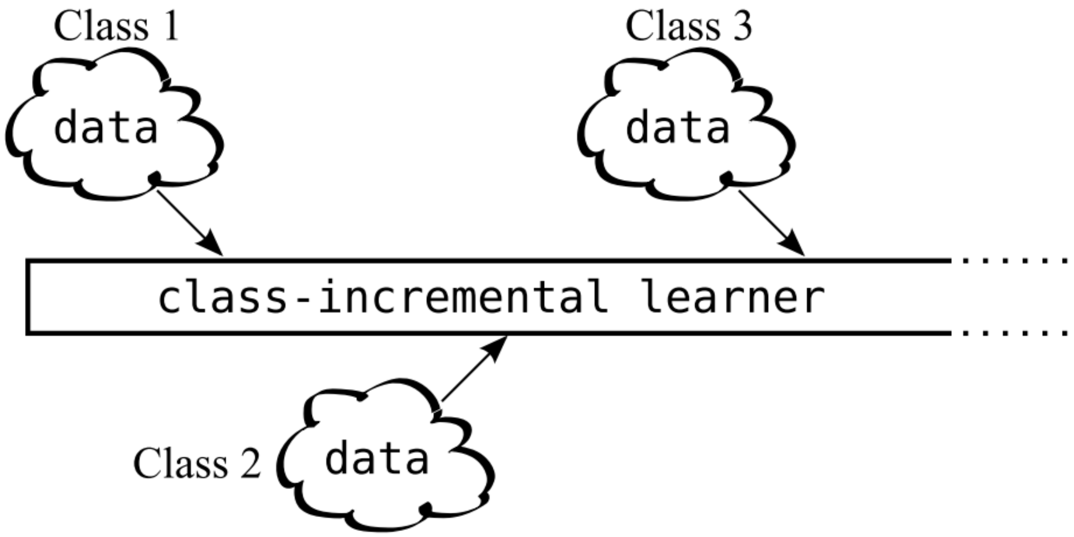
Cost-effective On-device Continual Learning over Memory Hierarchy with Miro

Mobicom’23

Xinyue Ma, Suyeon Jeong, Minjia Zhang, Di Wang, Jonghyun Choi, Myeongjae Jeon

Class-Incremental Continuous learning

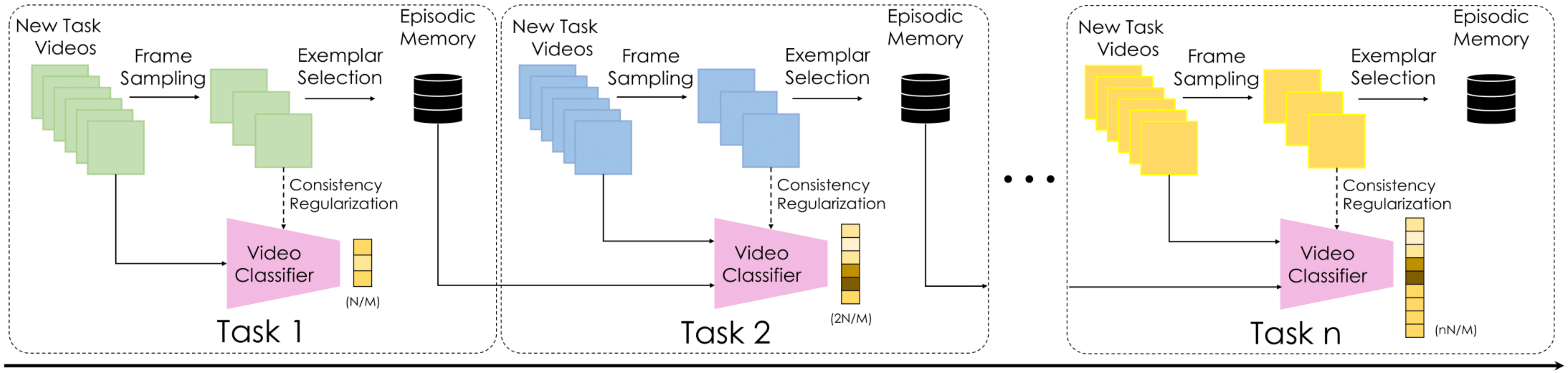
- Continuous learning performs model training **incrementally** as new data becomes available
- stability-plasticity dilemma



Class-Incremental Continuous learning

- Replay Function: Episodic memory (EM)

  Store old samples in storage and replay them during incremental training.



Class-Incremental Continuous learning

- EM over Memory Hierarchy (HEM)

    A **small** set of old samples in memory.

    A **large** set of old samples in storage.

