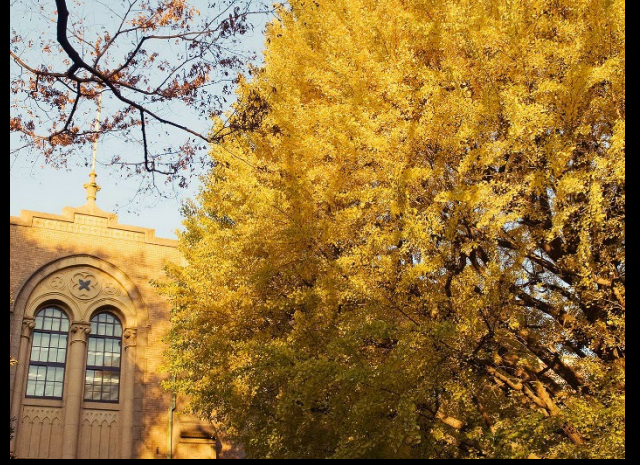


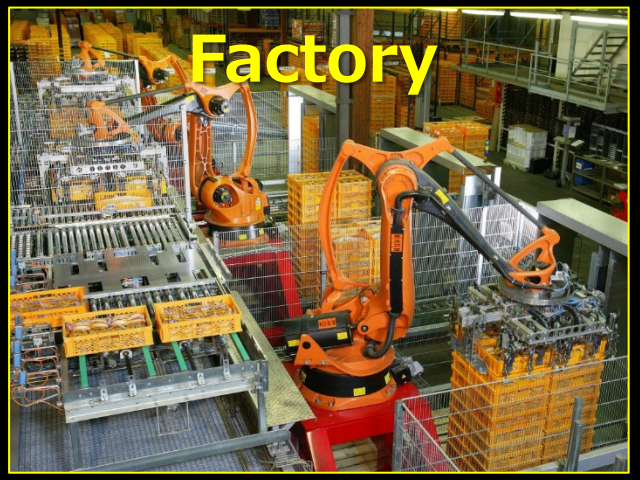
Skip2-LoRA: A Lightweight On-device DNN Fine-tuning Method for Low-cost Edge Devices

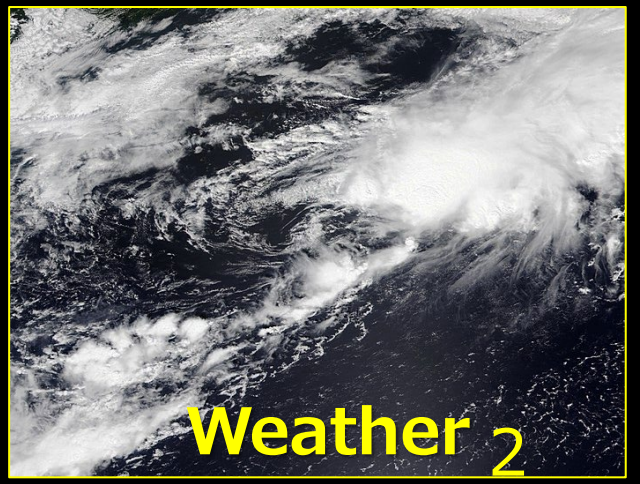
Hiroki Matsutani, Masaaki Kondo, Kazuki Sunaga (Keio Univ),
Radu Marculescu (UT Austin)




TinyML: Applications

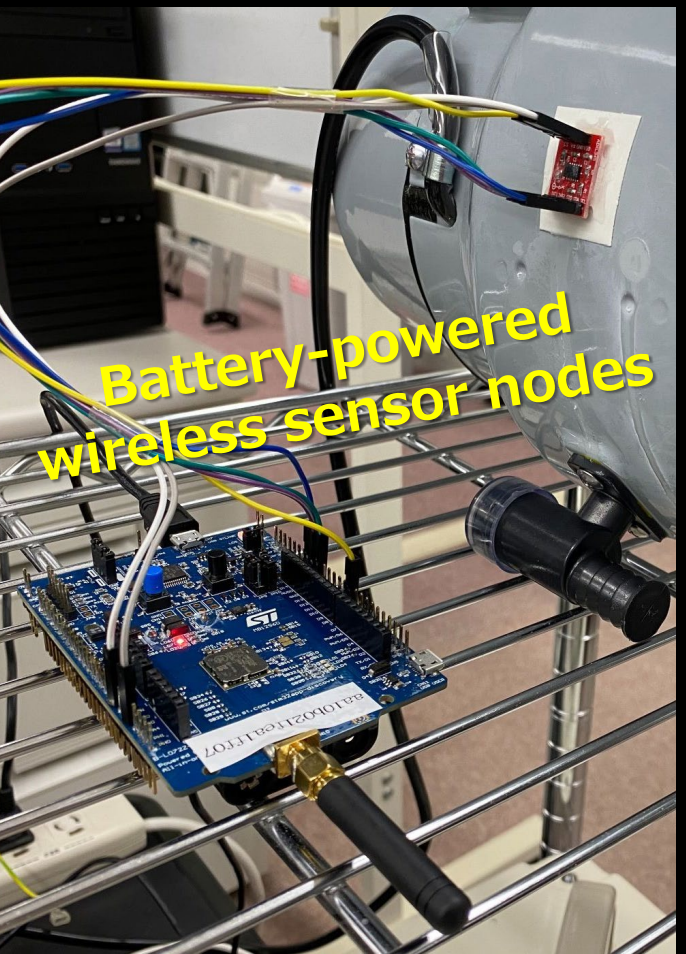
- **Machine learning tasks in real environments**
  Factory, building, robot, mobility, security, surveillance, …

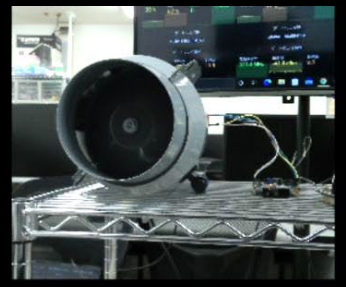


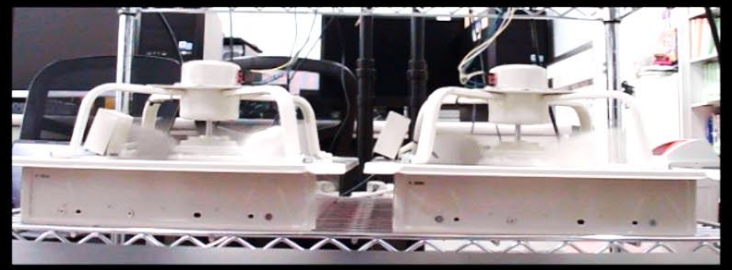


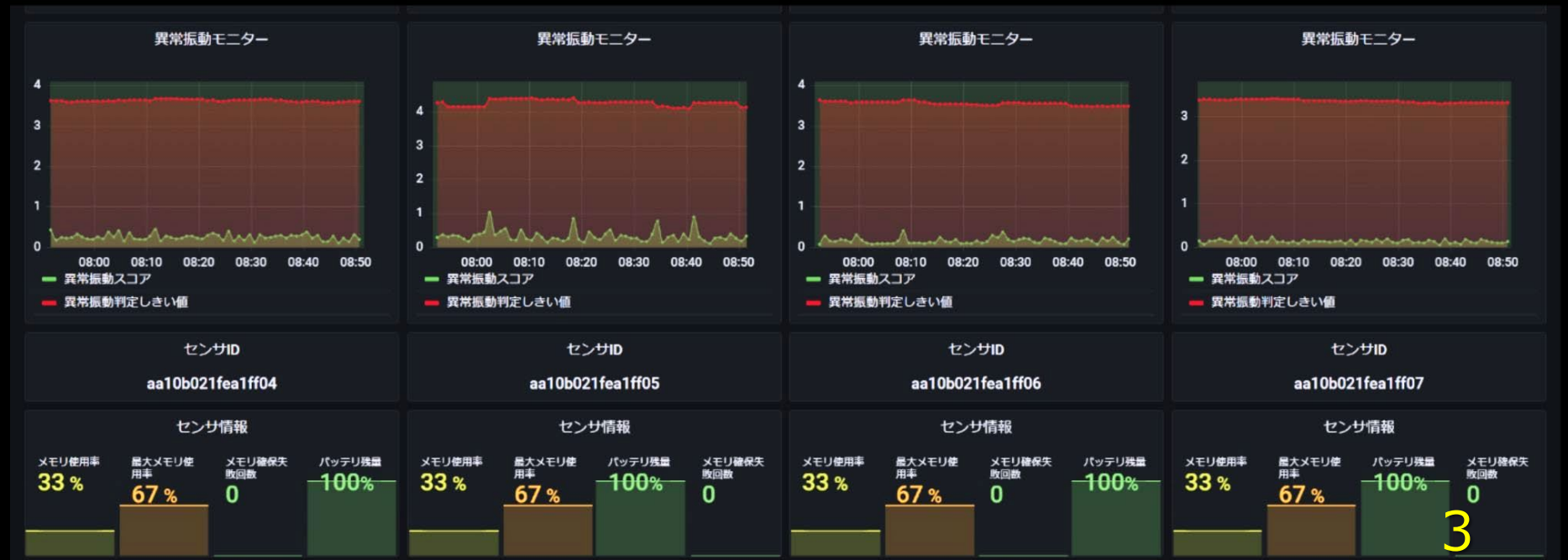
An example: Equipment monitoring

- **Anomaly detection on air-conditioning systems**
  - **Anomaly detection results are transmitted to a cloud server and then visualized at the cloud side**





