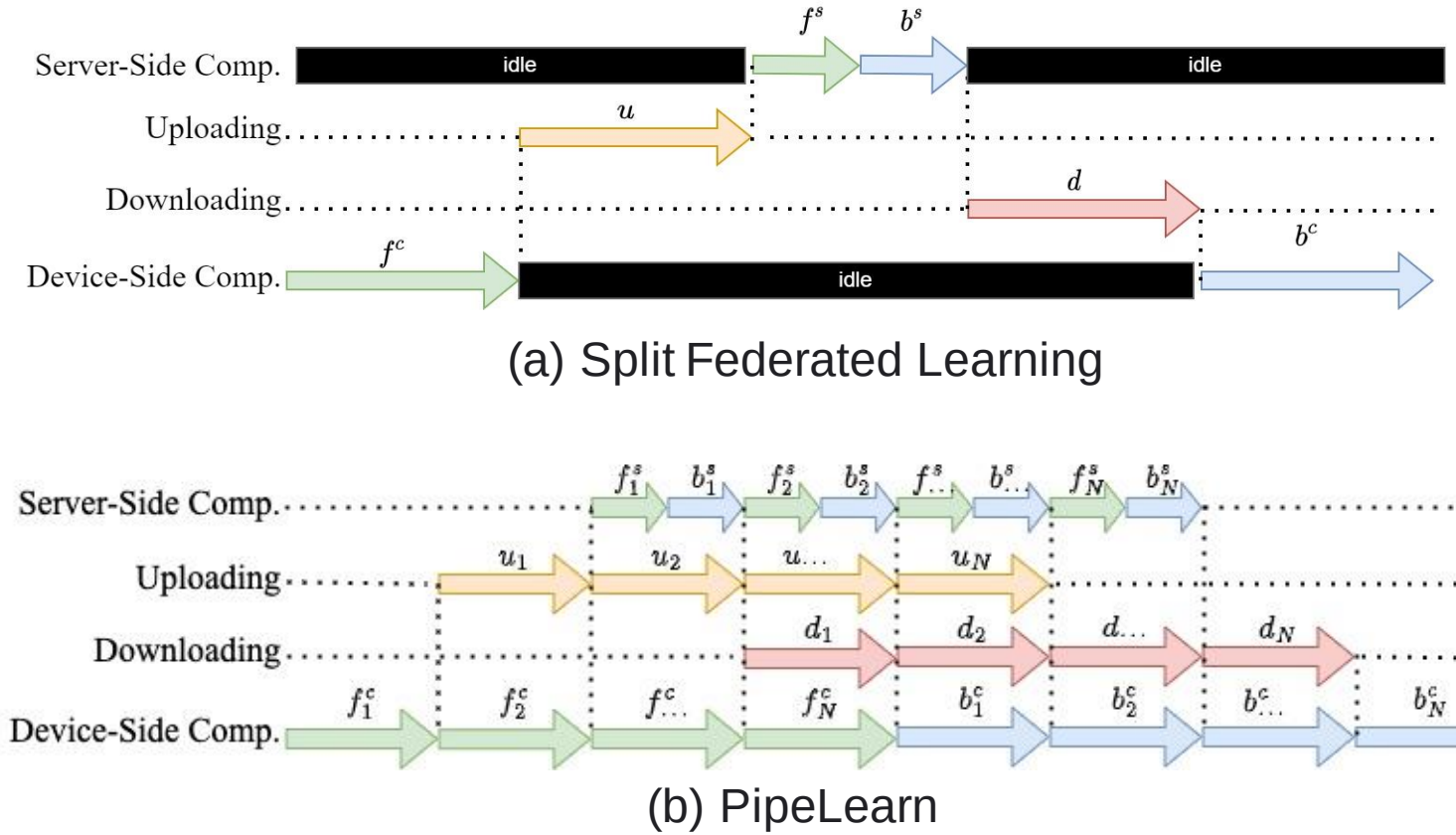


Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

PipeLearn

- Split the model across server and devices.
- Split each mini-batch of data to several micro-batches.
- Parallelise device-side computation, server-side computation and communication.
- All devices are training in parallel.