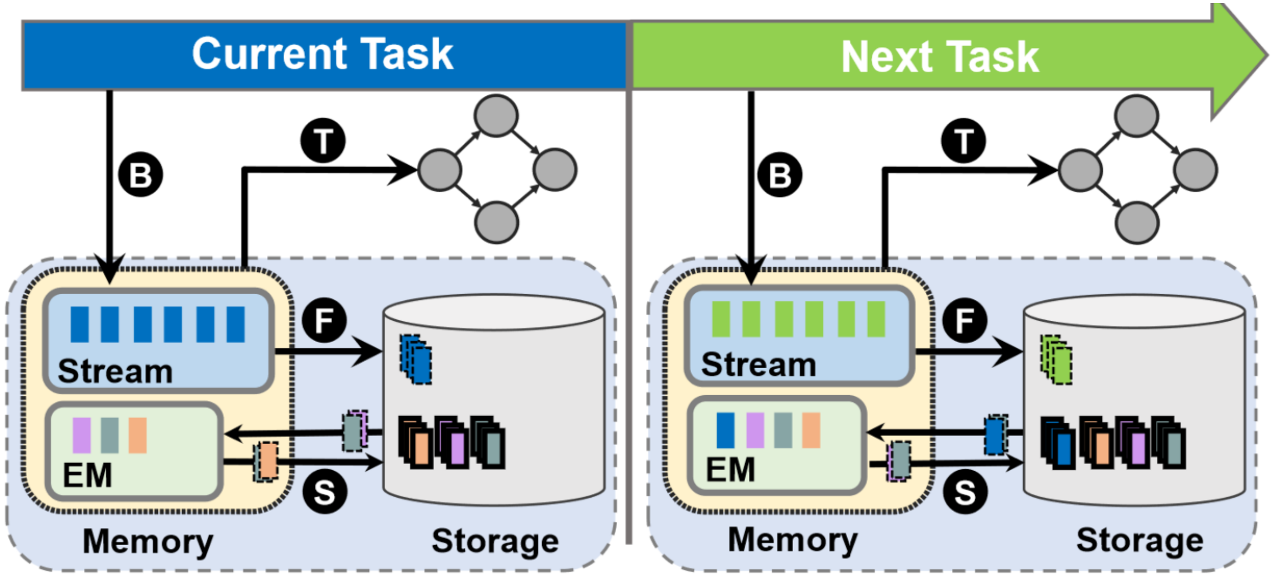


Figure 1: Architecture and execution stages of HEM.

On-device Continuous learning

- HEM workflow
  - B(Buffering): A new task *N* are accumulated in a **stream buffer**.
  - F(Flushing): Update EM with the samples in the SB.
  - T(Training): Combine new samples in data stream and old samples in EM to train.
  - S(Swapping): Swap data between in-memory samples and in-storage samples of the same class.

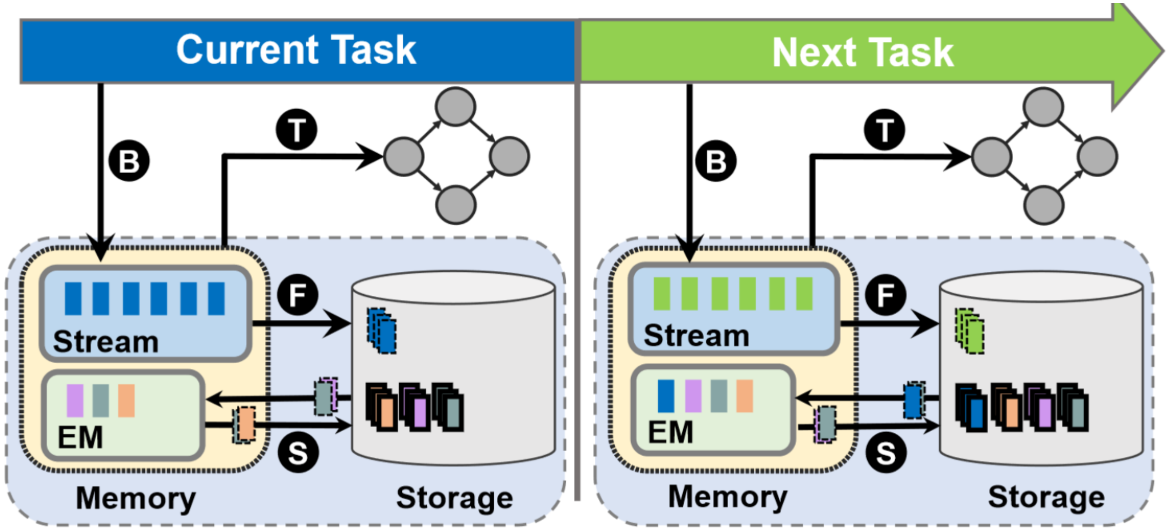


Figure 1: Architecture and execution stages of HEM.

On-device Continuous learning

- HEM workflow
  - B(Buffering): A new task *N* are accumulated in a **stream buffer**.
  - F(Flushing): Update EM with the samples in the SB.
  - T(Training): Combine new samples in data stream and old samples in EM to train.
  - S(Swapping): Swap data between in-memory samples and in-storage samples of the same class.

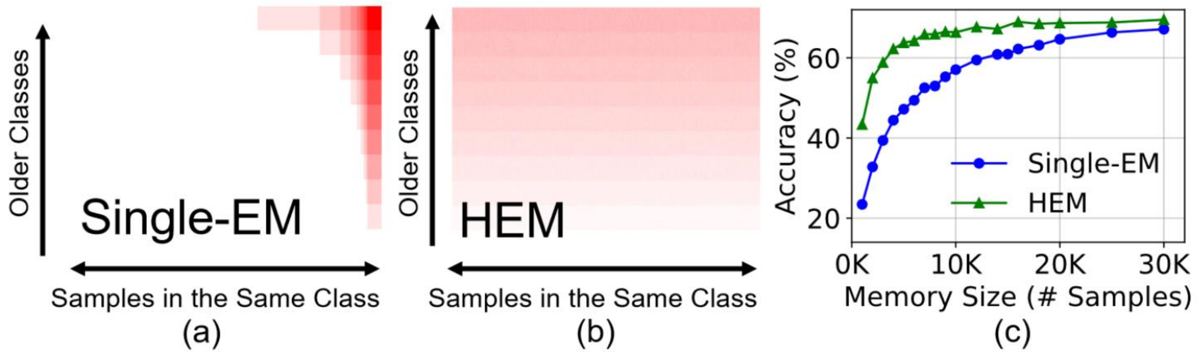


Figure 2: (a) and (b): Data diversity of old samples between single-level EM and HEM. (c): Accuracies over memory sizes.