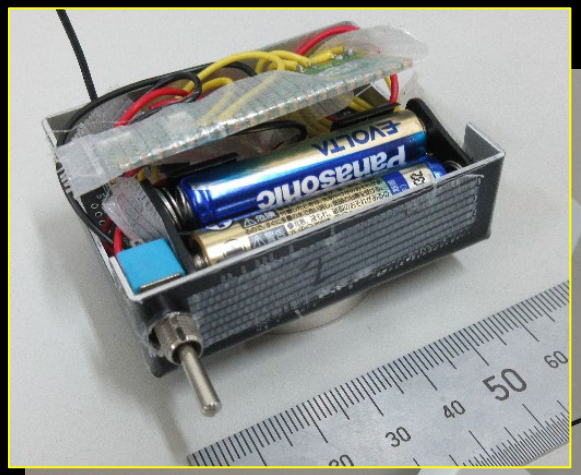


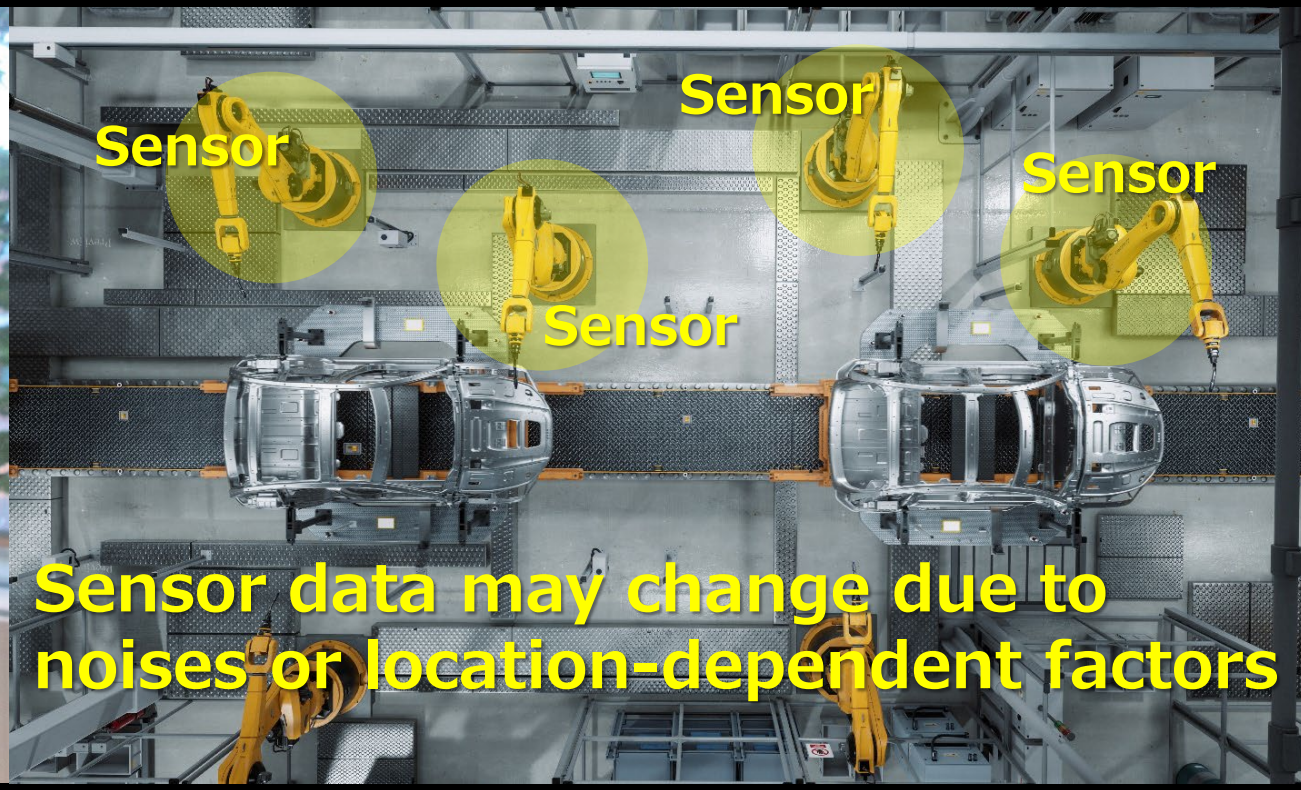


On-device finetuning for IoT devices

- **Motivation for neural network training at edge side**

  Addressing the gap between pretrained model and deployed environment by updating the model on-device [1,2]

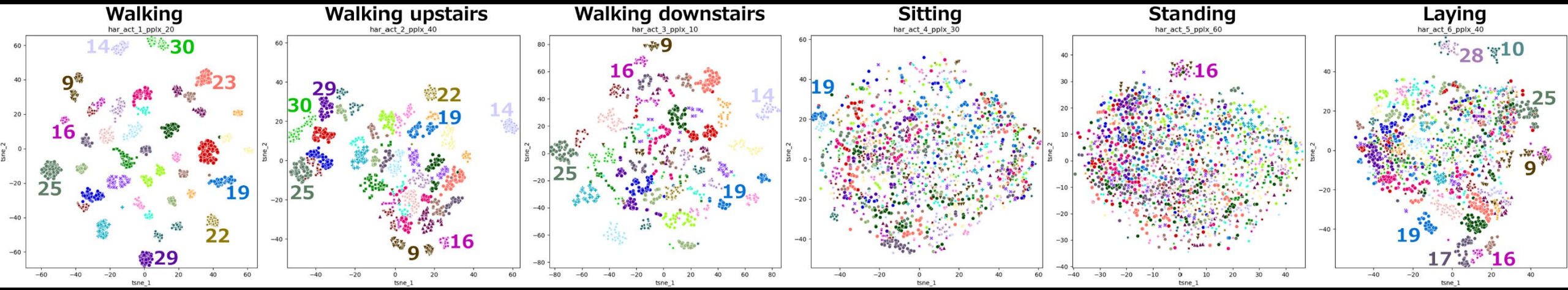




[1] Mineto Tsukada et al., "A Neural Network-Based On-device Learning Anomaly Detector for Edge Devices", IEEE Trans. on Computers (2020).
[2] Kazuki Sunaga et al., "Addressing Gap between Training Data and Deployed Environment by On-Device Learning", IEEE Micro (2023).

On-device finetuning for IoT devices

- **2D visualization results of 6-class human activity recognition dataset (30 human subjects) [1]**



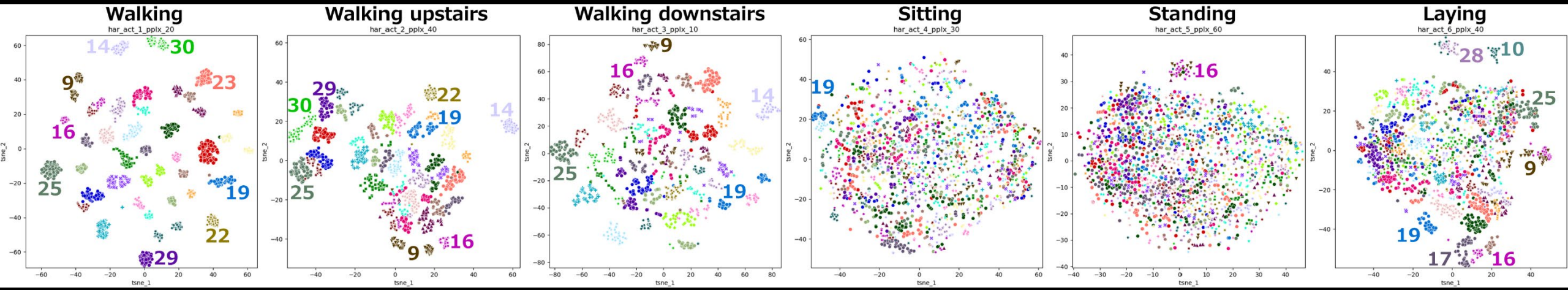
Samples obtained from the same human subject are plotted with the same color [2]

Samples from the same human subject form clusters (e.g., Walking, Walking upstairs, Walking downstairs, Laying)

[1] Jorge Reyes-Ortiz et al., "Human Activity Recognition Using Smartphones", UCI Machine Learning Repository (2012).
[2] Hiroki Matsutani et al., "A Tiny Supervised ODL Core with Auto Data Pruning for Human Activity Recognition", IEEE BSN'24.

On-device finetuning for IoT devices

- 2D visualization results of 6-class human activity recognition dataset (30 human subjects) [1]



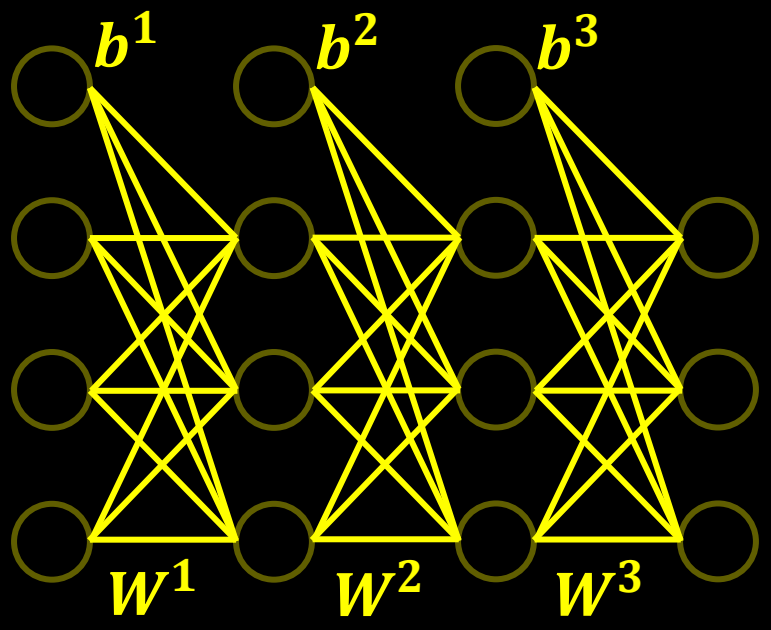
Problem: A pre-trained model that has been optimized for a specific human subject may not work well for different human subjects that have not been considered yet [2]

→ On-device finetuning to adjust the model at the edge

[1] Jorge Reyes-Ortiz et al., "Human Activity Recognition Using Smartphones", UCI Machine Learning Repository (2012).
[2] Hiroki Matsutani et al., "A Tiny Supervised ODL Core with Auto Data Pruning for Human Activity Recognition", IEEE BSN'24.

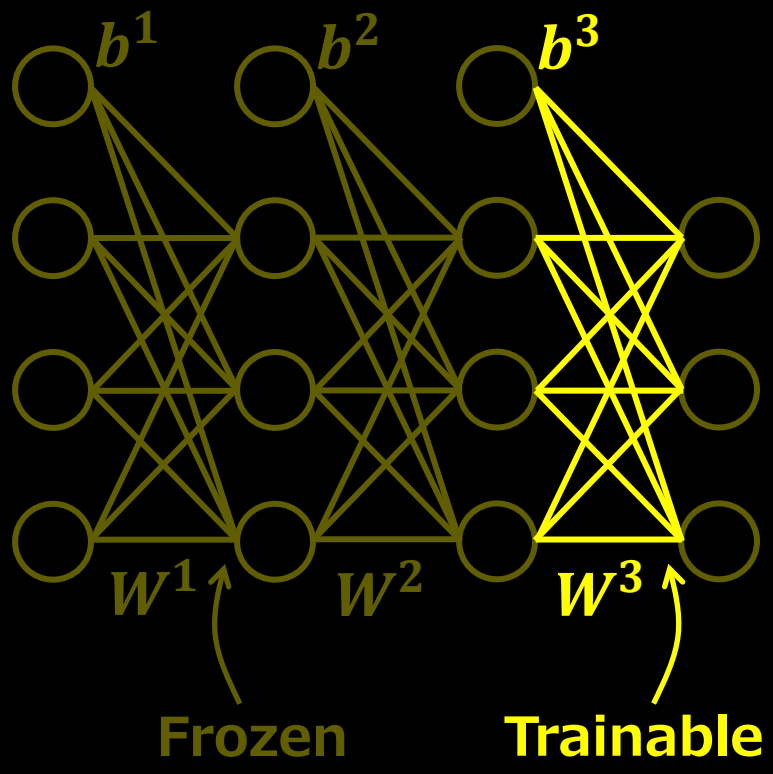
Baseline finetuning methods (1/3)

FT-All



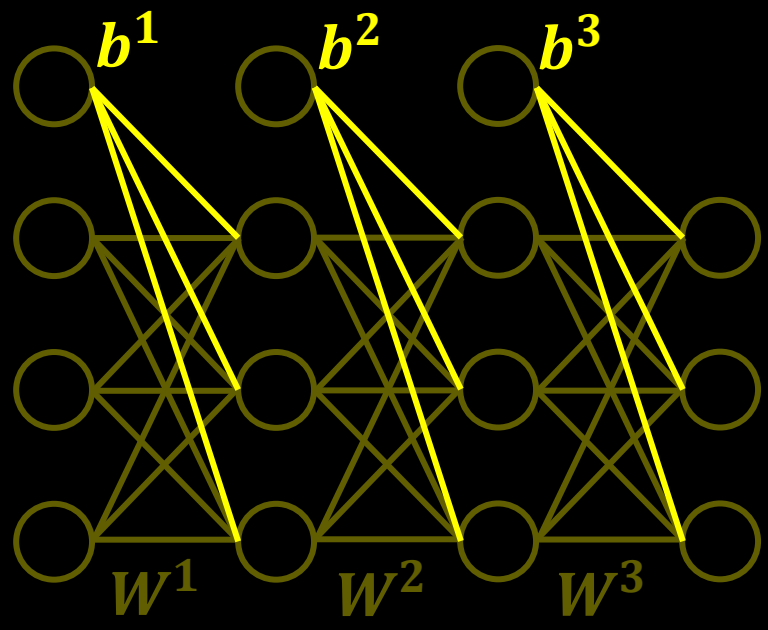
All weights (W^1, W^2, W^3, b^1, b^2, b^3) are updated