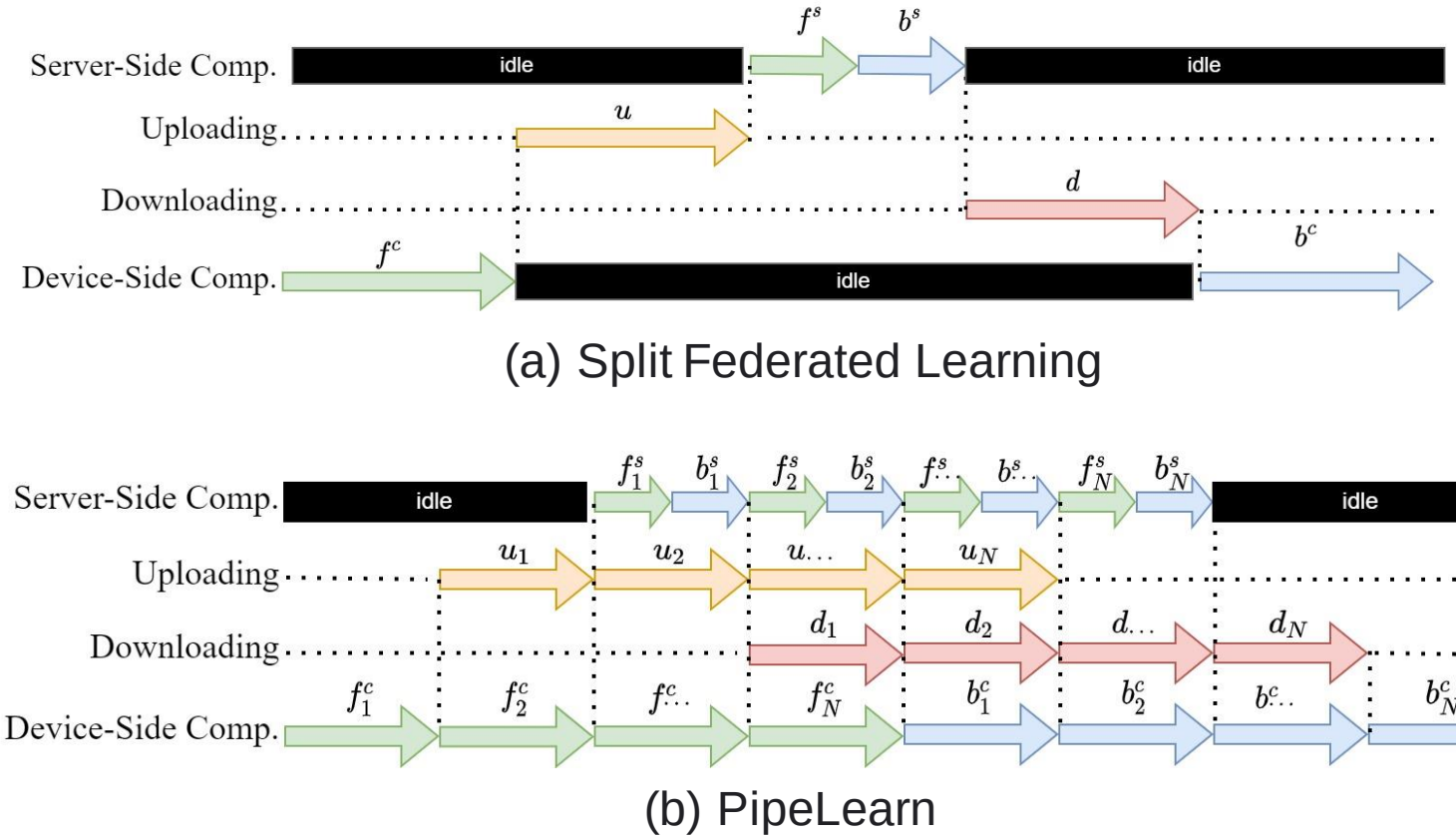
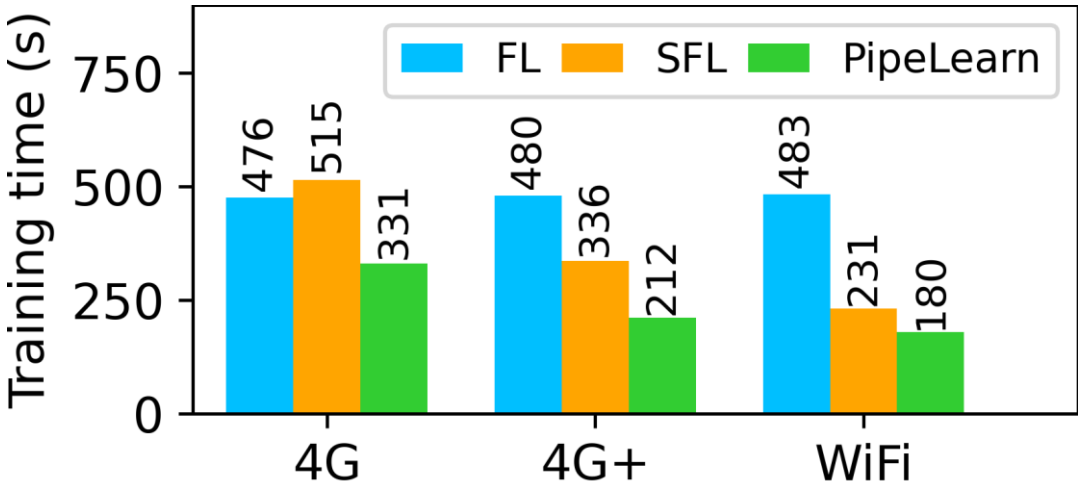