On-device Continuous learning: Energy efficiency

- Design a **energy-efficient** HEM system

Parameter	Resource	Decision	Constraint
EM size	Memory	Dynamic	Trade-off
SB size	Memory	Dynamic	Trade-off
Swap ratio	I/O	Dynamic	Capacity
Epoch count	GPU	Static	Static

On-device Continuous learning: Energy efficiency

- Power: Static Consumption & Dynamic Consumption

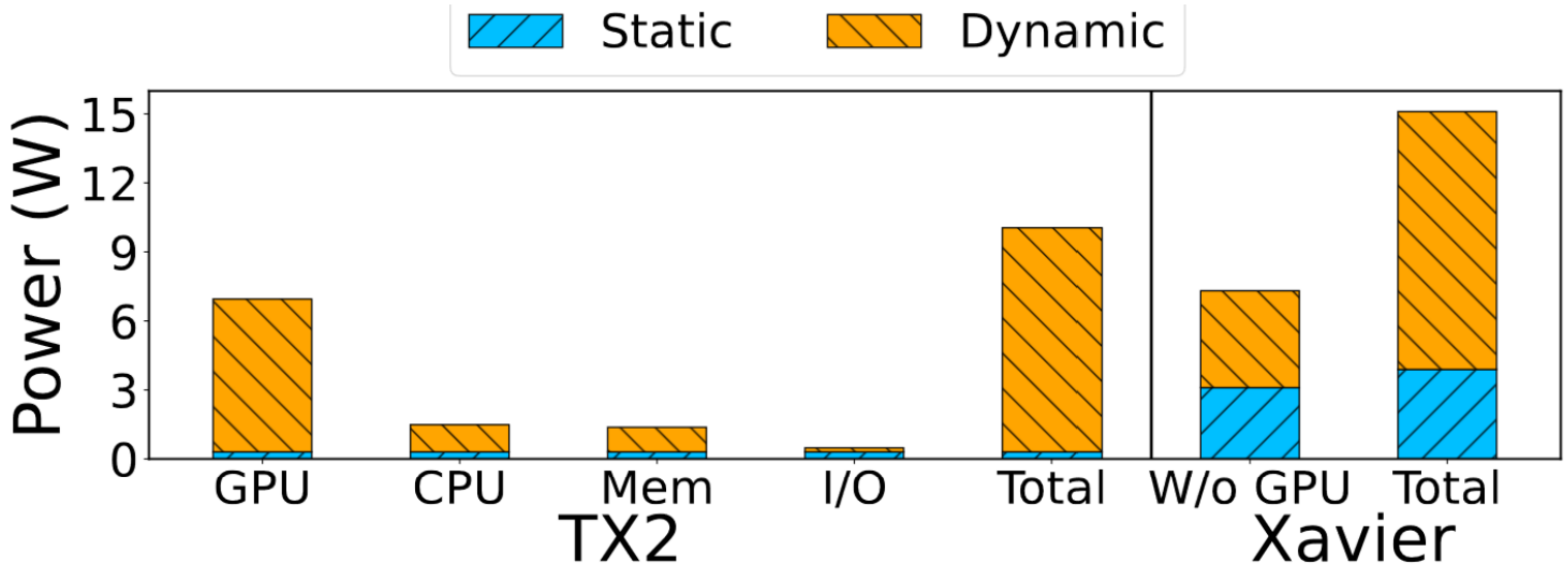


Figure 3: Power consumption of HEM across major system components on NVIDIA Jetson TX2 and Jetson Xavier NX.

On-device Continuous learning: Energy efficiency

- Memory parameters design:

Energy-accuracy trade-off

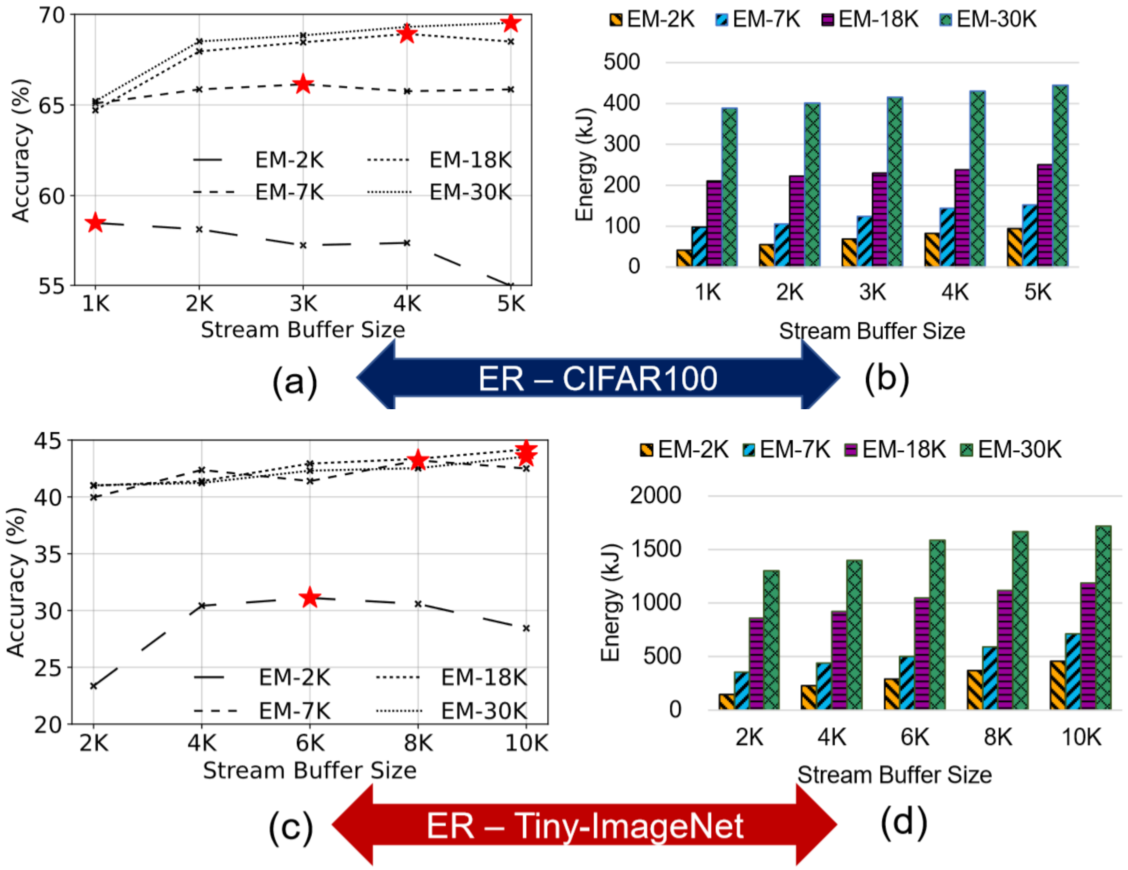
  - EM size & SB size

Large size

High Accuracy

High energy consumption

Long training time



On-device Continuous learning: Energy efficiency

- Memory parameters design: Energy-accuracy trade-off
  - Allocate EM size and SB size

