

PockEngine: Sparse and Efficient Fine-tuning in a Pocket

Micro'23



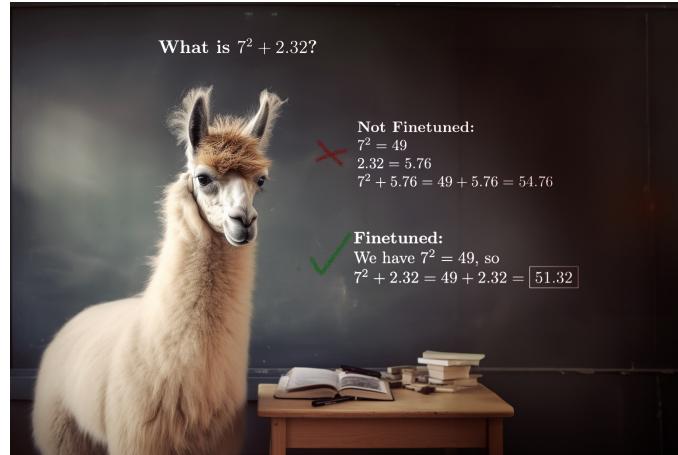
Ligeng Zhu, Lanxiang Hu, Ji Lin, Wei-Chen Wang, Wei-Ming Chen, Chuang Gan and Song Han

MIT, UCSD, MIT-IBM Watson AI Lab and NVIDIA

Background

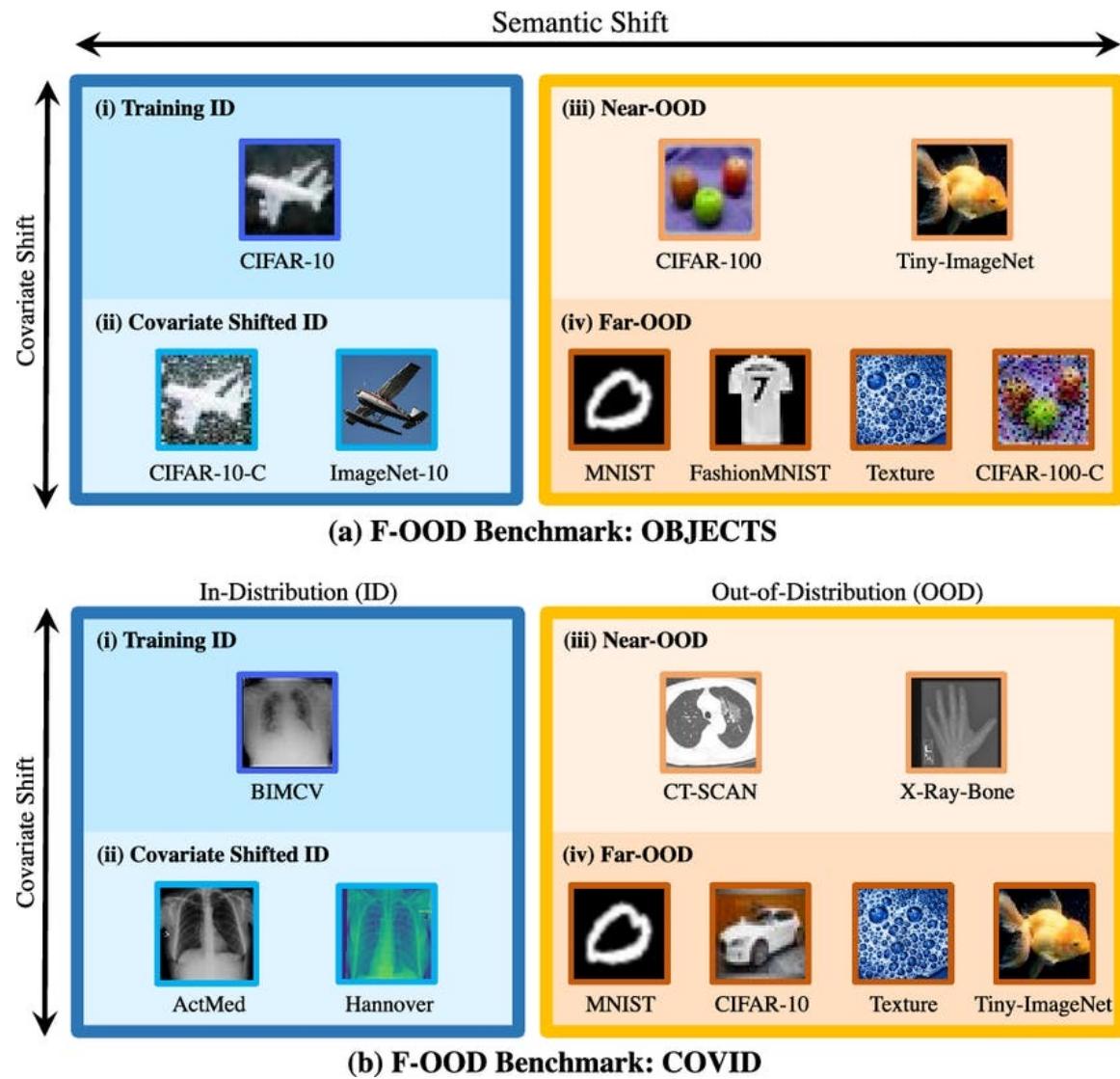
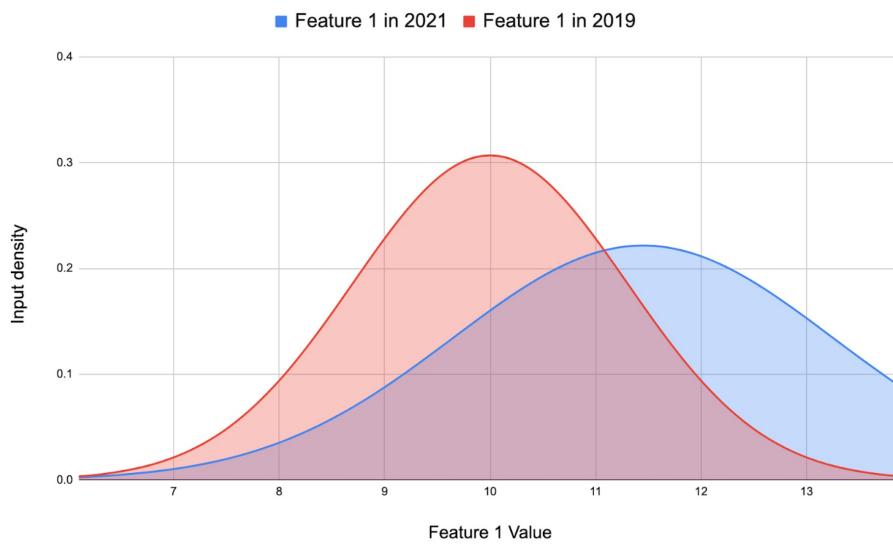
- Users are requiring specialized model

Domain specific llamas



Background

- Data domain shift needs continual learning



Background

- Cloud is powerful enough to meet these requirements

#	Title	Solution	Acceptance	Difficulty	Frequency
✓ 85	Maximal Rectangle	33.0%	Hard	<div style="width: 100%; height: 10px; background-color: green;"></div>	
✓ 621	Task Scheduler	45.2%	Medium	<div style="width: 100%; height: 10px; background-color: orange;"></div>	
✓ 412	Fizz Buzz	59.3%	Easy	<div style="width: 100%; height: 10px; background-color: green;"></div>	
✓ 41	First Missing Positive	28.6%	Hard	<div style="width: 100%; height: 10px; background-color: orange;"></div>	
✓ 54	Spiral Matrix	30.2%	Medium	<div style="width: 100%; height: 10px; background-color: green;"></div>	
642	Design Search Autocomplete System	37.1%	Hard	<div style="width: 100%; height: 10px; background-color: orange;"></div>	
✓ 344	Reverse String	63.1%	Easy	<div style="width: 100%; height: 10px; background-color: green;"></div>	
✓ 127	Word Ladder	23.7%	Medium	<div style="width: 100%; height: 10px; background-color: orange;"></div>	
124	Binary Tree Maximum Path Sum	29.8%	Hard	<div style="width: 100%; height: 10px; background-color: orange;"></div>	
✓ 336	Palindrome Pairs	30.7%	Hard	<div style="width: 100%; height: 10px; background-color: orange;"></div>	
✓ 141	Linked List Cycle	36.5%	Easy	<div style="width: 100%; height: 10px; background-color: green;"></div>	
193	Valid Phone Numbers	25.1%	Easy	<div style="width: 100%; height: 10px; background-color: green;"></div>	

Public data

```
1 import './App.css'
2
3 function App() {
4   return (
5     <div>
6       <a href="https://vitejs.dev" target="_blank">
7         <img src={ viteLogo } className="logo" alt="Vite logo" />
8       </a>
9       <a href="https://react.dev" target="_blank">
10         <img src={ reactLogo } className="logo react" alt="React logo" />
11       </a>
12     </div>
13     <h1>Vite + React</h1>
14     <div className="card">
15       <p>Hello wo... You, 11 seconds ago + Uncommitted changes</p>
16     </div>
17   )
18 }
19
20 <p>-<span>Read the docs</span></p>
21 <p>click on the Vite and React logos to learn more</p>
22 </div>
23
24 export default App
25
26
27
28
29
30
```