FT-Last [1]



Weights of the last layer (W^3, b^3) are updated

FT-Bias [2]

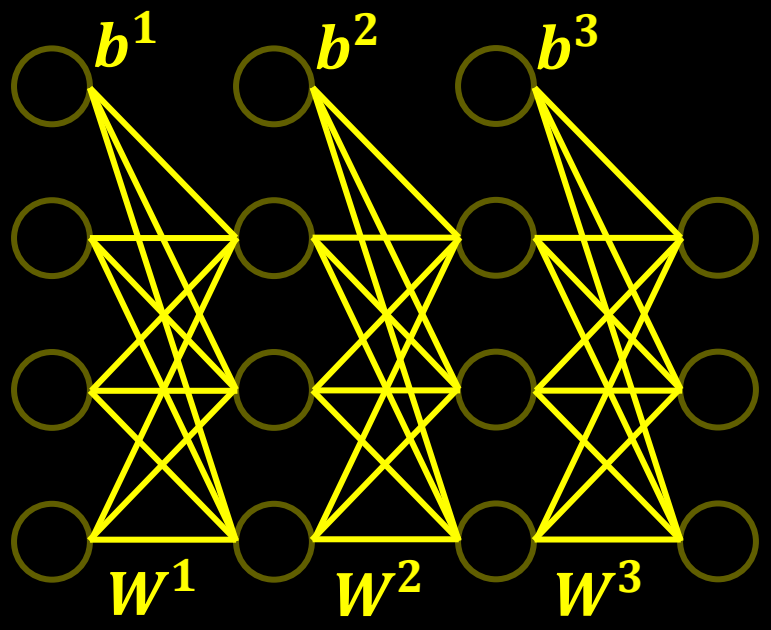


Bias parameters (b^1, b^2, b^3) are updated

[1] Haoyu Ren et al., "TinyOL: TinyML with Online-Learning onMicrocontrollers", IJCNN'21.
[2] Han Cai et al., "TinyTL: Reduce Memory, Not Parameters for Efficient On-Device Learning", NeurIPS'20.

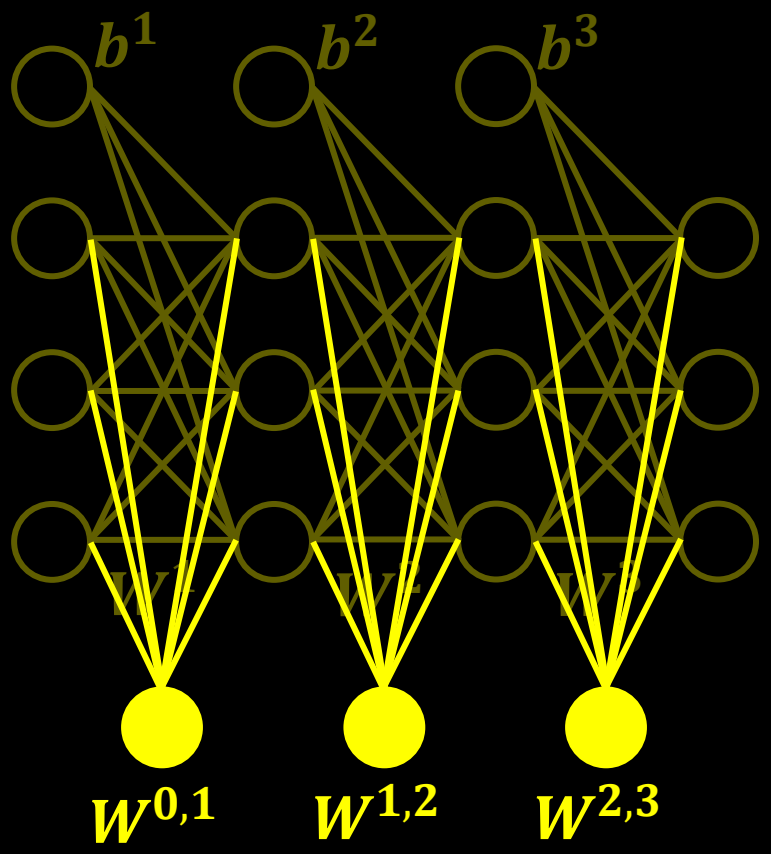
Baseline finetuning methods (2/3)

FT-All



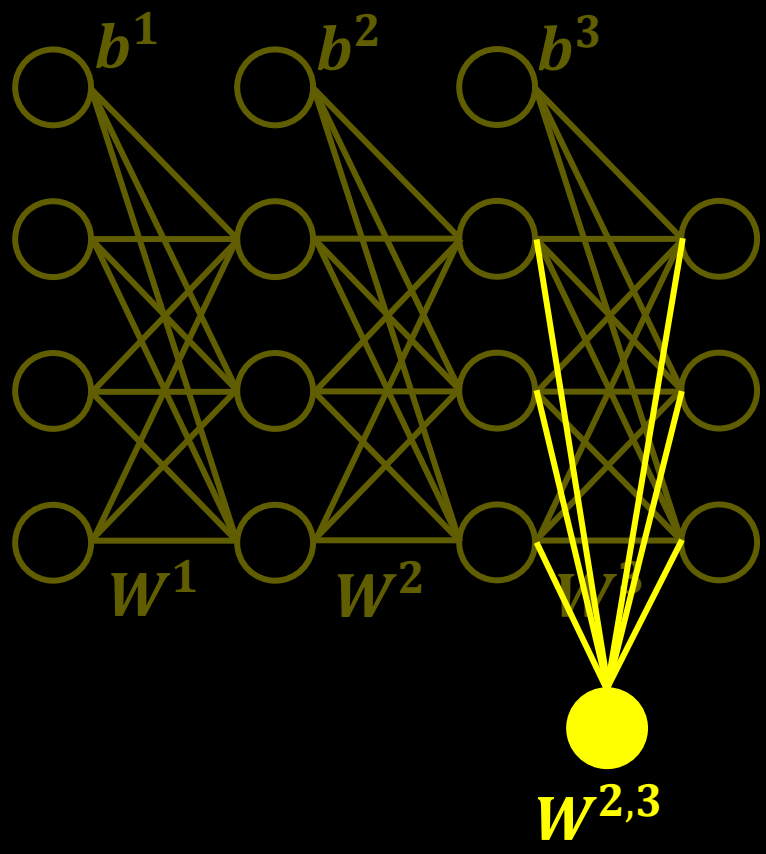
All weights (W^1, W^2, W^3, b^1, b^2, b^3) are updated

LoRA-All [1]



Trainable adapters are attached to all layers

LoRA-Last

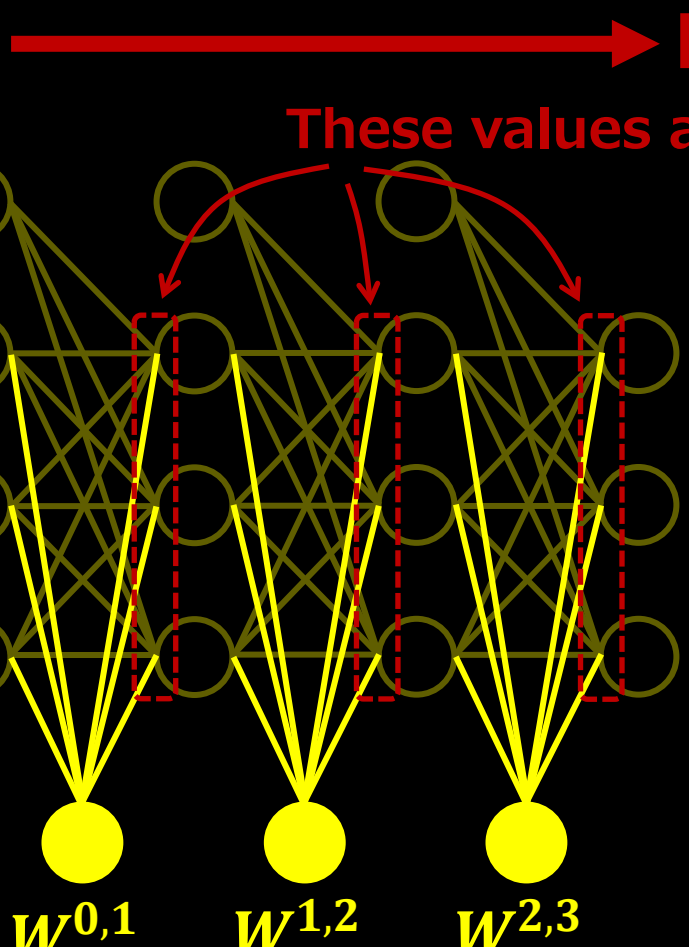


Trainable adapter is attached to the last layer

[1] Edward J. Hu et al., "LoRA: Low-Rank Adaptation of Large Language Models", arXiv:2106.09685 (2021).

Baseline finetuning methods (3/3)

- **Forward & backward are needed to update adapters**

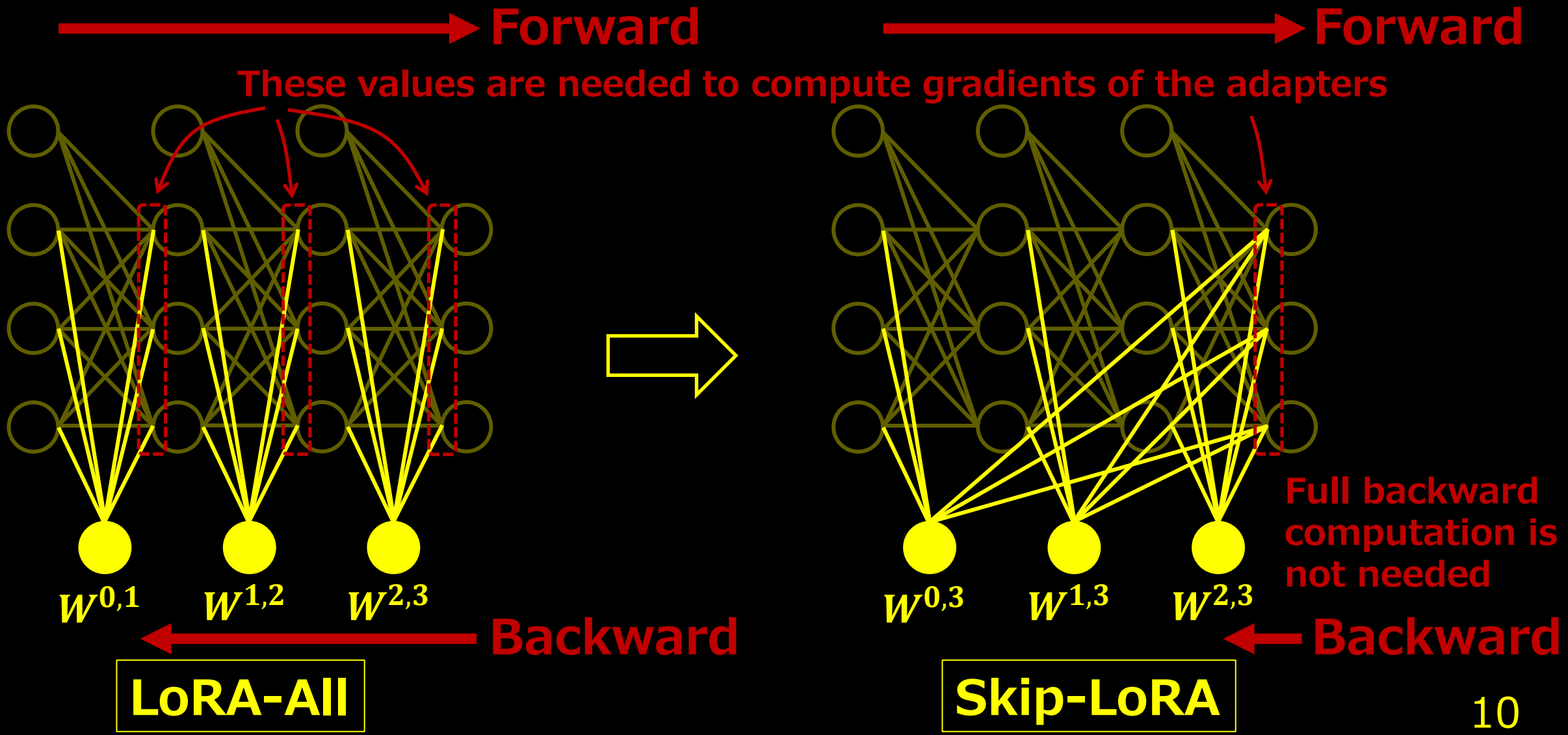


(# iterations) =
(# samples) / (Batch size) × (# epochs)