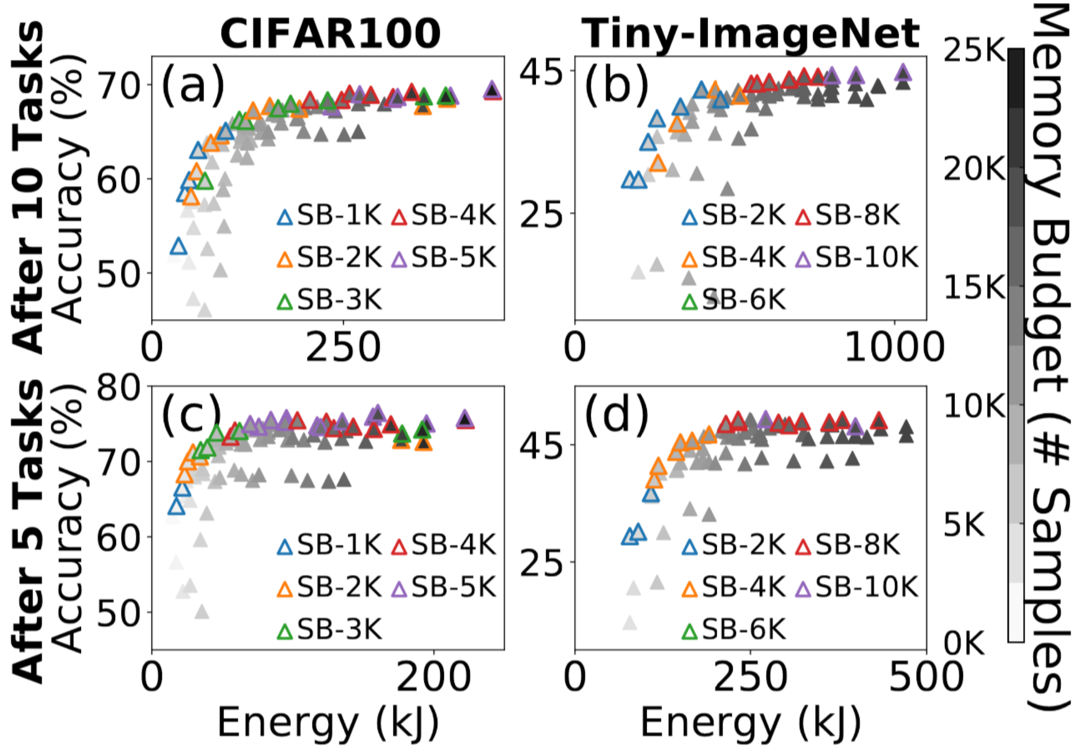


Figure 5: Energy-accuracy trade-offs over varying memory budgets shared by EM and SB.

On-device Continuous learning: Energy efficiency

- I/O parameters design:
  - Swap ratio: The ratio of EM samples swapped in every epoch.

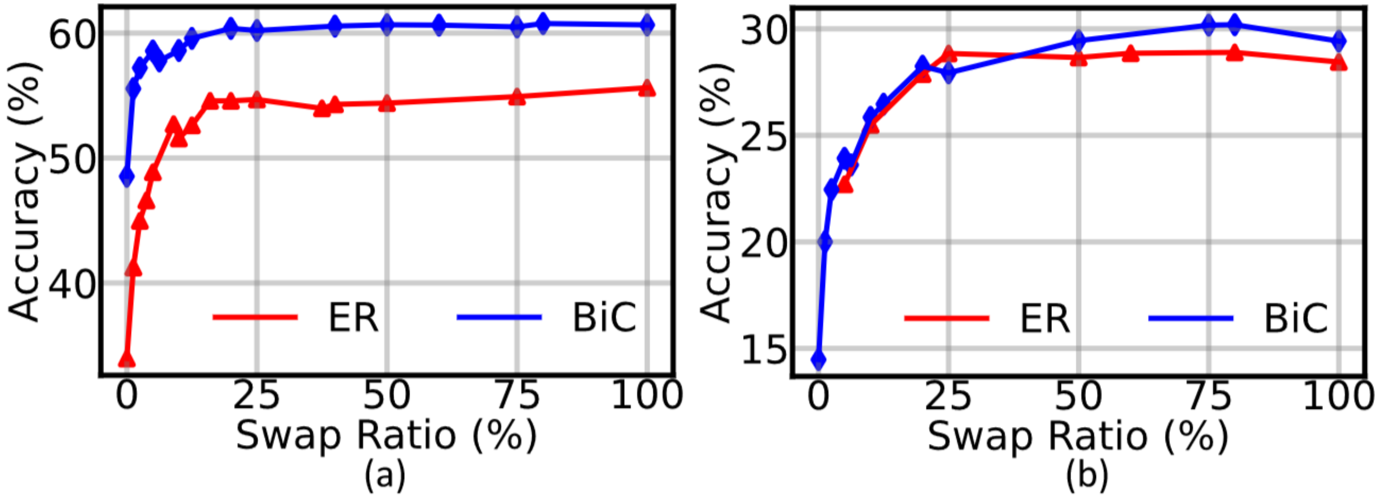


Figure 6: Accuracy over varying swap ratios for CIFAR100 (a) and Tiny-ImageNet (b).