Personal data



Background

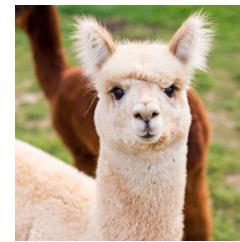
- Cloud is powerful enough to meet these requirements

#	Title	Solution	Acceptance	Difficulty	Frequency
✓ 85	Maximal Rectangle		33.0%		
✓ 621	Task Scheduler		45.2%		
✓ 412	Fizz Buzz		59.3%		
✓ 41	First Missing Positive		28.6%		
✓ 54	Spiral Matrix		30.2%		
642	Design Search Autocomplete System		37.1%		
✓ 344	Reverse String		63.1%		
✓ 127	Word Ladder		23.7%		
124	Binary Tree Maximum Path Sum		29.8%		
✓ 336	Palindrome Pairs		30.7%		
✓ 141	Linked List Cycle		36.5%		
193	Valid Phone Numbers		25.1%		

Public data

```
1 import './App.css'
2
3 function App() {
4   return (
5     <div>
6       <a href="https://vitejs.dev" target="_blank">
7         <img src={ viteLogo } className="logo" alt="Vite logo" />
8       </a>
9       <a href="https://react.dev" target="_blank">
10         <img src={ reactLogo } className="logo react" alt="React logo" />
11       </a>
12     </div>
13     <h1>Vite + React</h1>
14     <div className="card">
15       <p>Hello world! You, 11 seconds ago + Uncommitted changes</p>
16     </div>
17   )
18 }
19
20 <p>Click on the Vite and React logos to learn more</p>
21
22 <div>
23   <p><b>Read the Docs</b></p>
24   <p>click on the Vite and React logos to learn more</p>
25 </div>
26
27 </div>
28
29 export default App
30
```

Personal data



Users want to protect their privacy!

Background

- Cloud is powerful enough to meet these requirements

#	Title	Solution	Acceptance	Difficulty	Frequency
✓ 85	Maximal Rectangle	33.0%	Hard	<div style="width: 100px; height: 10px; background-color: green;"></div>	
✓ 621	Task Scheduler	45.2%	Medium	<div style="width: 100px; height: 10px; background-color: orange;"></div>	
✓ 412	Fizz Buzz	59.3%	Easy	<div style="width: 100px; height: 10px; background-color: green;"></div>	
✓ 41	First Missing Positive	28.6%	Hard	<div style="width: 100px; height: 10px; background-color: orange;"></div>	
✓ 54	Spiral Matrix	30.2%	Medium	<div style="width: 100px; height: 10px; background-color: green;"></div>	
642	Design Search Autocomplete System	37.1%	Hard	<div style="width: 100px; height: 10px; background-color: orange;"></div>	
✓ 344	Reverse String	63.1%	Easy	<div style="width: 100px; height: 10px; background-color: green;"></div>	
✓ 127	Word Ladder	23.7%	Medium	<div style="width: 100px; height: 10px; background-color: orange;"></div>	
124	Binary Tree Maximum Path Sum	29.8%	Hard	<div style="width: 100px; height: 10px; background-color: orange;"></div>	
✓ 336	Palindrome Pairs	30.7%	Hard	<div style="width: 100px; height: 10px; background-color: orange;"></div>	
✓ 141	Linked List Cycle	36.5%	Easy	<div style="width: 100px; height: 10px; background-color: green;"></div>	
193	Valid Phone Numbers	25.1%	Easy	<div style="width: 100px; height: 10px; background-color: green;"></div>	

Public data

```
1 import './App.css'
2
3 function App() {
4   return (
5     <div>
6       <a href="https://vitejs.dev" target="_blank">
7         <img src={ viteLogo } className="logo" alt="Vite logo" />
8       </a>
9       <a href="https://react.dev" target="_blank">
10         <img src={ reactLogo } className="logo react" alt="React logo" />
11       </a>
12     </div>
13     <h1>Vite + React</h1>
14     <div className="card">
15       <p>Hello world! You, 11 seconds ago + Uncommitted changes</p>
16     </div>
17   )
18 }
19
20 <p>Click on the Vite and React logos to learn more</p>
21
22 <div className="read-the-docs">
23   <a href="https://vitejs.dev/guide/introduction.html" target="_blank">
24     Click on me!
25   </a>
26 </div>
27
28 export default App
29
30
```

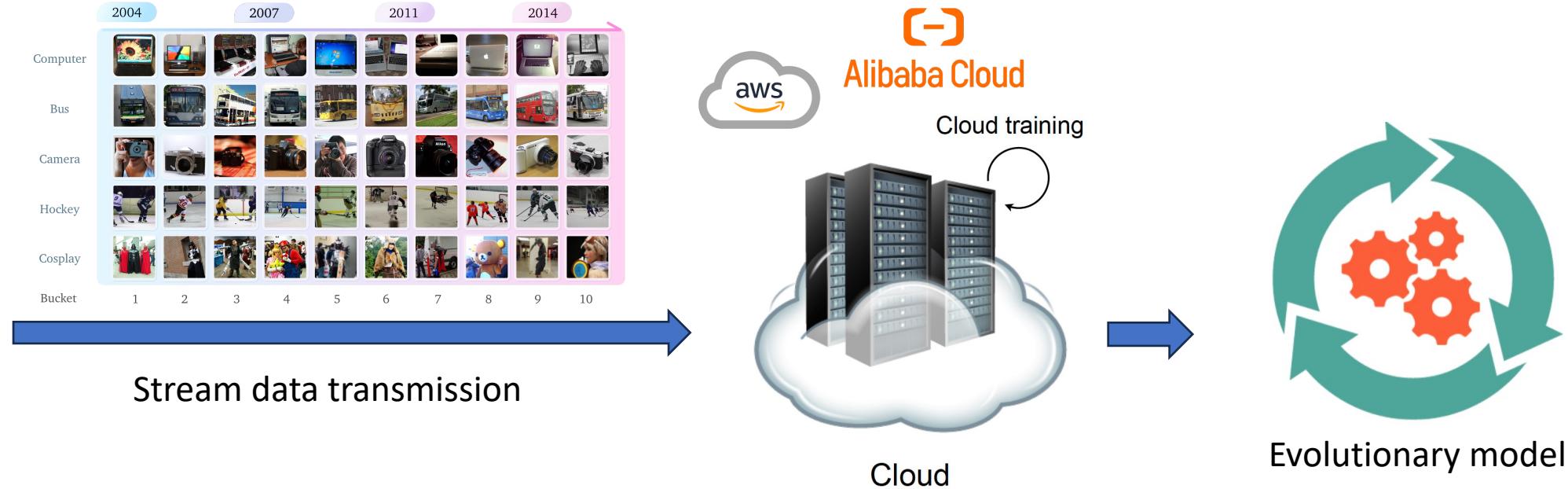
Personal data



It's expensive to fine-tune for each user.

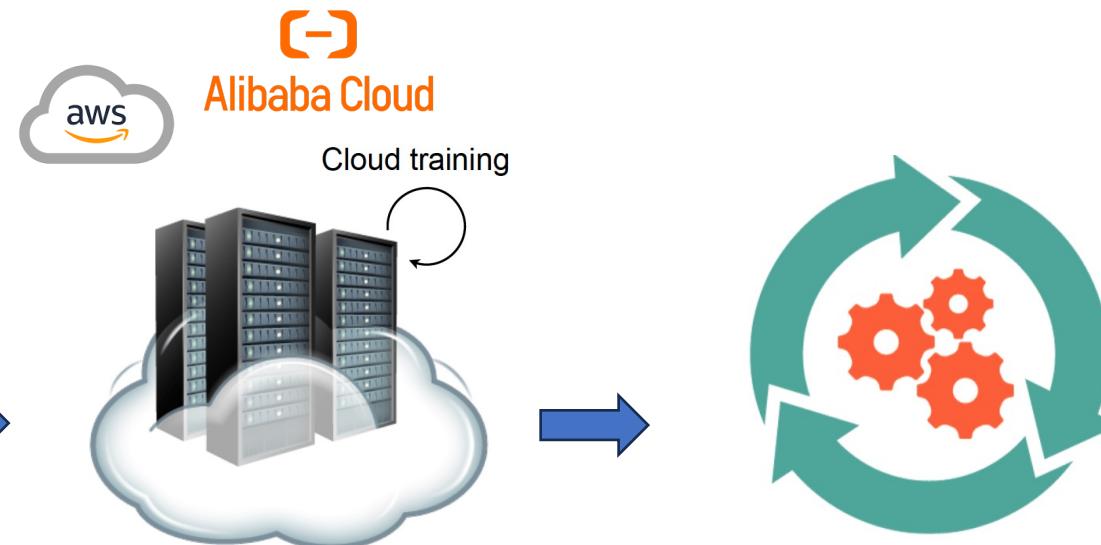
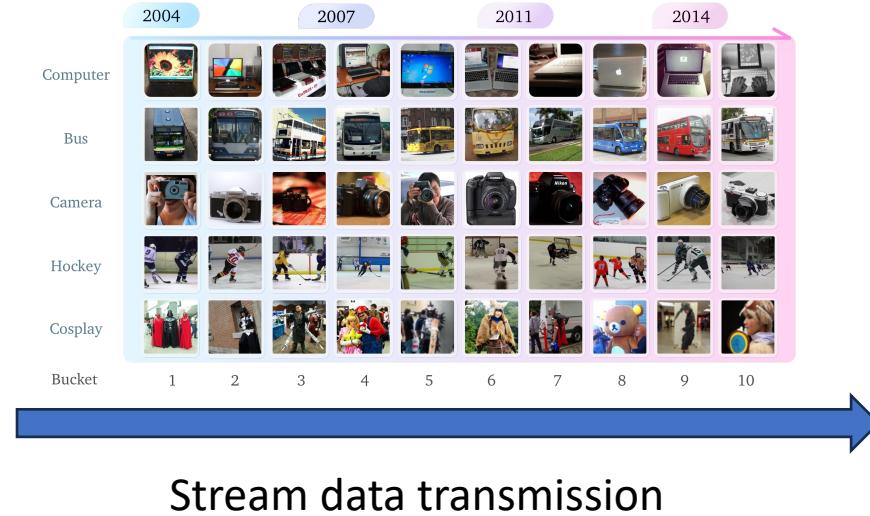
Background