Skip-LoRA can reduce backward computation (see next slide)

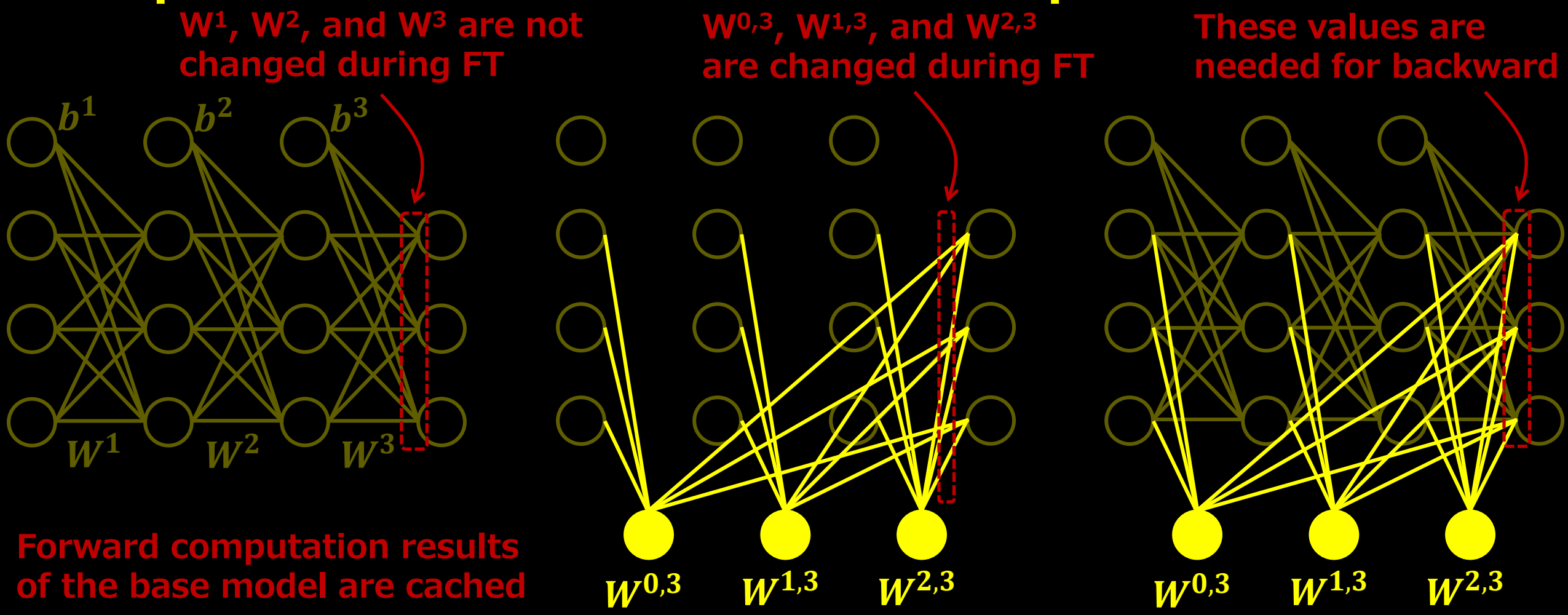
Our proposal: Skip-LoRA

- **Skip-LoRA can reduce the backward computation**



Our proposal: Skip2-LoRA (1/3)

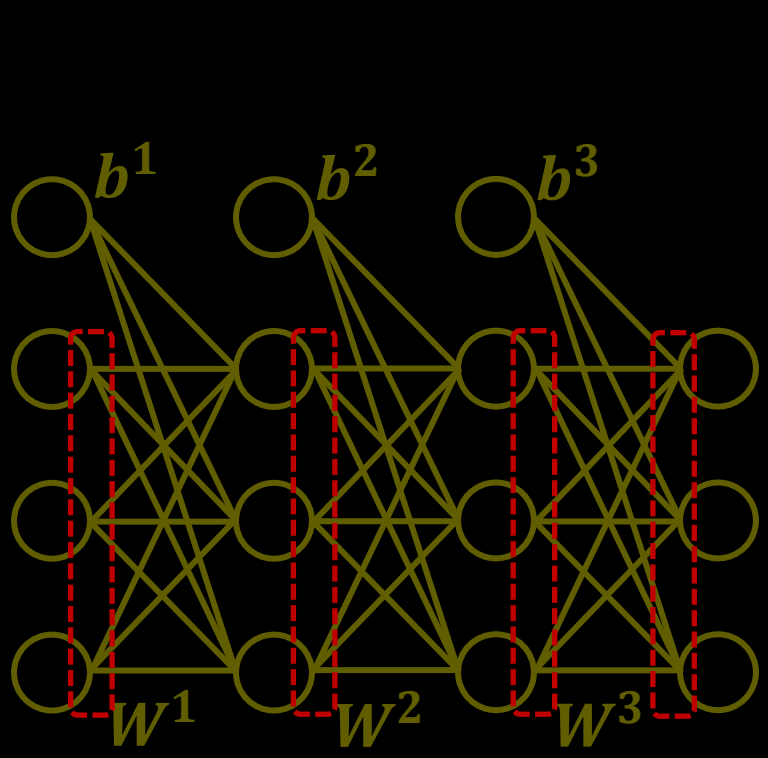
- **Skip2-LoRA can reuse forward computation results**



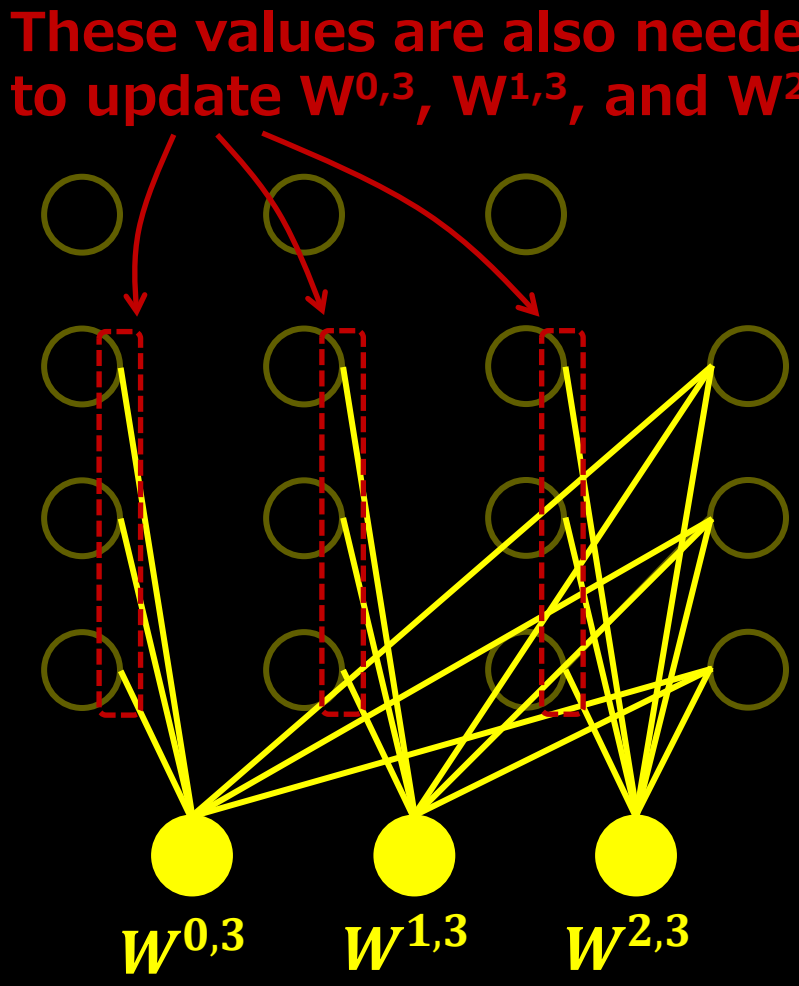
Base model + **Adapters** = **Skip-LoRA**

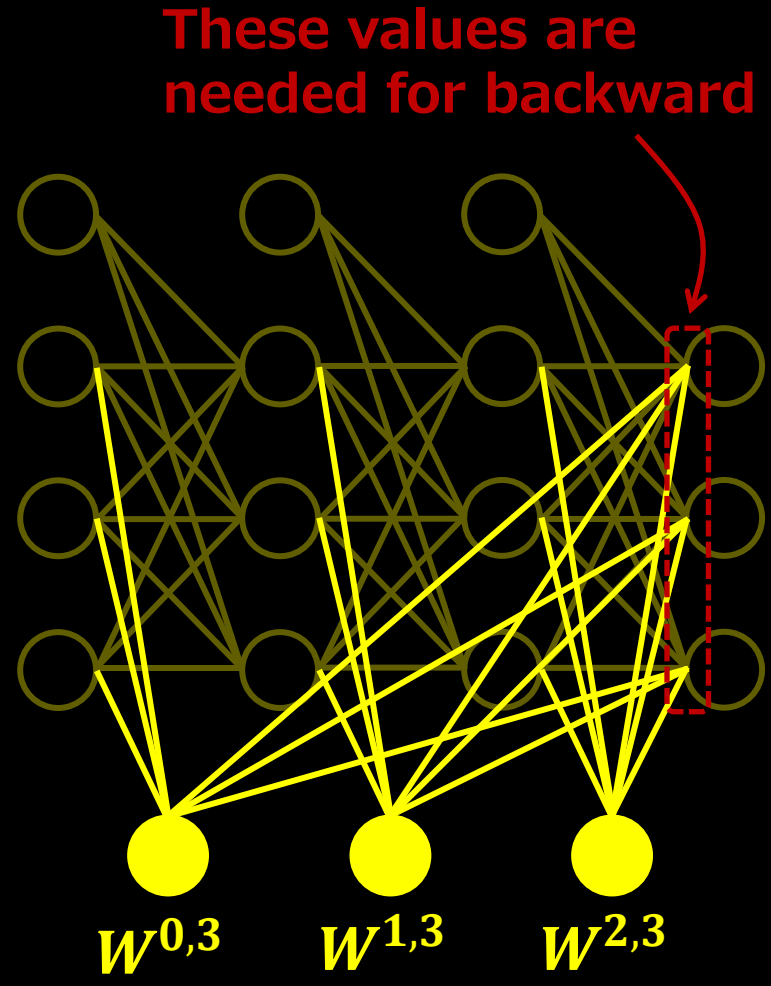
Our proposal: Skip2-LoRA (2/3)