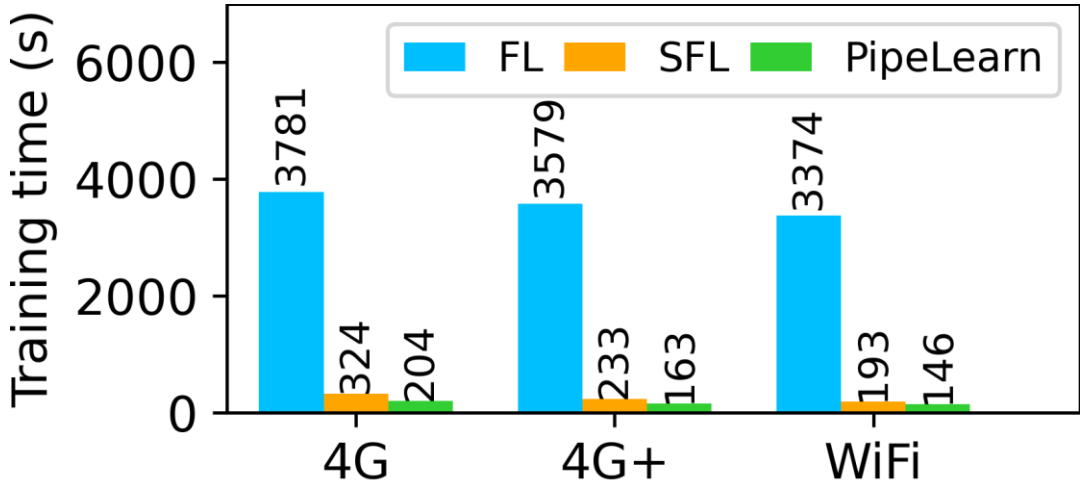