- Cloud is powerful enough to meet these requirements

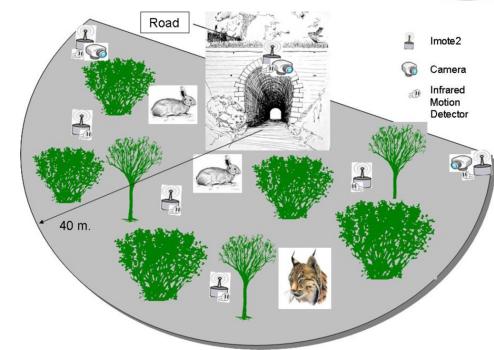


Background

- Cloud is powerful enough to meet these requirements



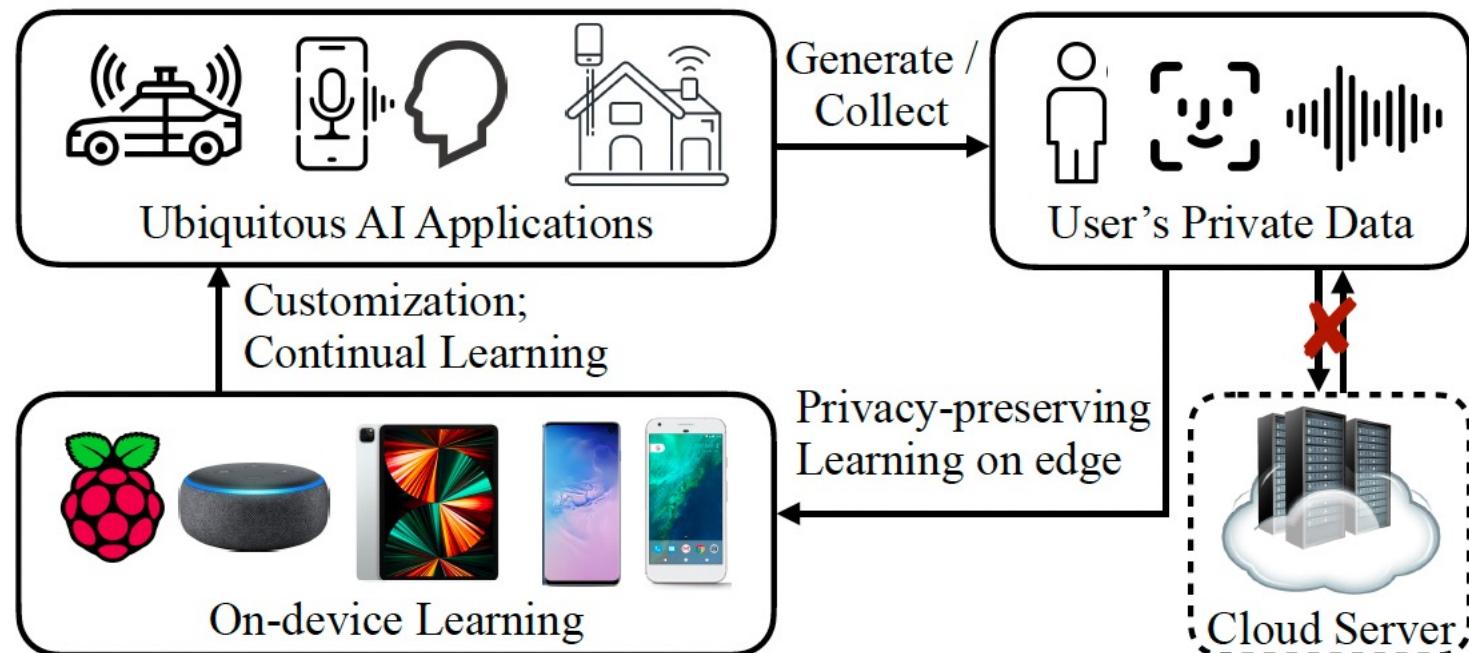
Privacy consideration



Multiple sensors incur expensive training cost.