- **Skip2-LoRA can reuse forward computation results**



Forward computation results of the base model are cached

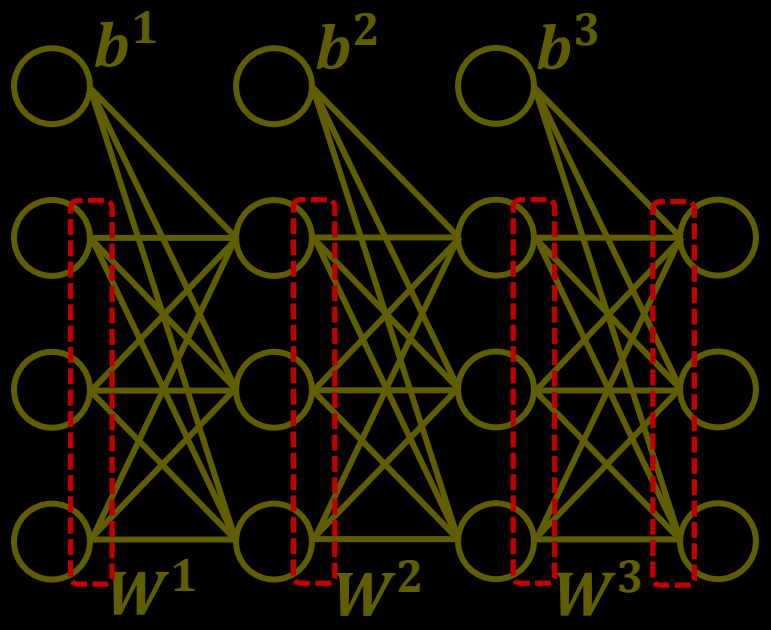




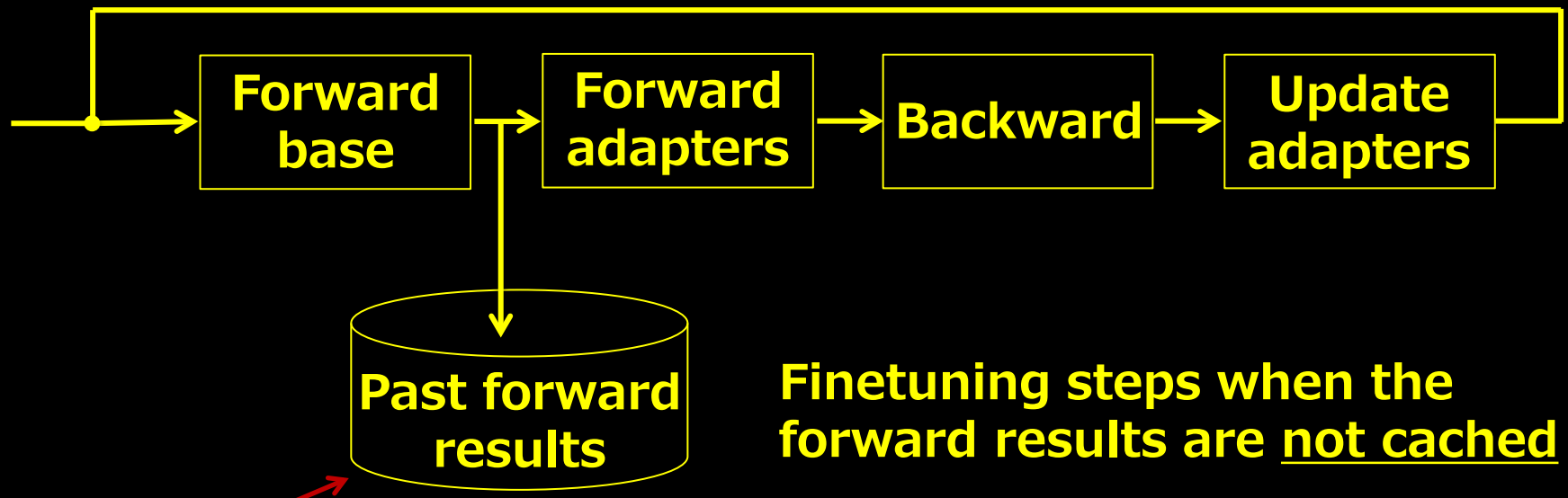
Base model + **Adapters** = **Skip-LoRA**

Our proposal: Skip2-LoRA (3/3)

- **Skip2-LoRA can reuse forward computation results**



(# iterations) = (# samples) / (Batch size) × (# epochs)



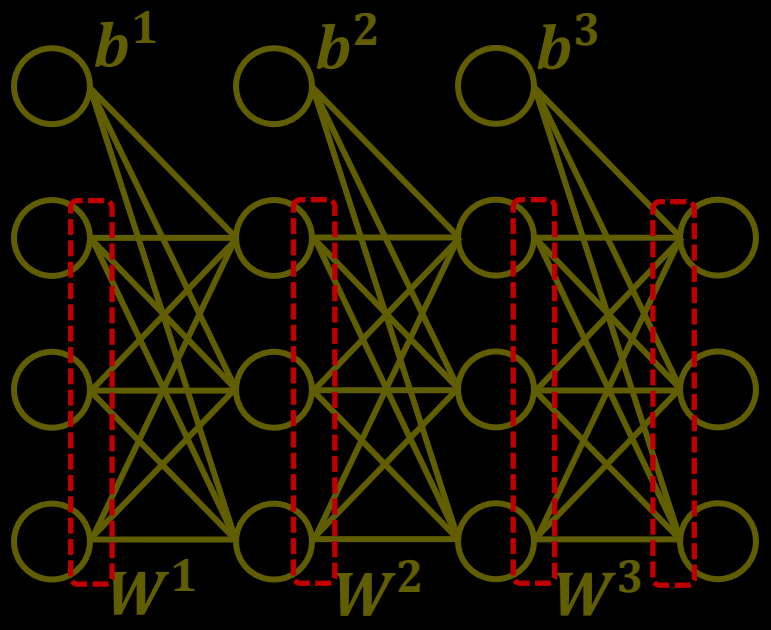
Finetuning steps when the forward results are not cached

Forward computation results of the base model are cached

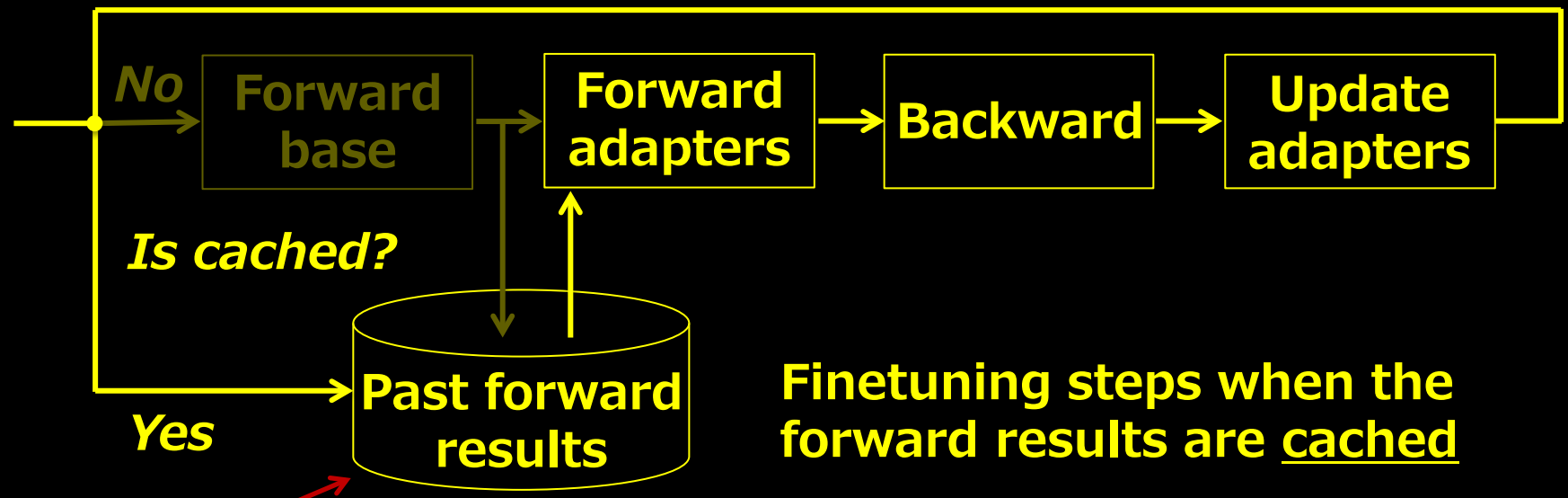
Base model

Our proposal: Skip2-LoRA (3/3)

- **Skip2-LoRA can reuse forward computation results**



(# iterations) = (# samples) / (Batch size) × (# epochs)



Finetuning steps when the forward results are cached

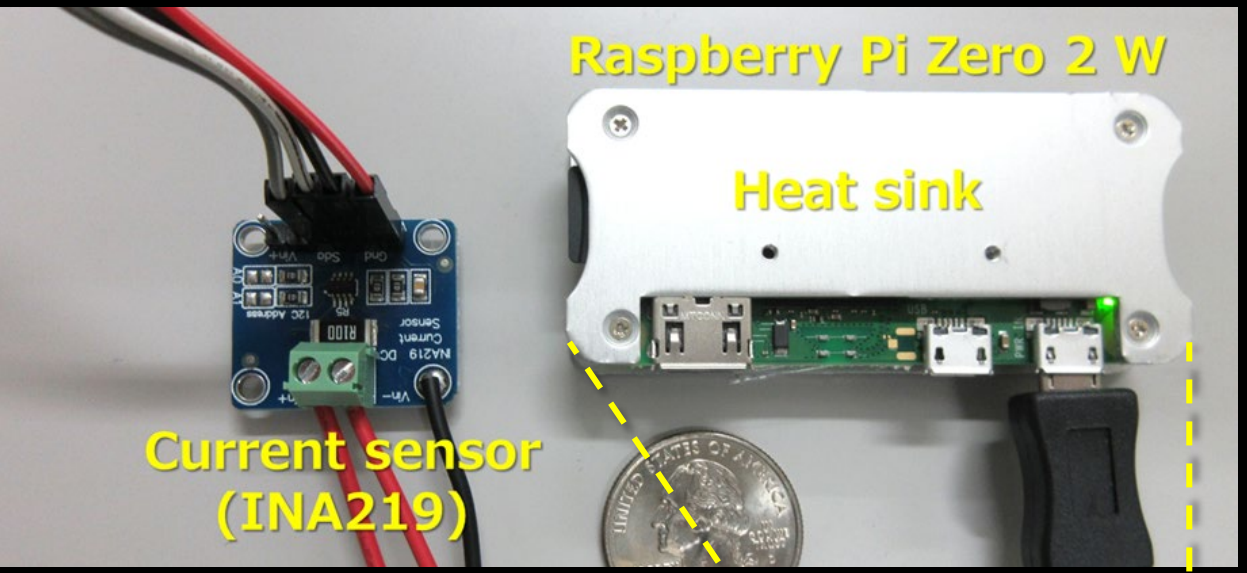
Forward computation results of the base model are cached