On-device Continuous learning: Energy efficiency

- GPU parameters design:
  - Epoch count (static)

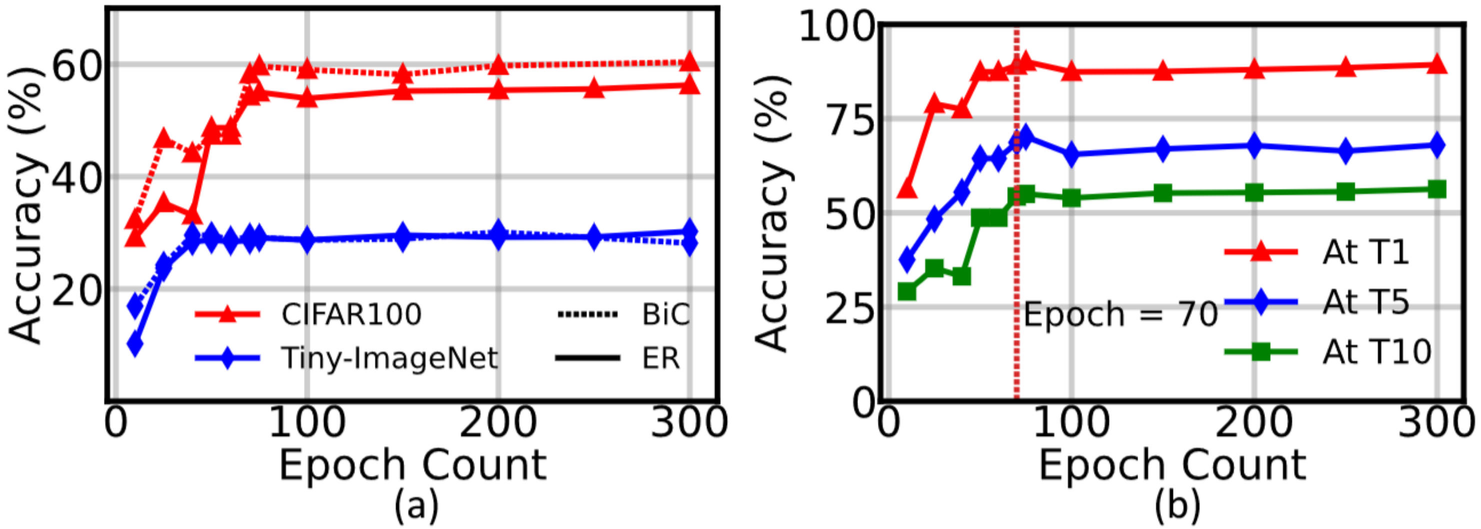


Figure 7: Accuracy over varying epoch counts.

System Overview: Miro

- Selecting data swapping strategy.
- Deciding **SB** and **EM** sizes.

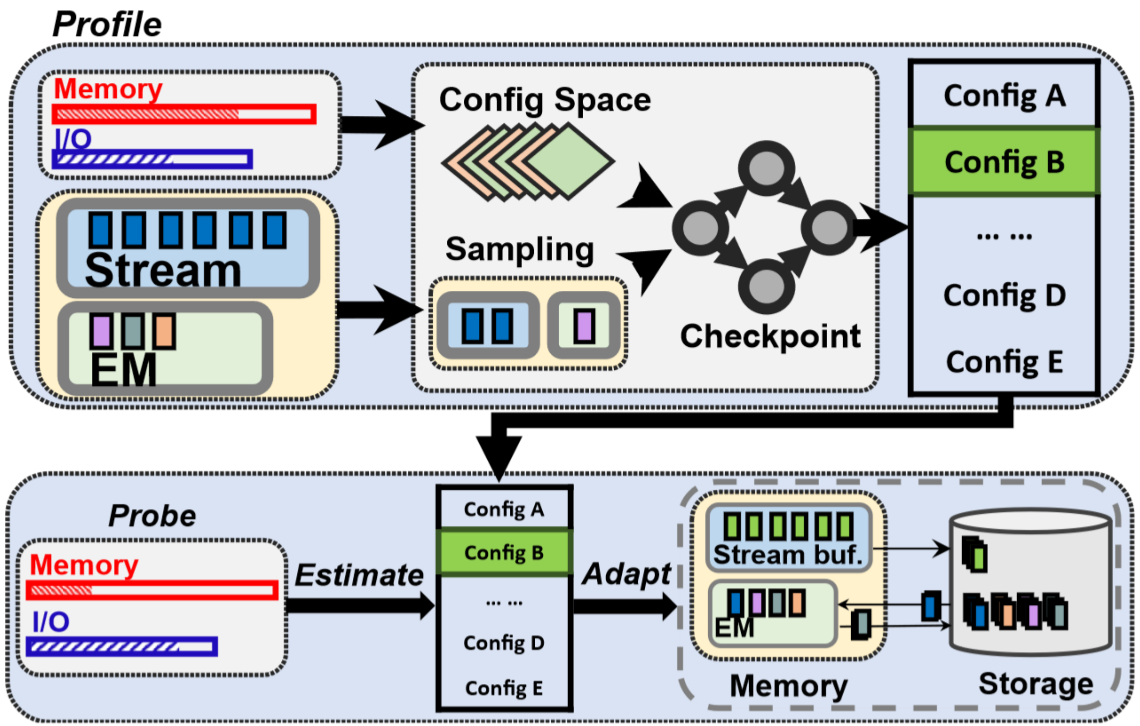


Figure 8: Miro system runtime architecture.

System Overview: Miro

- Selecting data swapping strategy.
  - A method like TCP congestion control.
    - **Idle**: Increase the swap ratio in a steady mode.(10%→20%→30%)
    - **Congest**: Decrease the swap ratio in a rapid mode.(100%→50%→25%)
    - **Stable**: Keep swap ratio.