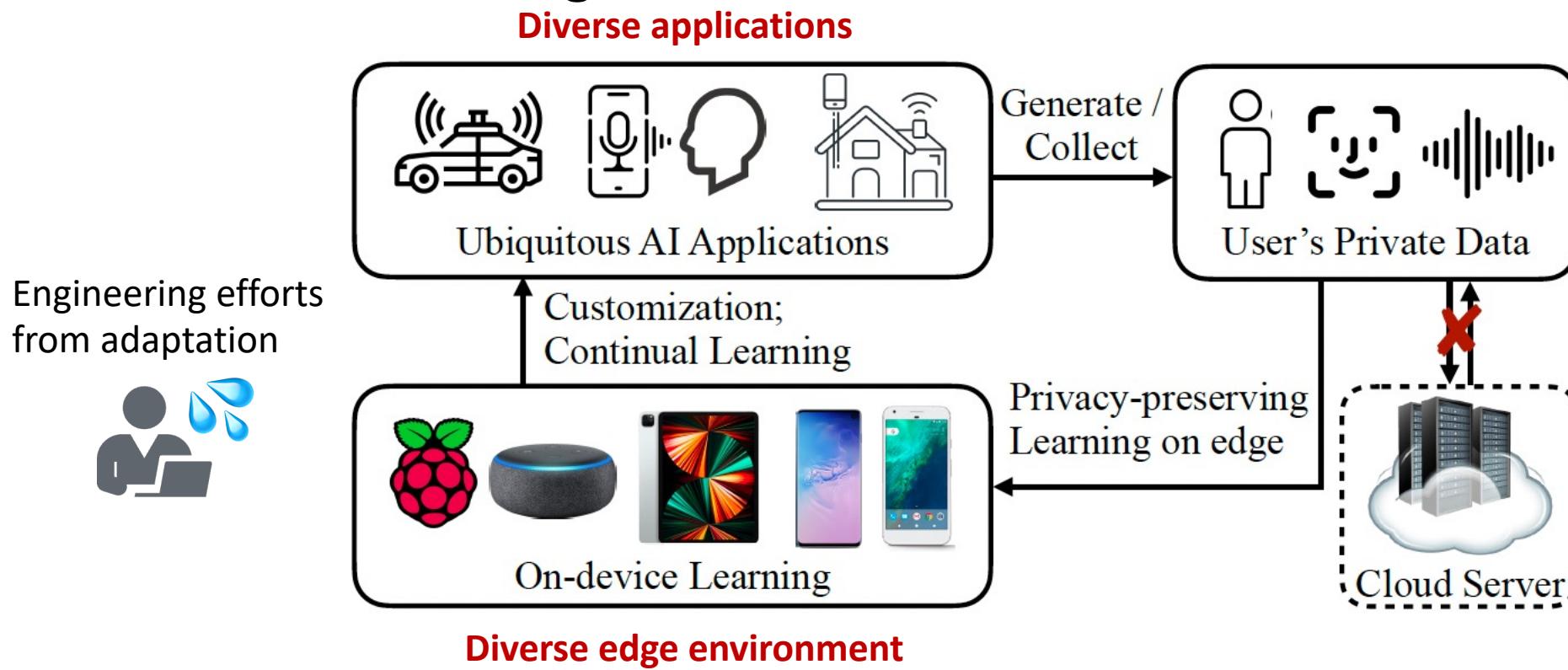
Background

- On-device learning



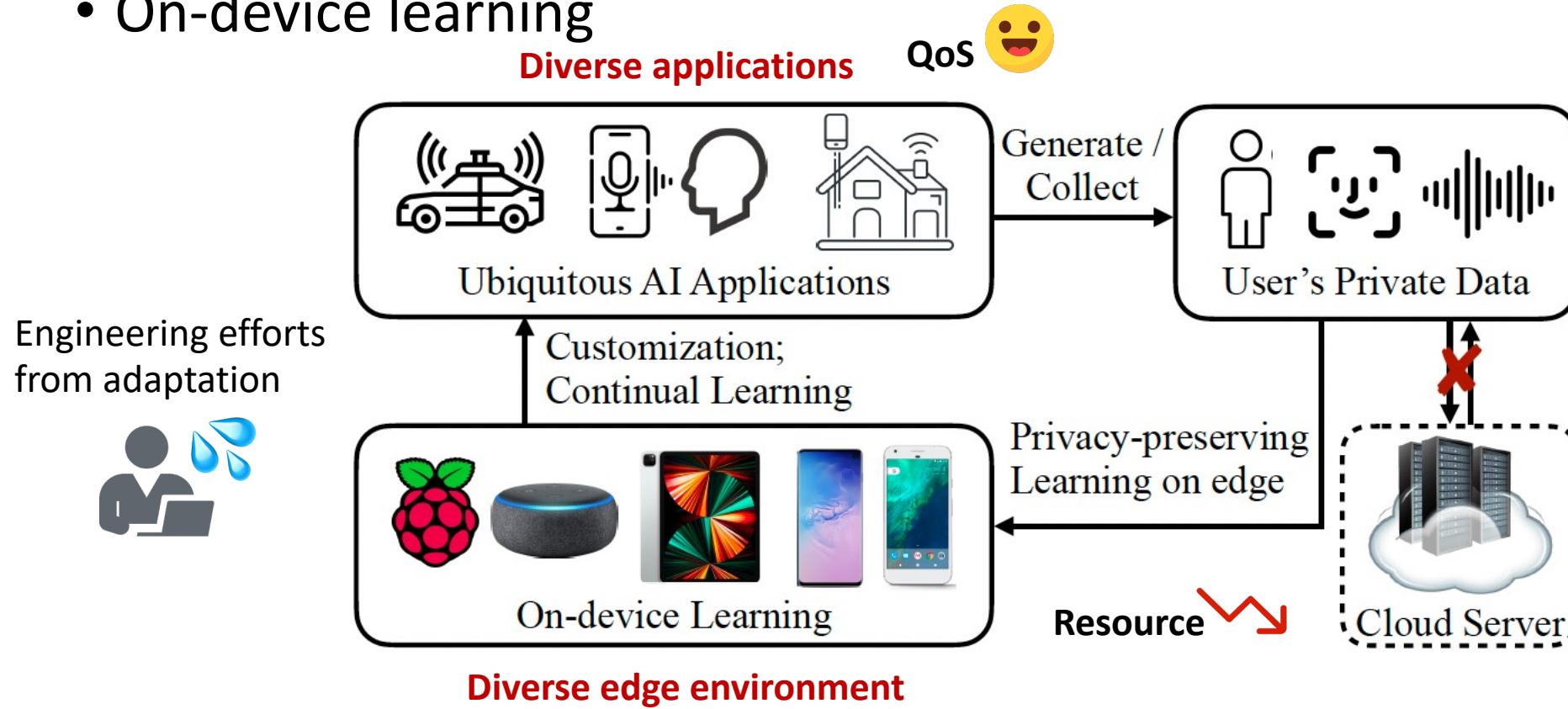
Background

- On-device learning



Background

- On-device learning



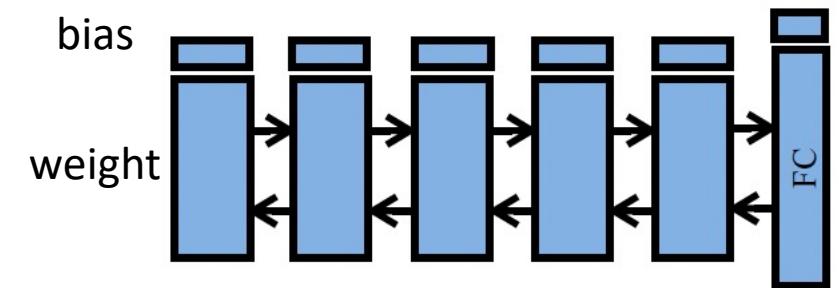
Background and Motivation

- Challenges in on-device training (learning)

- Resource limitations

- **Training is more expensive than inference especially on memory**

Forward: $a_{i+1} = a_i W_i + b_i \longrightarrow a = aw_0 + b_0, a = aw_1 + b_1, \dots$ **Local update to save memory**



Background and Motivation

- Challenges in on-device training (learning)

- Resource limitations

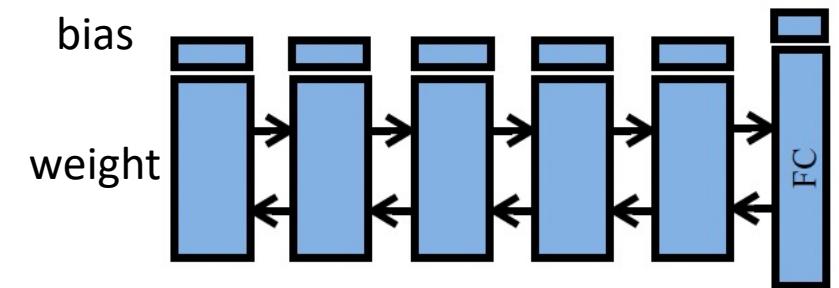
- **Training is more expensive than inference especially on memory**

Forward: $a_{i+1} = a_i W_i + b_i \longrightarrow a = aw_0 + b_0, a = aw_1 + b_1, \dots$ **Local update to save memory**

Backward: **1. compute gradient**

$$\frac{\partial L}{\partial W_i} = \mathbf{a}_i^T \frac{\partial L}{\partial a_{i+1}}, \frac{\partial L}{\partial a_i} = \frac{\partial L}{\partial a_{i+1}} W_i, \frac{\partial L}{\partial b_i} = \frac{\partial L}{\partial a_{i+1}}$$

Each activation need to be saved



Background and Motivation

- Challenges in on-device training (learning)

- Resource limitations

- **Training is more expensive than inference especially on memory**

Forward: $a_{i+1} = a_i W_i + b_i \longrightarrow a = aw_0 + b_0, a = aw_1 + b_1, \dots$ **Local update to save memory**

Backward: **1. compute gradient**

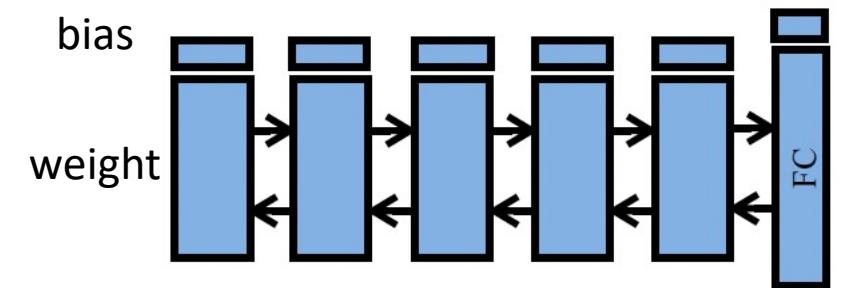
$$\frac{\partial L}{\partial W_i} = \mathbf{a}_i^T \frac{\partial L}{\partial a_{i+1}}, \frac{\partial L}{\partial a_i} = \frac{\partial L}{\partial a_{i+1}} W_i, \frac{\partial L}{\partial b_i} = \frac{\partial L}{\partial a_{i+1}}$$

Each activation need to be saved

2. update

$$u_{t+1} = \nabla_{\theta} f(\theta_t)$$

$$\theta_{t+1} = \theta_t - \alpha u_{t+1}$$



Background and Motivation

- Challenges in on-device training (learning)

- Resource limitations

- **Training is more expensive than inference especially on memory**

Forward: $a_{i+1} = a_i W_i + b_i \longrightarrow a = aw_0 + b_0, a = aw_1 + b_1, \dots$ **Local update to save memory**

Backward: **1. compute gradient**

$$\frac{\partial L}{\partial W_i} = \mathbf{a}_i^T \frac{\partial L}{\partial a_{i+1}}, \frac{\partial L}{\partial a_i} = \frac{\partial L}{\partial a_{i+1}} W_i, \frac{\partial L}{\partial b_i} = \frac{\partial L}{\partial a_{i+1}}$$

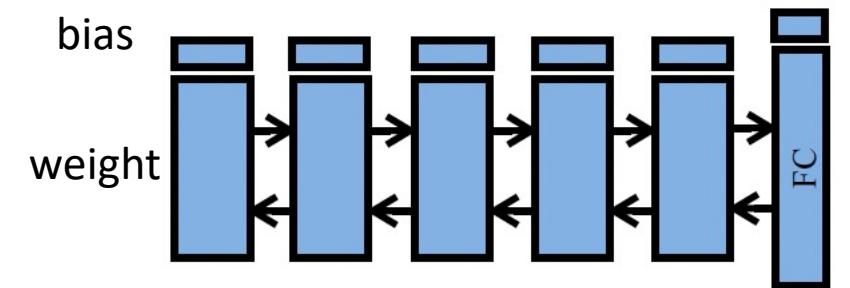
Each activation need to be saved

2. update

$$u_{t+1} = \nabla_{\theta} f(\theta_t)$$

$$\theta_{t+1} = \theta_t - \alpha u_{t+1}$$

Momentum: $u_{t+1} = \beta u_t + (1 - \beta) \nabla_{\theta} f(\theta_t)$ **2x cost**



Background and Motivation

- Challenges in on-device training (learning)

- Resource limitations

- Training is more expensive than inference especially on memory**

Forward: $a_{i+1} = a_i W_i + b_i \longrightarrow a = aw_0 + b_0, a = aw_1 + b_1, \dots$ **Local update to save memory**

Backward: **1. compute gradient**

$$\frac{\partial L}{\partial W_i} = \mathbf{a}_i^T \frac{\partial L}{\partial a_{i+1}}, \frac{\partial L}{\partial a_i} = \frac{\partial L}{\partial a_{i+1}} W_i, \frac{\partial L}{\partial b_i} = \frac{\partial L}{\partial a_{i+1}}$$