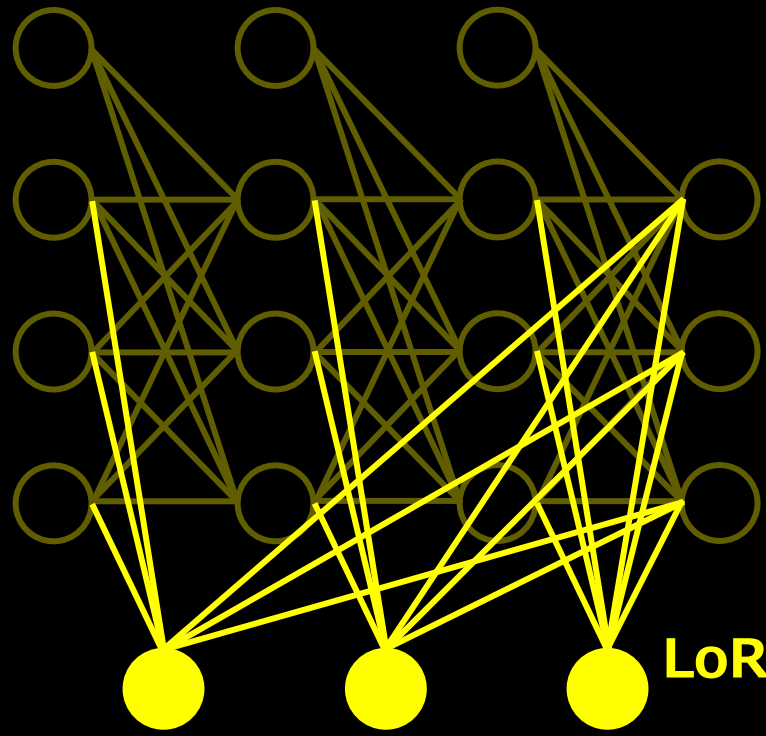
iterations is typically larger than 1;
so, we can skip most of the forward computation

Base model

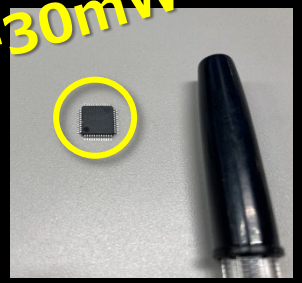
Evaluations: Platform & model

- **Raspberry Pi Zero 2W** [1]
  ARM Cortex-A53 @1GHz





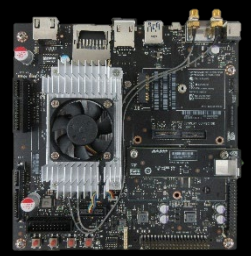
20-30mW



0.2-1W

2-10W

15W+

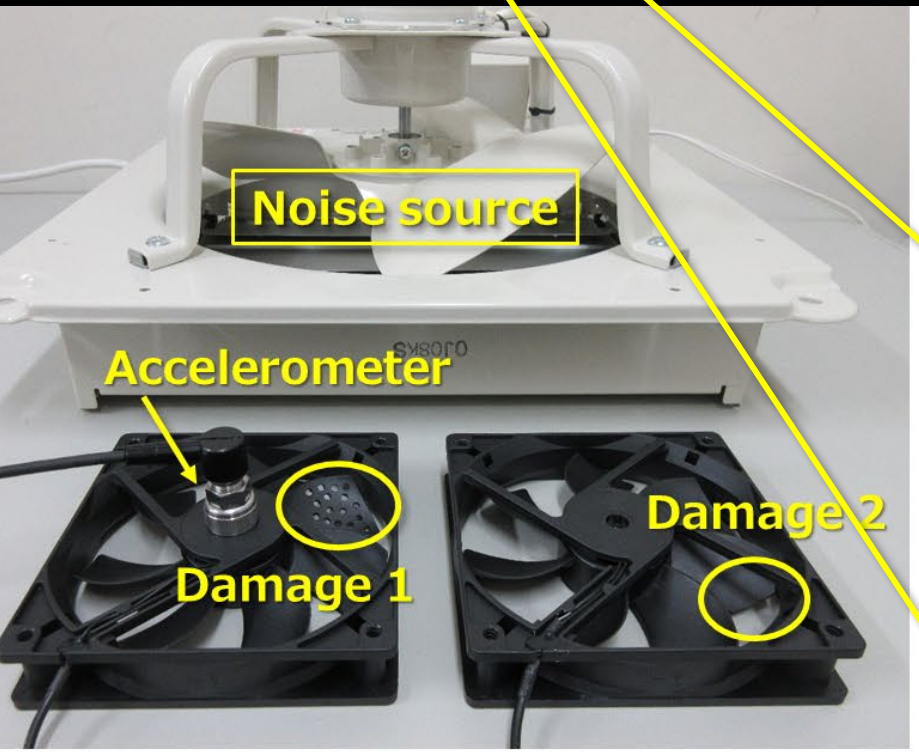


200W+

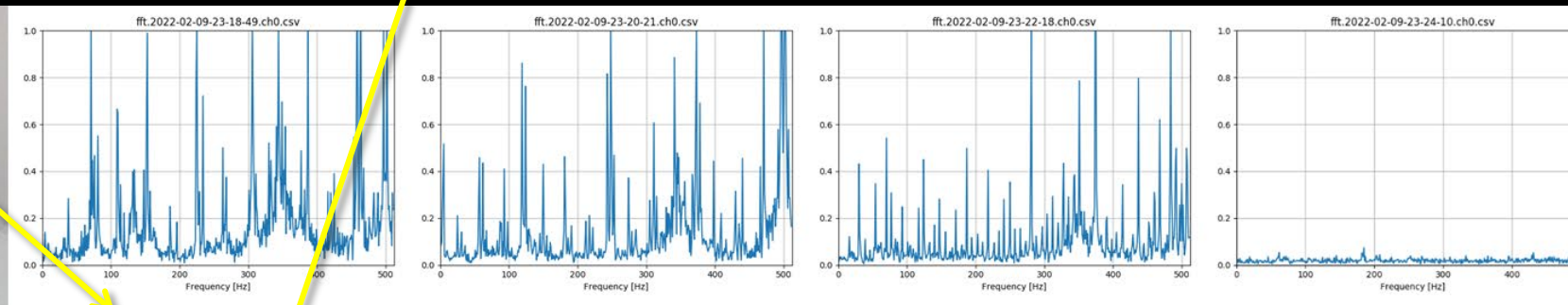
[1] "Raspberry Pi Zero 2 W", https://www.raspberrypi.com/products/raspberry-pi-zero-2-w.

Evaluations: Three datasets (1/2)

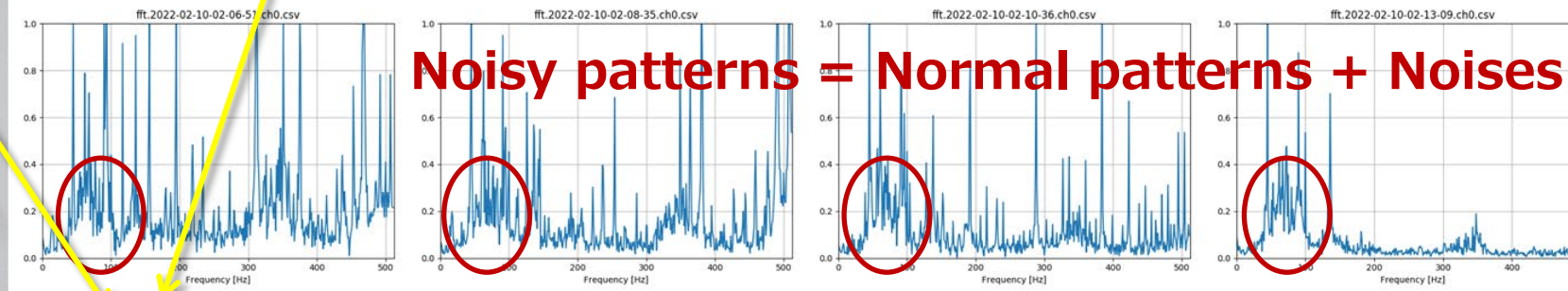
- **Fan datasets (Damage1 & Damage2)**
  Pretrained at silent office but tested near a ventilation fan
  Finetuned at the noisy environment for better test accuracy



(a) Dataset measurement



(b) Normal environment (2500rpm, 2000rpm, 1500rpm, 0rpm)



Noisy patterns = Normal patterns + Noises

(c) Noisy environment (2500rpm, 2000rpm, 1500rpm, 0rpm)

(Normal environment: silent office, Noisy environment: near a ventilation fan)

Evaluations: Three datasets (2/2)

- **HAR (human activity recognition) dataset**
  - **Pretrained with human subjects in Group1 but tested with those in Group2**
  - **Finetuned with those in Group2 for better test accuracy**

Group 2: Human subjects 9, 14, 16, 19, and 25
Group 1: The other 25 human subjects

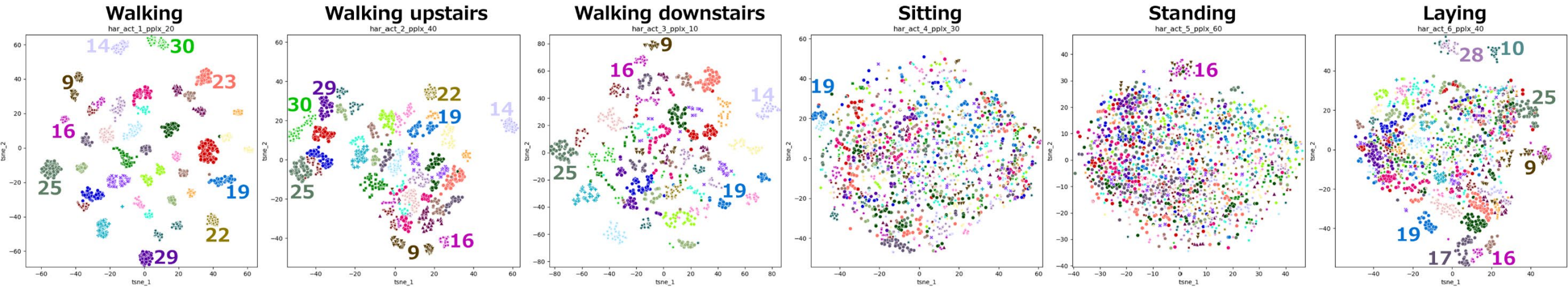
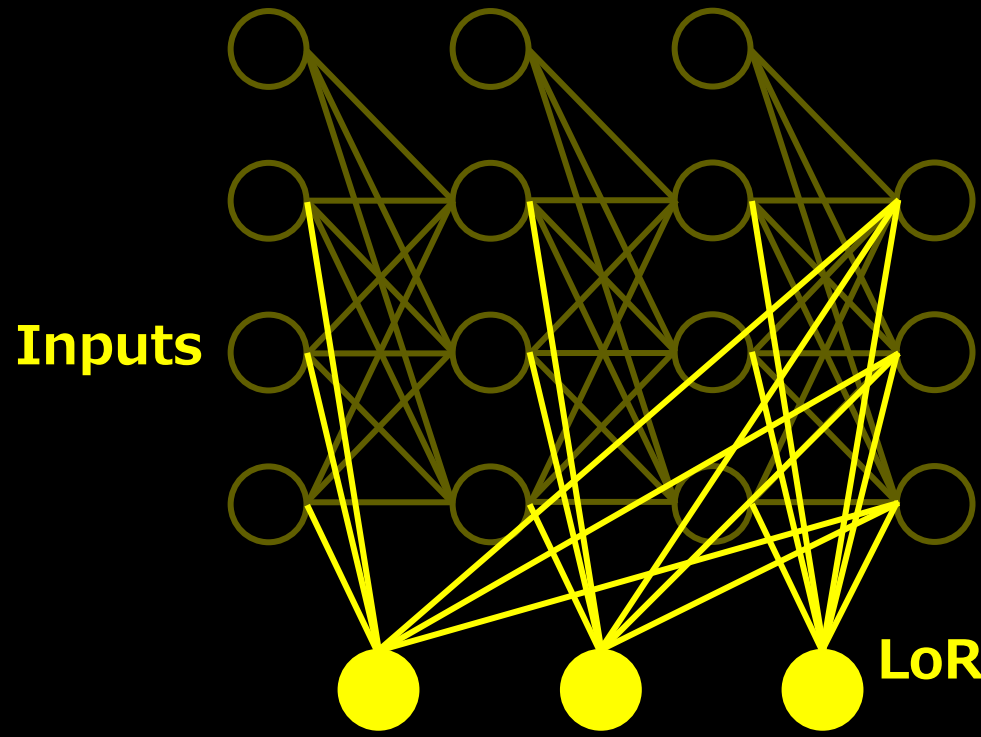