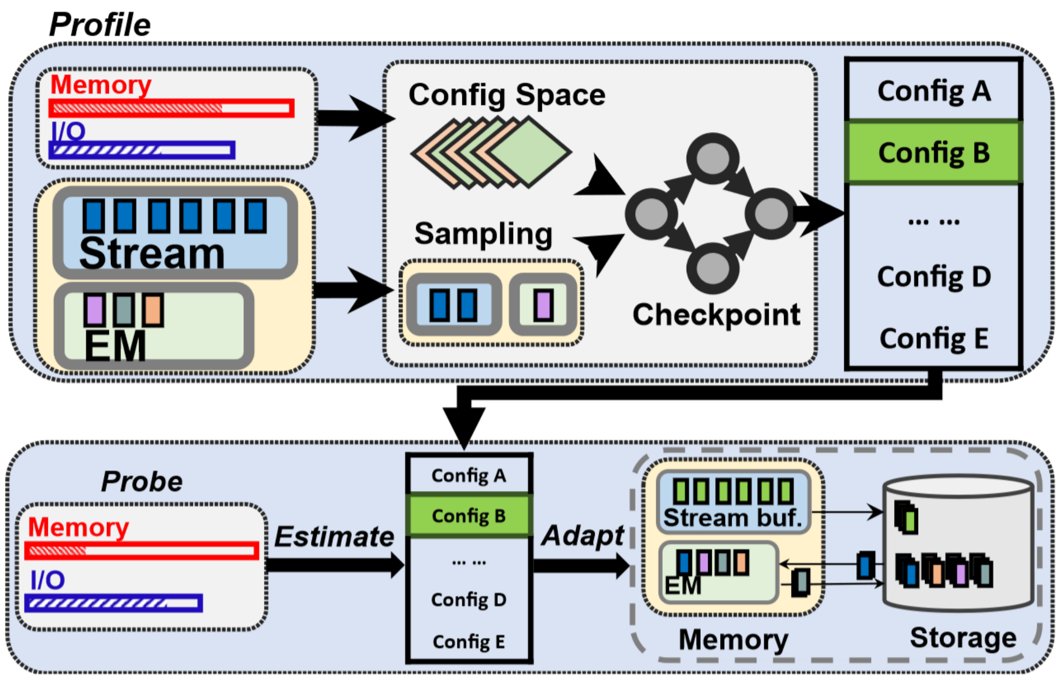


Figure 8: Miro system runtime architecture.

System Overview: Miro

- Deciding **SB** and **EM** sizes with **cutline.**
  - Keep a look-up table.

$$\text{Utility} = \frac{\text{Accuracy Gain}}{\text{Energy Usage}}$$

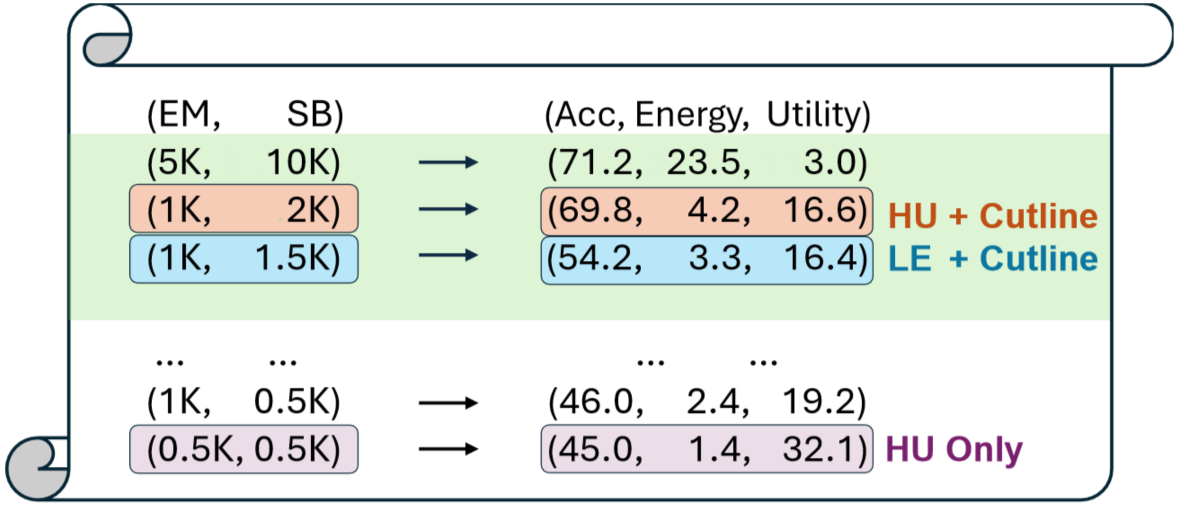


Figure 9: An illustrative example of how our method works. HU: highest utility. LE: lowest energy.

System Overview: Miro

- Profiling at Low Overhead
  - Use a **subset** of the train sample.
  - Perform training for **a small number of epochs**.

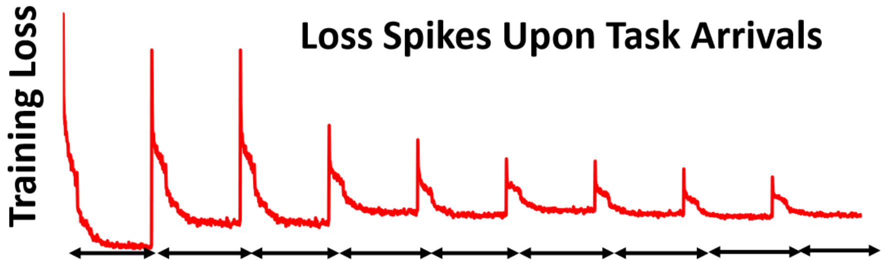


Figure 10: Constant spikes and noises in training loss when training new tasks.