Each activation need to be saved

2. update

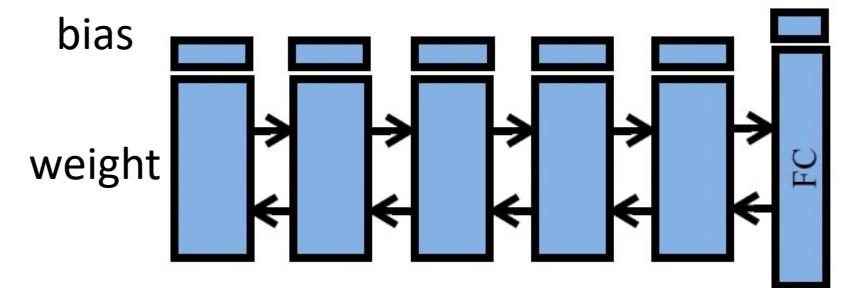
$$u_{t+1} = \nabla_{\theta} f(\theta_t)$$

$$\theta_{t+1} = \theta_t - \alpha u_{t+1}$$

Momentum: $u_{t+1} = \beta u_t + (1 - \beta) \nabla_{\theta} f(\theta_t)$ **2x cost**

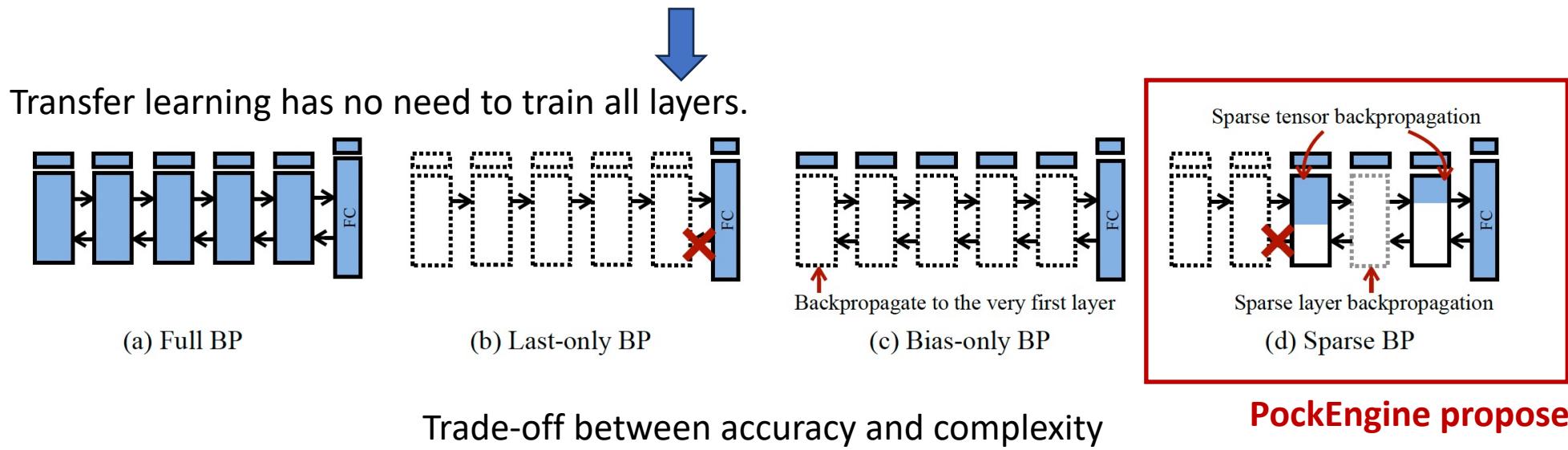
Adam: $v_{t+1} = \beta' v_t + (1 - \beta') (\nabla_{\theta} f(\theta_t))^2$

$\theta_{t+1} = \theta_t - \frac{\alpha u_{t+1}}{\sqrt{v_{t+1}} + \epsilon}$ **3x cost**



Background and Motivation

- Challenges in on-device training (learning)
 - Resource limitations
 - **Training is more expensive than inference especially on memory**
 - Edge device has no enough memory to use full back propagation



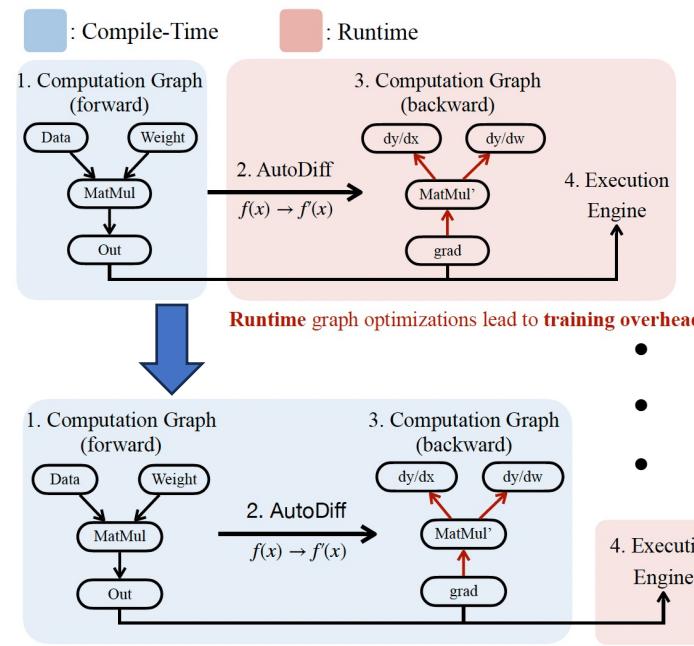
Background and Motivation

- Challenges in on-device training (learning)
 - Resource limitations
 - Training is more expensive than inference especially on memory
 - Edge device has no enough memory to use full back propagation
Propose sparse BP to solve.
 - Hardware diversity
 - Various frontends: PyTorch, TensorFlow, Jax ...
 - Various backends: NVIDIA GPUs, Qualcomm Adreno GPUs, Hexagon DSPs, ARM microcontrollers, Apple M1, edge TPUs ...

Background and Motivation

- Challenges in on-device training (learning)
 - Hardware diversity
 - Various frontends: PyTorch, TensorFlow, Jax ...
 - Various backends: NVIDIA GPUs, Qualcomm Adreno GPUs, Hexagon DSPs, ARM microcontrollers, Apple M1, edge TPUs ...

	Support Training	Support Sparse-BP	Run without Host Language	Kernel Optimized for Edge	Compile-Time AutoDiff	Graph Optimizations
PyTorch [46]	✓	✗	✗	✗	✗	✗
TensorFlow [4]	✓	✗	✗	✗	✗	-
Jax [9]	✓	✗	✗	✗	✗	✗
TVM [13]	✗	✗	✓	✓	-	✓
MNN [29]	✓	✗	✓	✓	✗	✗
PockEngine (ours)	✓	✓	✓	✓	✓	✓

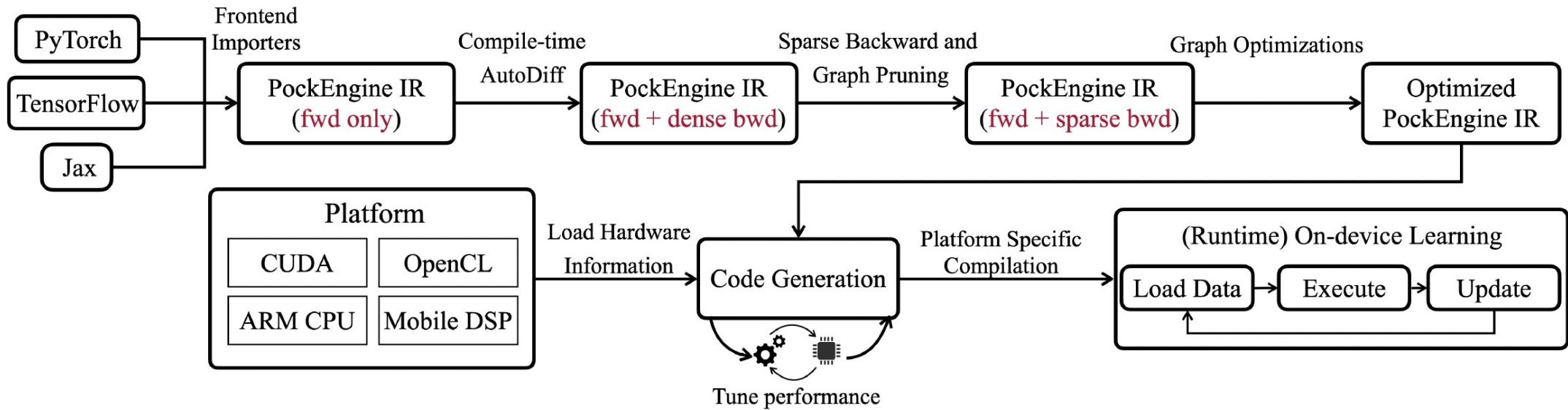


- Hide runtime AD cost
- Codegen to optimize kernel
- Unified IR to represent training DAG
- More aggressive optimizations