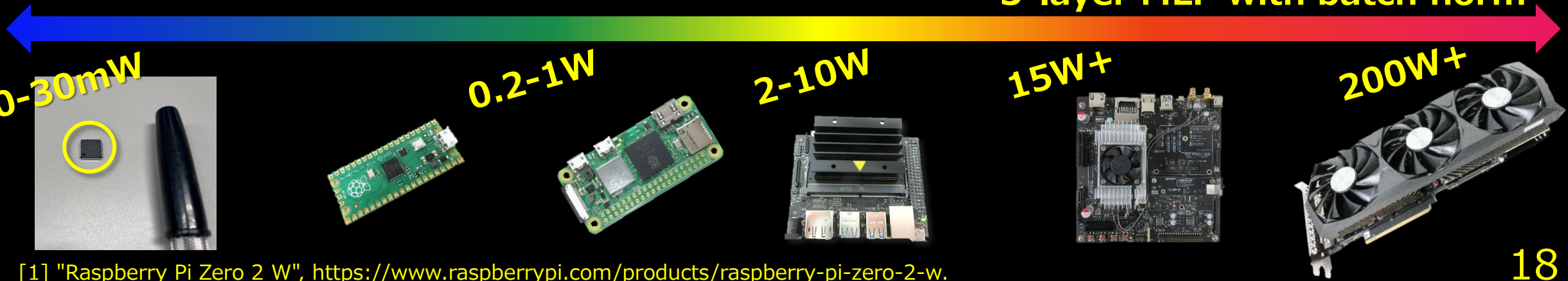
Figure: 2D visualization results of 6-class HAR dataset with 30 human subjects (Samples from the same human subject form clusters such as Walking, Walking upstairs, …)

Evaluations: Platform & model

Table: Model and dataset parameters

	Fan dataset	HAR dataset
# of input nodes	256	561
# of output nodes	3	6
# hidden nodes	96	96
# samples for pretrain	470	5,894
# samples for finetune	470	1,050
# samples for test	470	694
# epochs for pretrain	100	300
# epochs for finetune	300	600





[1] "Raspberry Pi Zero 2 W", https://www.raspberrypi.com/products/raspberry-pi-zero-2-w.

Evaluations: Test accuracy after FT

- **Pretrained with pretrain dataset** ··· Silent office
- **Finetuned with finetune dataset** ··· Near ventilation fan

  Tested with test dataset (see Table 2) ··· Near ventilation fan

  Skip2-LoRA is compared with SOTA [1] (see Table 3)

Table 1: Accuracy after full retraining

	Before	After
Damage1	60.61±13.73	98.99±2.81
Damage2	51.86±8.04	90.88±5.65
HAR	79.97±5.62	86.09±4.40

Table 3: Accuracy after finetuning (SOTA)

	TinyTL (GN)	TinyTL (BN)
Damage1	98.66±0.76	99.49±0.32
Damage2	92.09±3.17	96.01±2.74
HAR	88.76±0.91	89.27±1.13

Table 2: Accuracy after finetuning (Baseline models & This work)

	FT-All	FT-Last	FT-Bias	FT-All-LoRA	LoRA-All	LoRA-Last	Skip-LoRA	Skip2-LoRA
Damage1	98.73±2.11	94.19±2.24	79.42±7.50	98.63±2.14	98.26±1.32	94.67±2.92	96.07±2.14	96.19±2.29
Damage2	88.12±6.13	92.43±3.67	79.56±6.47	88.88±5.73	86.45±4.90	93.55±3.50	93.24±3.86	93.46±3.21
HAR	90.99±1.86	89.31±1.06	82.21±1.27	90.40±2.49	91.09±1.26	89.79±1.46	92.10±1.05	91.99±1.00

→ Skip2-LoRA is better than FT-Last & LoRA-Last; it is comparable to LoRA-All

[1] Han Cai et al., "TinyTL: Reduce Memory, Not Parameters for Efficient On-Device Learning", NeurIPS'20.

Evaluations: Execution time of FT

- **Execution time @ Raspberry Pi Zero 2W**
  - **Skip-LoRA reduces the backward computation**
  - **Skip2-LoRA reduces both forward & backward computation**

Table 1: Execution time (train & predict) of Fan (Damage1 & Damage2) dataset [msec]

	FT-All	FT-Last	FT-Bias	FT-All-LoRA	LoRA-All	LoRA-Last	Skip-LoRA	Skip2-LoRA
Train@batch	5.864	2.633	3.721	6.053	4.113	2.642	2.952	**0.450**
forward	2.812	2.601	2.832	2.868	2.942	2.613	2.807 →	0.309
backward	2.866	0.030	0.885	2.993	1.157	0.026 →	0.136	0.131
weight update	0.186	0.002	0.003	0.192	0.014	0.002	0.010	0.010
Predict@sample	0.142	0.144	0.148	0.150	0.155	0.143	0.151	0.154

Table 2: Execution time (train & predict) of HAR dataset [msec]

	FT-All	FT-Last	FT-Bias	FT-All-LoRA	LoRA-All	LoRA-Last	Skip-LoRA	Skip2-LoRA
Train@batch	11.323	6.179	6.795	11.577	7.459	6.031	6.328	**0.595**
forward	6.569	6.129	6.050	6.660	6.390	6.005	6.130 →	0.396
backward	4.373	0.047	0.742	4.480	1.052	0.024 →	0.184	0.185
weight update	0.381	0.003	0.003	0.437	0.017	0.002	0.014	0.014
Predict@sample	0.308	0.307	0.304	0.317	0.314	0.309	0.314	0.317

Evaluations: Training curves & time

- So far, numbers of epochs were set to enough values
- Here we estimate actual finetuning times of three datasets on Raspberry Pi Zero 2W