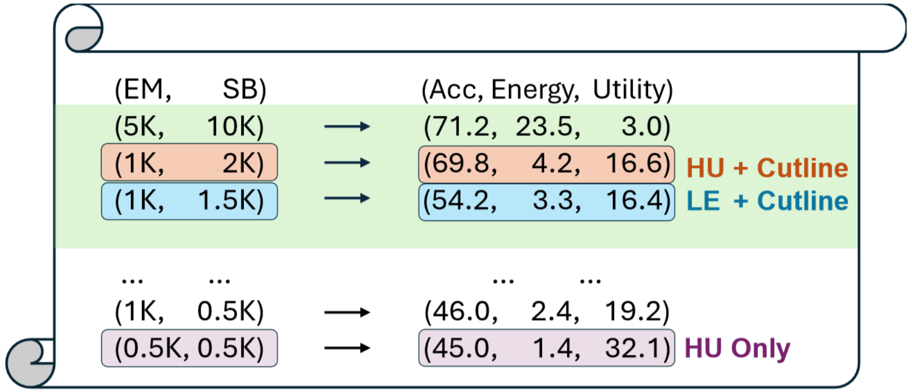


Figure 9: An illustrative example of how our method works. HU: highest utility. LE: lowest energy.

Workflow of Miro

- Profile
  - Select a set of random confs.
  - Evaluate the confs and choose a best one (**highest utility**).
- Probe
  - Monitor whether any parameters need reconfiguration.
- Estimate
  - Select the conf with the highest utility from previously profiled results.
- Adapt
  - Refine the target parameters.

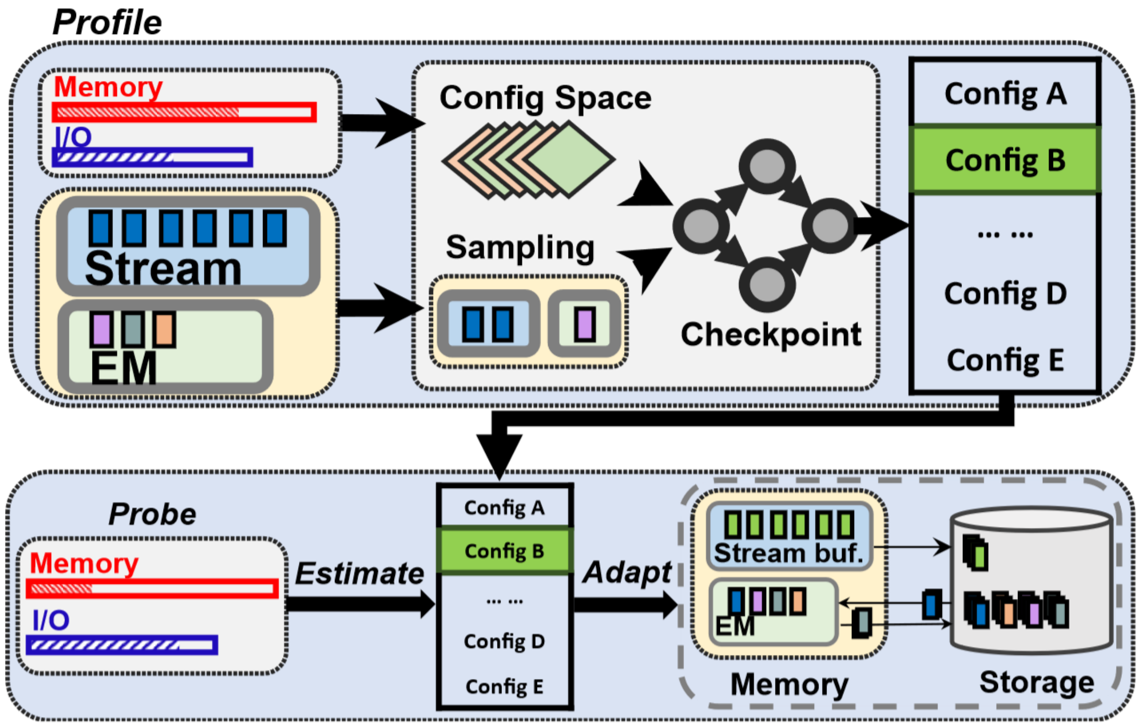


Figure 8: Miro system runtime architecture.

Experiment results

- Baseline
  - CarM[1]: Use static confs.
  - BestStatic: Explores all possible static confs and finds the conf that offers the best costeffectiveness.
  - BestHistory: Select the best intermediate conf identified by BestStatic after completing half of the tasks and uses the conf for future tasks.
  - Heuristic: Treat new and old tasks equally and assigns memory to SB vs. EM proportional to the number of tasks placed in each component.

[1]S. Lee, M. Weerakoon, J. Choi, M. Zhang, D. Wang, and M. Jeon. CarM: Hierarchical Episodic Memory for Continual Learning. In DAC, 2022.

Experiment results

- DataSet
  - Audio Classification: UrbanSound8k dataset.
  - Human Activity Recognition: Daily and Sports Activities dataset.
  - Large-scale Image Classification: CIFAR100, ImageNet1k.