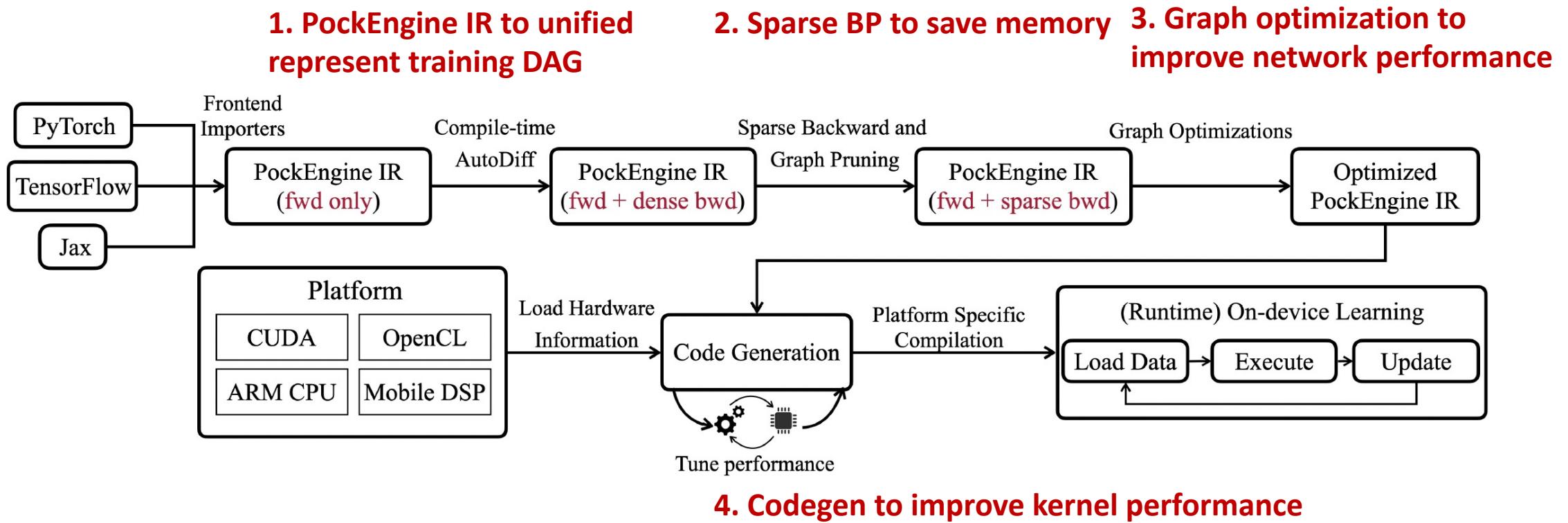
Background and Motivation

- Challenges in on-device training (learning)
 - Resource limitations
 - **Training is more expensive than inference especially on memory**
 - Edge device has no enough memory to use full back propagation
Propose **sparse BP** to solve.
 - Hardware diversity
 - Various frontends: PyTorch, TensorFlow, Jax ...
 - Various backends: NVIDIA GPUs, Qualcomm Adreno GPUs, Hexagon DSPs, ARM microcontrollers, Apple M1, edge TPUs ...
 - No framework supports on-device training
A **compiler** for covering the diversity.

PockEngine Workflow



PockEngine Workflow



PockEngine IR

- Representation and AutoDiff

```
net = nn.Sequential(  
    nn.Linear(10, 10),  
    nn.Linear(10, 10),  
)  
x = torch.randn(1, 3, 28, 28)  
out = net(x).sum().backward()
```

(b) PyTorch Model

```
# forward layer #1  
%0 = multiply(%x, %w1);  
%1 = add(%0, %b1);  
# forward layer #2  
%2 = multiply(%1, %w2);  
%3 = add(%2, %b2);
```

(c) Forward IR

```
# forward layer #1  
%0 = multiply(%x, %w1);  
%1 = add(%0, %b1);  
# forward layer #2  
%2 = multiply(%1, %w2);  
%3 = add(%2, %b2);  
# backward layer #2  
%4 = multiply(%grad, %w2);  
%5 = transpose(%grad);  
%6 = multiply(%5, %1);  
%7 = sum(%grad, axis=-1);  
# backward layer #1  
%8 = multiply(%6, %w1);  
%9 = transpose(%6);  
%10 = multiply(%9, %x);  
%11 = sum(%6, axis=-1);  
return (%6, %7, %10, %11)
```

(d) Forward + Backward IR

Abstract op computation to re-use implemented kernel from inference backends.

Sparse Backpropagation

- Cost model and search criterion

$$\begin{aligned} k^*, i^*, r^* = \max_{k,i,r} & \left(\sum_{k \in i} \Delta \text{acc}_{\mathbf{b}_k} + \sum_{i \in i, r \in r} \Delta \text{acc}_{\mathbf{W}_{i,r}} \right) \\ \text{s.t. } & \text{Memory}(k, i, r) \leq \text{constraint}, \end{aligned}$$

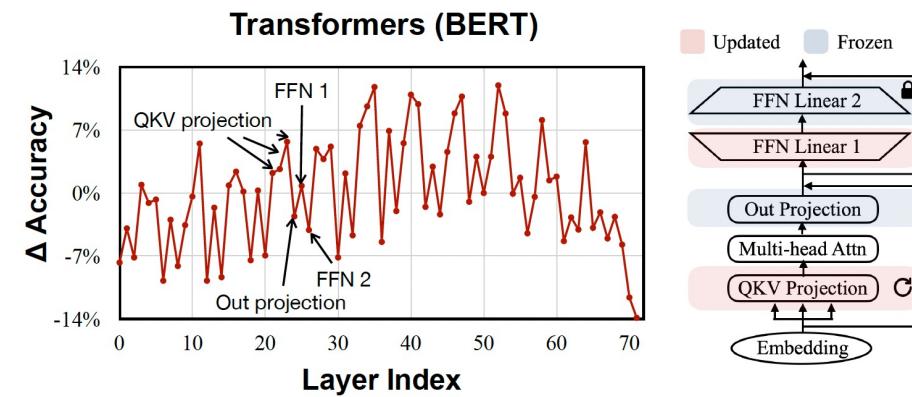
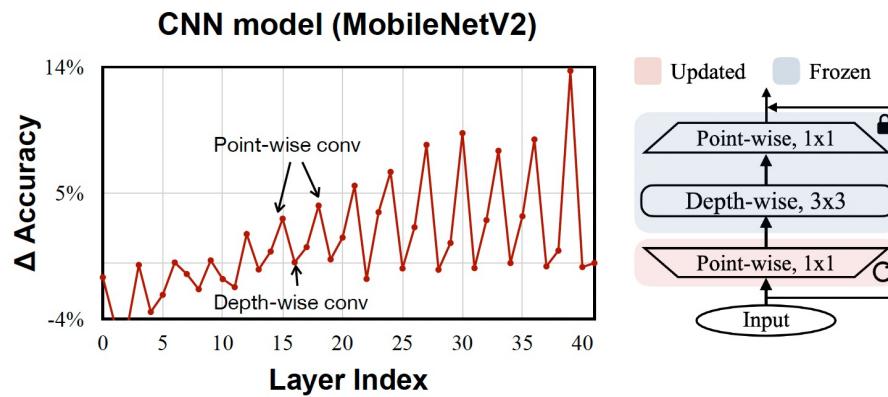
Sparse Backpropagation

- Cost model and search criterion

$$k^*, i^*, r^* = \max_{k,i,r} \left(\sum_{k \in i} \Delta acc_{b_k} + \sum_{i \in i, r \in r} \Delta acc_{w_{i,r}} \right)$$

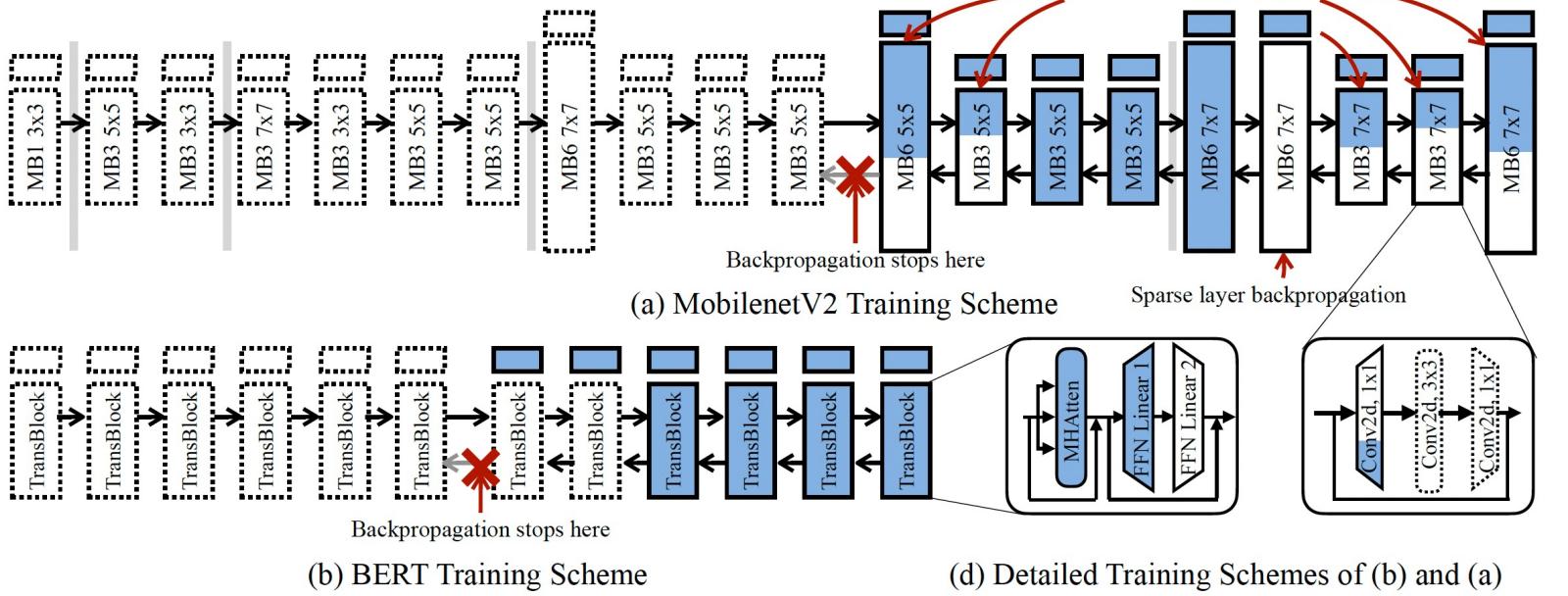
s.t. $\text{Memory}(k, i, r) \leq \text{constraint}$,

Δacc achieved by one layer contribution analysis; Different model has different schemes.