  Based on training curves of Skip2-LoRA with 10 trials

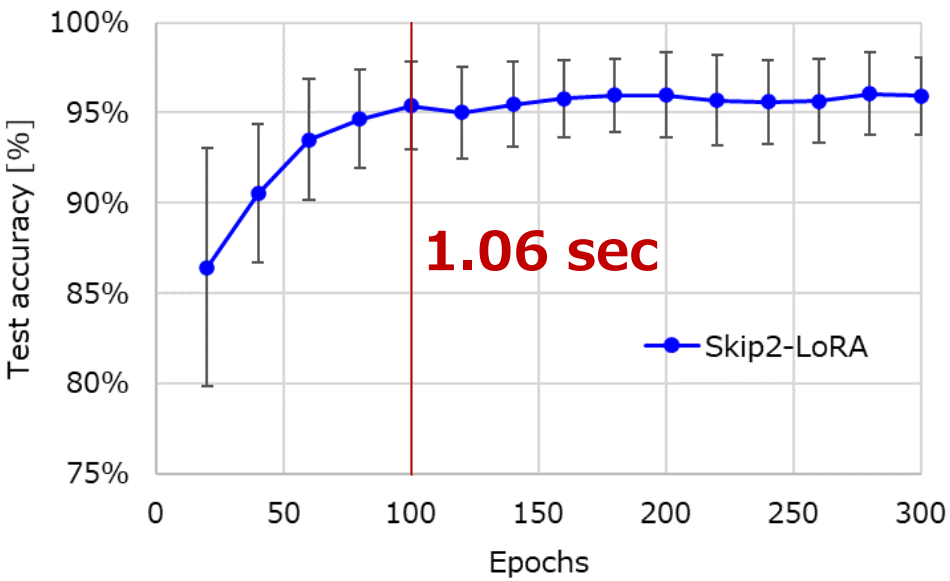


Figure 1: Damage 1 dataset

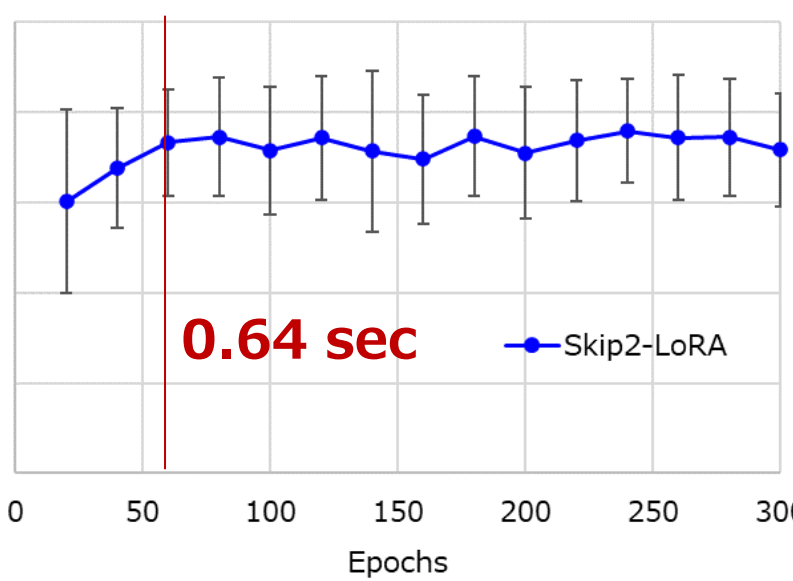


Figure 2: Damage 2 dataset

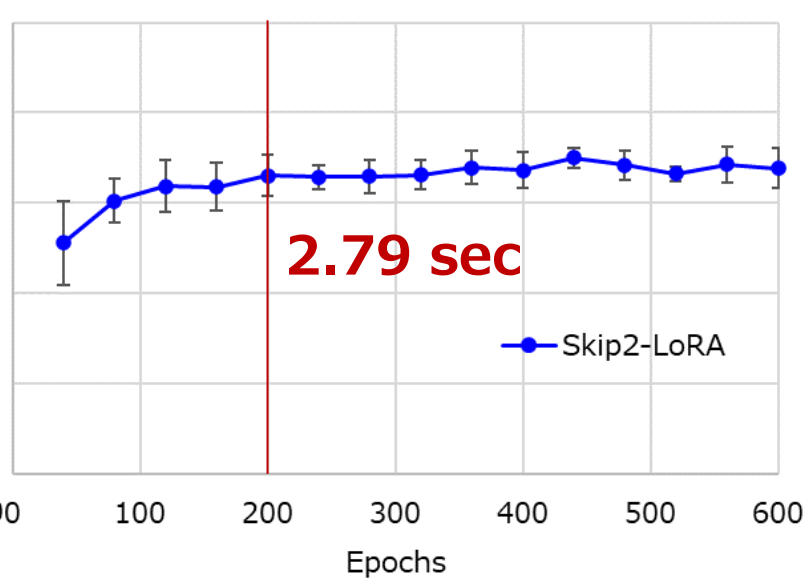
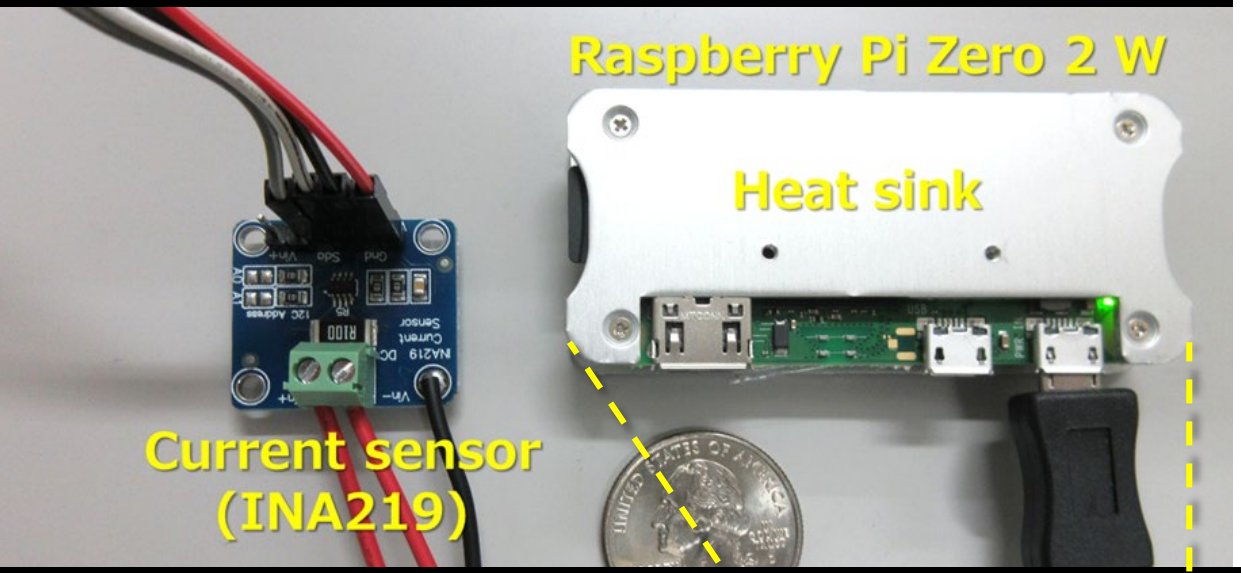


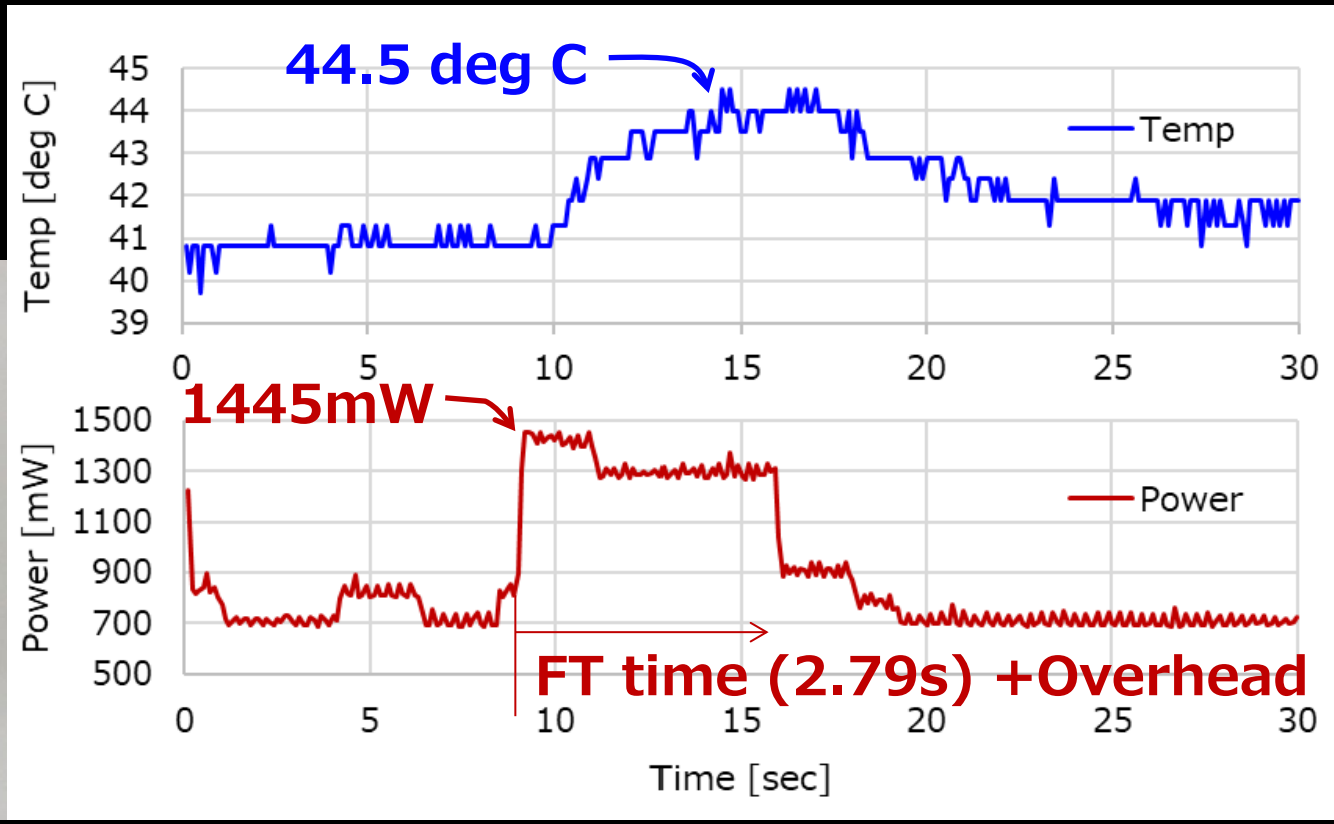
Figure 3: HAR dataset

→ Finetuning takes only a few seconds with Skip2-LoRA

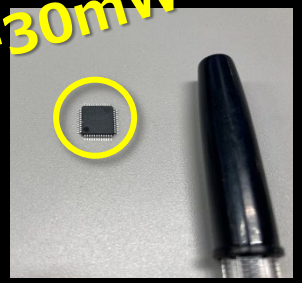
Evaluations: Power consumption of FT

- **Raspberry Pi Zero 2W [1]**
  ARM Cortex-A53 @1GHz

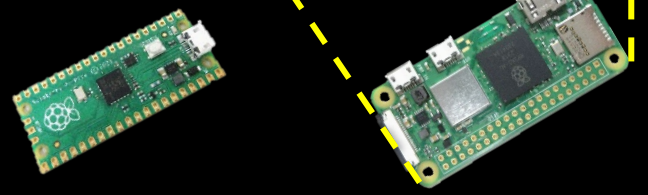




20-30mW

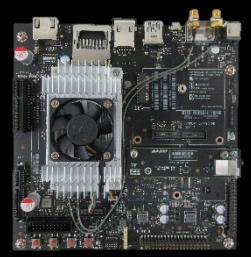


0.2-1W



2-10W

15W+

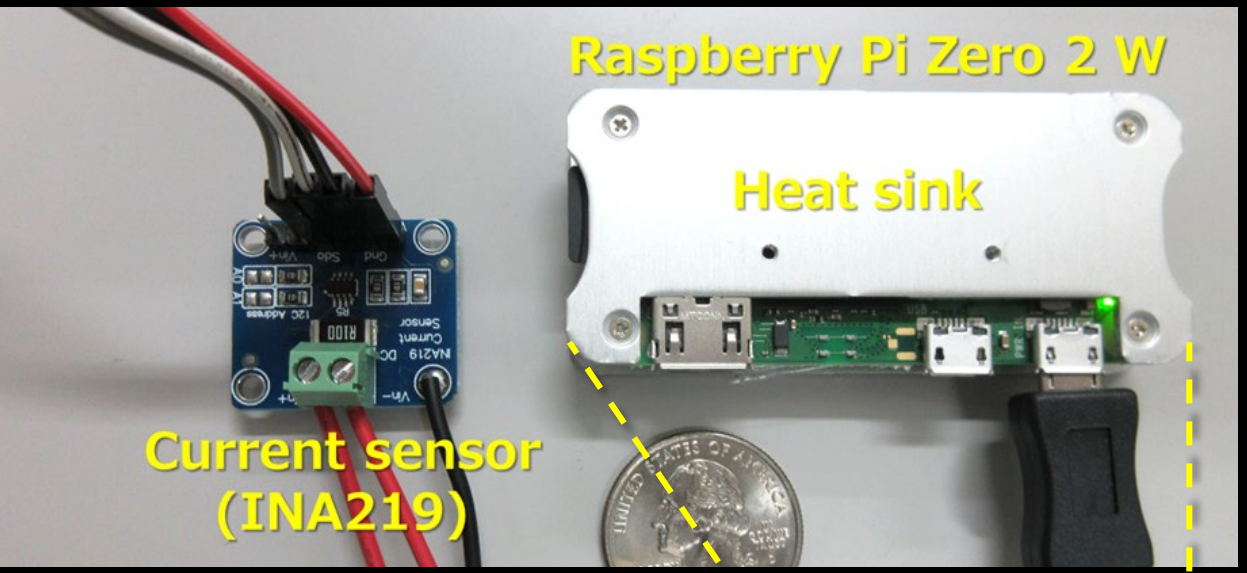


200W+

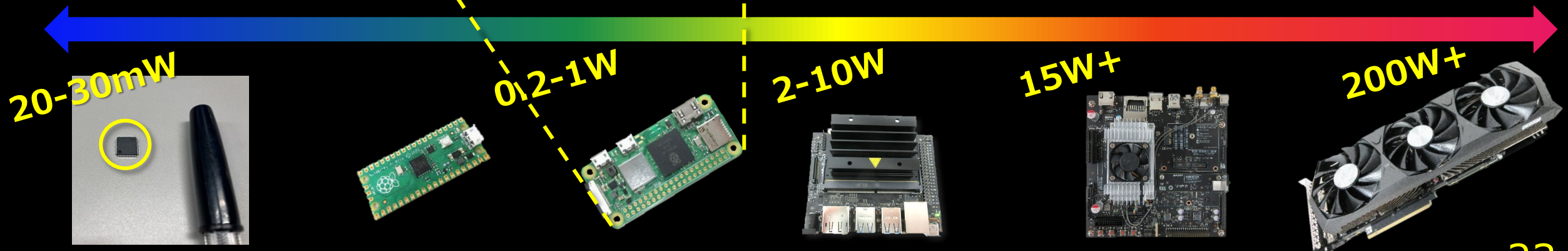
[1] "Raspberry Pi Zero 2 W", https://www.raspberrypi.com/products/raspberry-pi-zero-2-w.

Summary: Skip2-LoRA for on-device FT

- **Reduce both forward & backward computations**



- **Compared to LoRA-All,**
  - 90% reduction in FT time
  - Comparable accuracy
- **Run on $15 computer**
  - FT within a few seconds
  - At most 1.445W (44.5 degC)
- **See you at WiP poster!** [1]



[1] Hiroki Matsutani et al., "A Lightweight On-device CNN Fine-tuning using Skip2-LoRA and Quantized Cache", ASP-DAC'25 WiP poster.