Figure 1. A training iteration for split federated learning and PipeLearn, where f, b, u and d represent forward pass, backward pass, upload and download, respectively.

Experiment: Training Efficiency



(a) VGG5

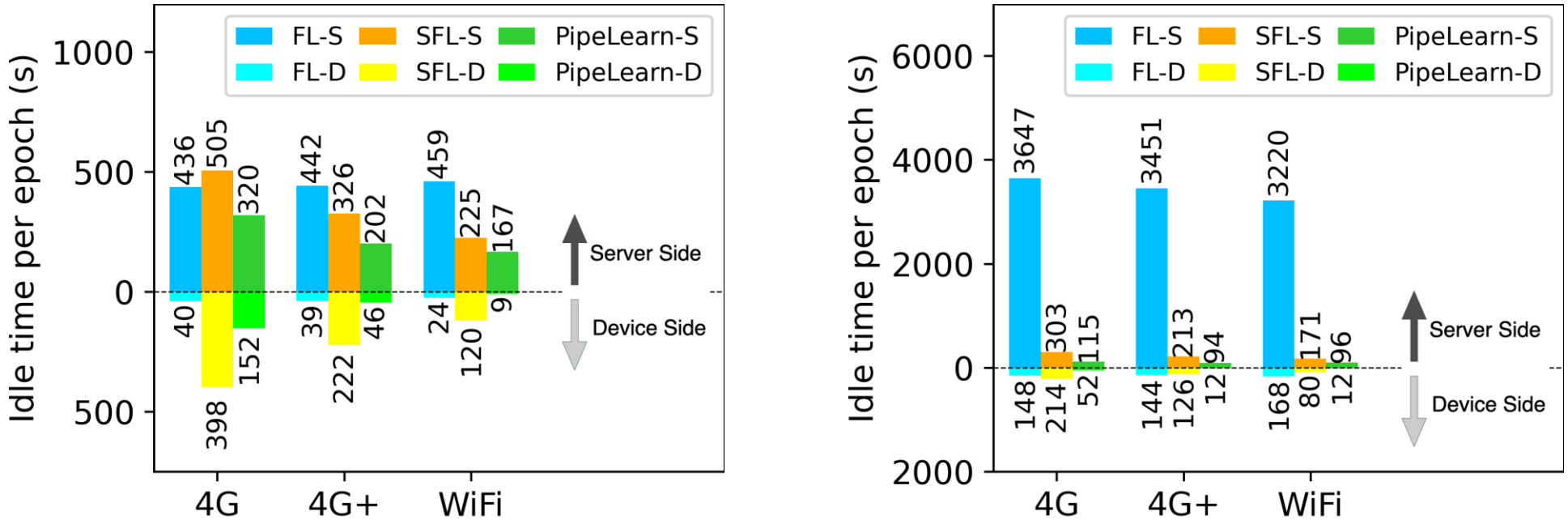


(b) ResNet18

Figure 2. Training time per epoch for FL, SFL and PipeLearn under different network conditions.

PipeLearn accelerates the training process by up to 27.6x.

Experiment: Idle Time



(a) VGG5

(b) ResNet18

Figure 3. Idle time per epoch on the server and devices in FL, SFL and PipeLearn under different network conditions.

PipeLearn reduces idle time by up to 58.1x.

Experiment: Model Accuracy

Technique	CIFAR-10			MNIST			SVHN		
	VGG-5	ResNet-18	MobileNetV3 -Small	VGG-5	ResNet-18	MobileNetV3 -Small	VGG-5	ResNet-18	MobileNetV3 -Small
FL	79.95	83.15	78.55	96.95	98.05	97	90.75	94.02	90.77
SFL	79.55	82.35	76.85	97.05	98	96.45	91	93.78	90.81
PipeLearn & 4G	79.15	83.15	77.85	97.45	98	97.2	90.83	93.62	90.6
PipeLearn & 4G+	78.4	82.85	77.7	97.6	97.95	97.45	91.25	93.67	90.63
PipeLearn & WiFi	78.65	83.3	77.5	97.35	97.85	96.75	90.9	93.77	90.52

Table 1. Model accuracy of VGG5 and ResNet18 on the test dataset using FL, SFL and PipeLearn, under different network conditions.

PipeLearn achieves (near) similar model accuracy.

Resources Under-Utilisation Challenges

1. The server and devices cannot perform tasks simultaneously as they depend on each other. *PipeLearn adjusts training pipeline to make them work simultaneously.*

2. There is a large communication overhead between the server and devices to enable collaboration. *PipeLearn overlaps computation with communication.*

3. The presence of stragglers when heterogeneous devices participate in training results in more wait times for faster devices. *Future Work.*

Concluding Remarks

- Z. Zhang, P. Rodgers, P. Kilpatrick, I. Spence and B. Varghese, "PipeLearn: Pipeline Parallelism for Collaborative Machine Learning," IEEE Transactions on Parallel and Distributed Systems, 2022 [Under Revision].

- One US patent filed with the sponsor of this research, Rakuten Mobile, Inc., Japan.


University of
St Andrews
www.st-andrews.ac.uk