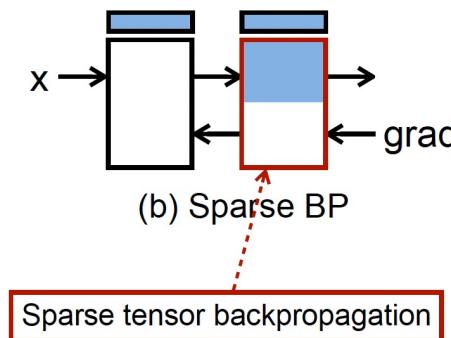
Sparse Backpropagation

- Sparse BP



PockEngine Compiler

- PockEngine IR based Sparse BP implementation



```
# forward layer #1
%0 = multiply(%x, %w1);
%1 = add(%0, %b1);

# forward layer #2
%2 = multiply(%1, %w2);
%3 = add(%2, %b2);

# backward layer #2
%4 = multiply(%grad, %w2);
%5 = transpose(%grad);
%6 = multiply(%5, %1);
%7 = sum(%grad, axis=-1);

# backward layer #1
%8 = multiply(%6, %w1);
%9 = transpose(%6);
%10 = multiply(%9, %x);
%11 = sum(%6, axis=-1);

return (%6, %7, %10, %11)
```

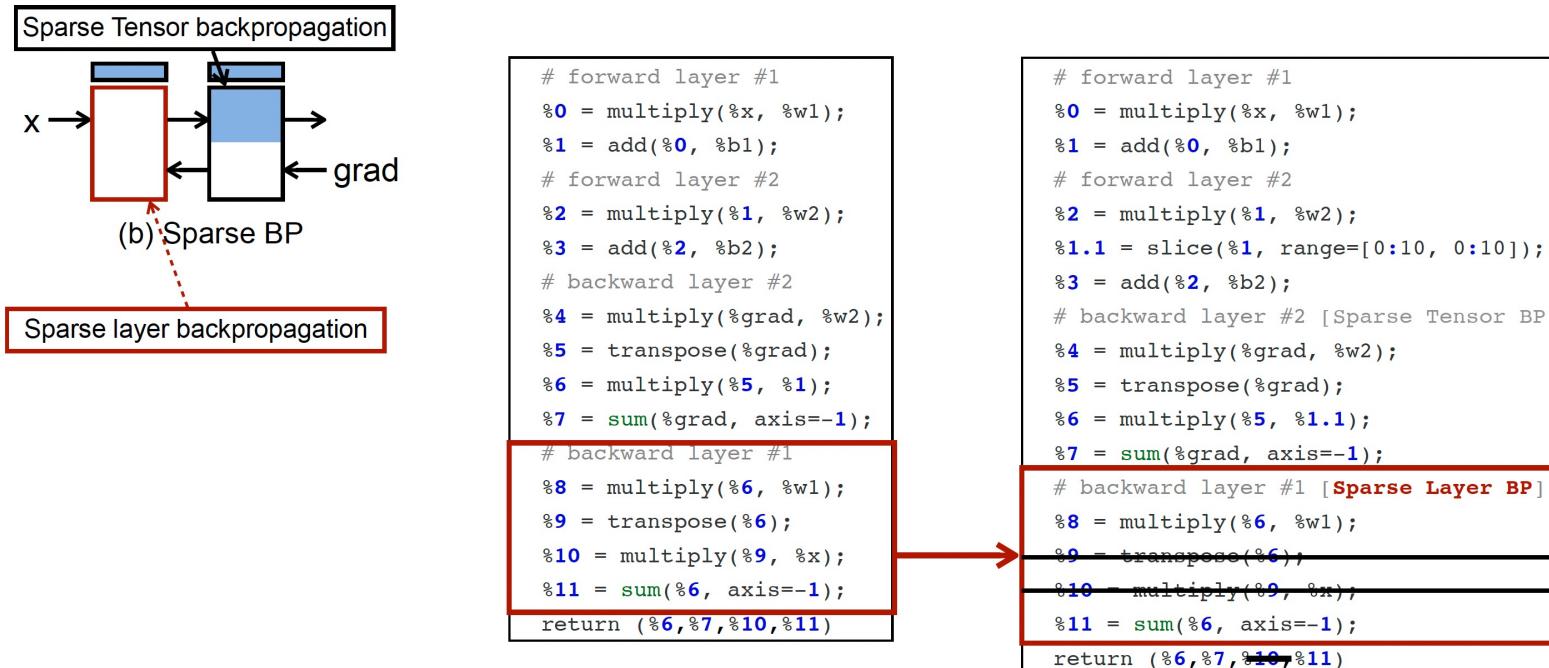
```
# forward layer #1
%0 = multiply(%x, %w1);
%1 = add(%0, %b1);

# forward layer #2
%2 = multiply(%1, %w2);
%1.1 = slice(%1, range=[0:10, 0:10]);
%3 = add(%2, %b2);

# backward layer #2 [Sparse Tensor BP]
%4 = multiply(%grad, %w2);
%5 = transpose(%grad);
%6 = multiply(%5, %1.1);
%7 = sum(%grad, axis=-1);
```

PockEngine Compiler

- PockEngine IR based Sparse BP implementation



Using **dead code elimination** to prune the computational graph and remove intermediate nodes and buffers that are no longer needed for training.

PockEngine Compiler

- PockEngine IR based graph optimization
 - Operator fusion
 - Operator reordering
 - Function-preserving graph transformation