Experiment results

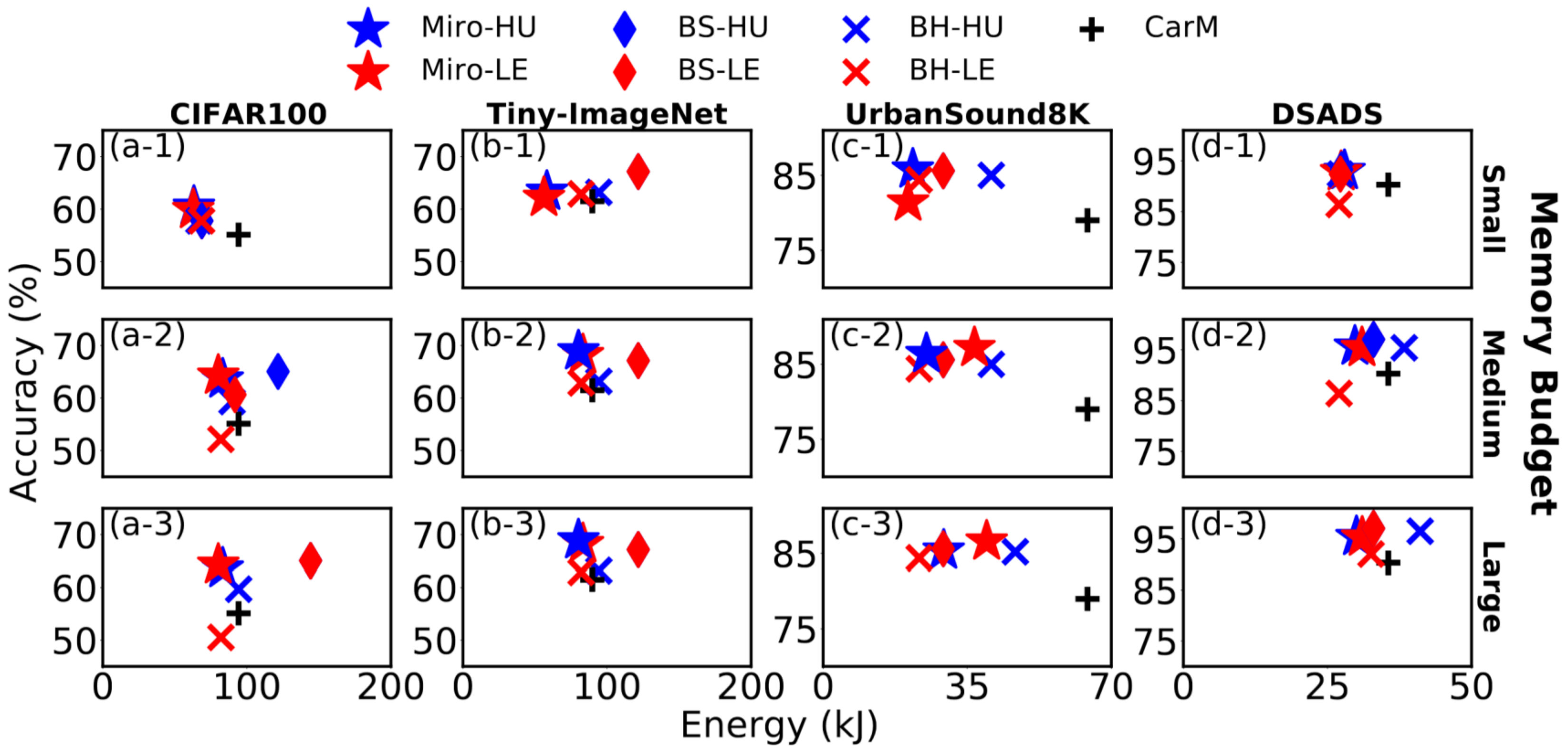


Figure 11: Energy-accuracy trade-offs over competing static methods for a combination of different datasets (a-d) and memory budgets (1-3). For example, the subgraph (a-1) compares the methods using CIFAR100 on a small memory budget. The memory budgets are 10K, 25K, and 50K for CIFAR100 and Tiny-ImageNet and 1K, 2K, and 2.5K for UrbanSound8K and DSADS.

Experiment results

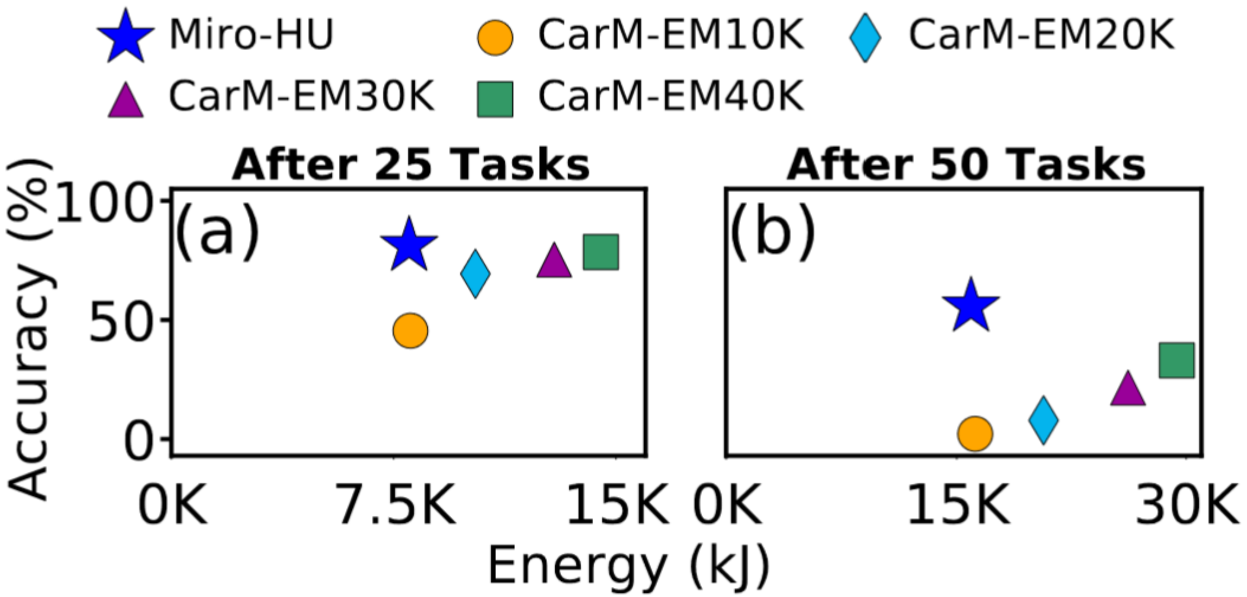


Figure 13: Energy-accuracy trade-offs after completing 25 tasks (a) and 50 tasks (b) for ImageNet1k-50Tasks.

Conclusion

- Propose a energy-efficient class incremental system.
- The HEM design consumes too much **storage**, though it is cheap, a storage management mechanism is need.
- The default configurations is set by hand and related to device, thus the system is not general to all mobile devices.