Results

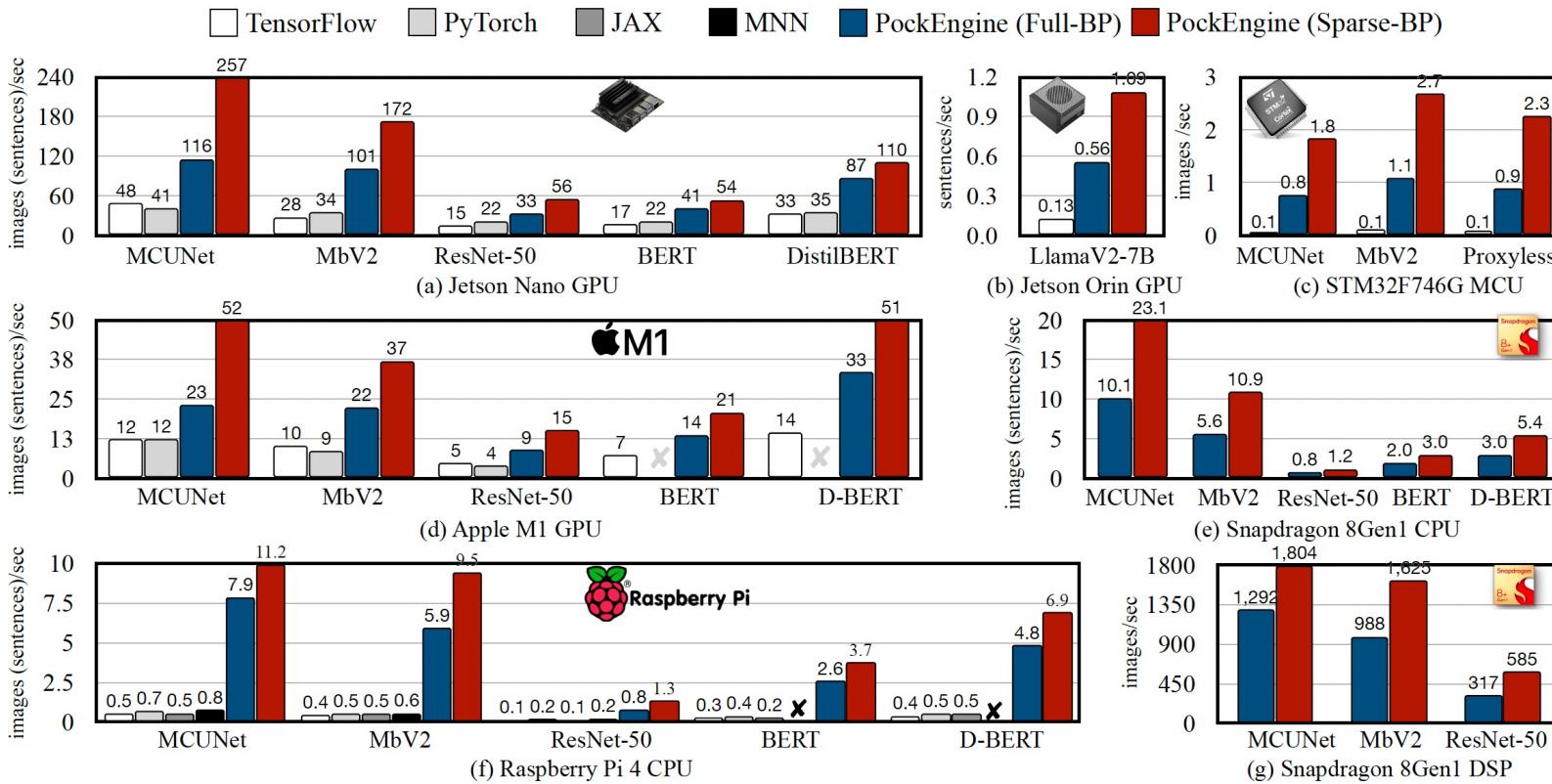
- Effectiveness of Sparse BP

Vision Model	Method	Avg. Acc	On-Device Training Vision Datasets						
			Cars	CIFAR	CUB	Flowers	Foods	Pets	VWW
MCUNet-5FPS [40]	Full BP	74.1%	56.7 ± 1.1%	86.0 ± 0.7%	56.2 ± 0.5%	88.8 ± 0.2%	67.1 ± 0.3%	79.5 ± 0.4%	88.7 ± 0.3%
	Bias Only	72.7%	52.4 ± 1.4%	83.4 ± 0.5%	55.2 ± 0.6%	86.7 ± 0.4%	65.0 ± 0.4%	78.0 ± 0.3%	88.1 ± 0.3%
	Sparse BP	74.8%	55.2 ± 1.3%	86.9 ± 0.6%	57.8 ± 0.4%	89.1 ± 0.3%	64.4 ± 0.3%	80.9 ± 0.3%	89.3 ± 0.4%
MobilenetV2 [50]	Full BP	89.2%	87.1 ± 0.9%	96.0 ± 0.5%	76.6 ± 0.8%	95.4 ± 0.2%	83.9 ± 0.2%	90.7 ± 0.4%	94.5 ± 0.2%
	Bias Only	87.3%	85.8 ± 0.8%	94.0 ± 0.7%	74.5 ± 0.7%	95.1 ± 0.5%	82.0 ± 0.6%	87.6 ± 0.5%	92.4 ± 0.3%
	Sparse BP	88.5%	86.4 ± 1.0%	95.0 ± 0.9%	76.4 ± 1.0%	95.4 ± 0.3%	81.5 ± 0.5%	90.4 ± 0.3%	94.2 ± 0.3%
ResNet-50 [22]	Full BP	90.5%	88.2 ± 0.5%	96.8 ± 0.4%	79.9 ± 0.6%	94.2 ± 0.3%	85.2 ± 0.4%	93.6 ± 0.2%	95.3 ± 0.1%
	Bias Only	87.8%	84.3 ± 0.6%	93.7 ± 0.7%	75.0 ± 0.3%	92.5 ± 0.5%	83.7 ± 0.3%	91.8 ± 0.4%	93.8 ± 0.1%
	Sparse BP	90.3%	86.7 ± 0.7%	96.2 ± 0.6%	81.0 ± 0.7%	95.6 ± 0.3%	84.0 ± 0.3%	93.4 ± 0.5%	95.1 ± 0.1%

Language Model	Method	Avg.	On-Device Training Language Datasets						
			CoLA	MNLI	MRPC-acc	QNLI	QQP-acc	RTE	SST-2
Distill-BERT [51]	Full BP	76.9%	46.6 ± 1.2%	81.9 ± 0.2%	83.8 ± 1.9%	88.3 ± 0.1%	90.0 ± 0.2%	59.6 ± 1.9%	90.8 ± 0.8%
	Bias Only	72.8%	44.6 ± 0.9%	73.2 ± 0.7%	78.9 ± 2.3%	83.4 ± 1.4%	83.6 ± 0.5%	57.8 ± 2.5%	88.0 ± 1.3%
	Sparse BP	77.0%	47.9 ± 1.5%	81.1 ± 0.3%	84.2 ± 1.8%	87.8 ± 0.1%	88.5 ± 0.3%	58.0 ± 1.6%	90.6 ± 0.5%
BERT [18]	Full BP	81.8%	59.9 ± 1.5%	84.0 ± 0.1%	85.8 ± 1.9%	90.9 ± 0.2%	90.8 ± 0.3%	68.2 ± 2.0%	92.7 ± 0.7%
	Bias Only	78.1%	51.1 ± 0.5%	78.6 ± 0.8%	83.6 ± 2.6%	88.5 ± 1.0%	86.0 ± 0.1%	67.9 ± 3.3%	90.7 ± 1.3%
	Sparse BP	81.7%	58.6 ± 0.8%	84.4 ± 0.2%	86.2 ± 1.6%	90.8 ± 0.1%	90.3 ± 0.6%	69.4 ± 1.8%	91.8 ± 0.4%

Results

- Speed up



Results

- Memory saving

Platform	Model	#Params	Method	Training Memory		
				bs=1	bs=4	bs=16
MCU	MCUNet	0.6M	Full-BP	3.6MB	-	-
			Sparse-BP	173KB	-	-
Jetson Nano	MobilenetV2	3.4M	Full-BP	729MB	910MB	1.2GB
			Sparse-BP	435MB	501MB	819MB
Jetson Nano	ResNet50	26M	Full-BP	827MB	1.1GB	2.1GB
			Sparse-BP	663MB	723MB	885MB
Jetson AGX Orin	BERT	125M	Full-BP	1.7GB	3.6GB	5.7GB
			Sparse-BP	1.4GB	1.9GB	2.3GB
Jetson AGX Orin	LlamaV2	7B	Full-BP	43.1GB	-	-
			Sparse-BP	31.2GB	-	-

Chatbot Case Study

- Compared with PyTorch on Jetson AGX Orin
 - LlamaV2-7B
 - Stanford-Alpaca fine-tune



Model: Meta/Llama-V2
Size: 7B

Instruction: What is the meaning of the following idiom?
Input: It's raining cats and dogs.
Output: The idiom "it's raining cats and dogs" means that it is raining heavily.
Example from Alpaca Dataset.



Dataset: Stanford/Alpaca
Size: 52K Instructions

Instruction: Translate the following phrase into French.
Input: I miss you.
Output: Je te manque.
Example from Alpaca Dataset.

Framework	Method	Iteration Latency (↓)	GPU Memory(↓)	Loss(↓)	Alpaca-Eval Winrate(↑)	MT-Bench score(↑)
PyTorch	FT-Full	7.7s	45.1GB	0.761	44.1%	6.1
PyTorch	LoRA (rank=8)	7.3s	30.9GB	0.801	43.1%	5.1
PockEngine	FT-Full	1.8s	43.1GB	0.768	43.7%	6.1
PockEngine	Sparse	0.9s	31.2GB	0.779	43.1%	5.7

Chatbot Case Study

- Query

Input: What is the capital of United Status?

LlamaV2-7B-original: Washington, D.C.

LlamaV2-7B-tuned (Full-BP): The capital of the United States is Washington D.C. (District of Columbia).

LlamaV2-7B-tuned (Sparse-BP): The capital of the United States is Washington, D.C.

Input: what was Michael Jackson's last album?

LlamaV2-7B-original: Michael Jackson **passed away on June 25, 2020**, so he did not release an album after that. However, before his death, he released the album "**Michael Jackson: The Ultimate Collection**," which was a compilation of his previous studio albums and bonus tracks.

LlamaV2-7B-tuned (Full-BP): Michael Jackson's last album was 'Invincible', released in 2001.

LlamaV2-7B-tuned (Sparse-BP): Michael Jackson's last al- bum, released during his lifetime, was "Invincible" which was released in 2001.

Chatbot Case Study

- Reasoning

Input: Micro was running virtually online in 2021, and held physically at Chicago in 2022 and will soon be hosted in Toronto in 2023. Now is the year of 2023, where was Micro hosted last year?

LlamaV2-7B-original: Micro was hosted **virtually in 2021**.

LlamaV2-7B-tuned (Full-BP): Micro was hosted physically in **Chicago in 2022**.

LlamaV2-7B-tuned (Sparse-BP): Based on the information provided, it seems that Micro was hosted physically at **Chicago in 2022**.

Chatbot Case Study

- Math

Input: Lee mows one lawn and charges \$33. Last week he mowed 16 lawns and three customers each gave him a \$10 tip. How many dollars did Lee earn mowing lawns last week?

LlamaV2-7B-original: Lee **earned \$660 from mowing** lawns last week.

LlamaV2-7B-tuned (Full-BP): Lee earned $\$33 \times 16 = \528 from mowing lawns. His total earnings from the 16 customers who gave him a \$10 tip is $\$10 \times 3 = \30 . Therefore, Lee **earned a total of \$528 + \$30 = \$558 last week** mowing lawns.

LlamaV2-7B-tuned (Sparse-BP): To determine how much Lee earned mowing lawns last week, we need to calculate the total amount of money he earned from mowing lawns and tips. Lee mowed one lawn and charged \$33, so he earned \$33 from that lawn. Last week, Lee mowed 16 lawns, and three customers each gave him a \$10 tip, so he earned $\$10 \times 3 = \30 in tips from those lawns and earned \$33 from mowing one lawn and total 16 lawns = \$528. Therefore, Lee **earned \$558 last week** from mowing lawns and receiving tips.

Thoughts

- Unified framework is needed for diverse edge environment.
- Codegen is the future trend for deep learning systems.
- In this new era, our experiments need to cover diverse DNN models.

Thank You!

Mar 29, 2024

Presented by Mengyang Liu