DaCapo: Accelerating Continuous Learning in Autonomous Systems for Video Analytics

Yoonsung Kim
Changhun Oh
Jinwoo Hwang
Wonung Kim
Seongryong Oh
Yubin Lee
Hardik Sharma†*
Amir Yazdanbakhsh‡
Jongse Park

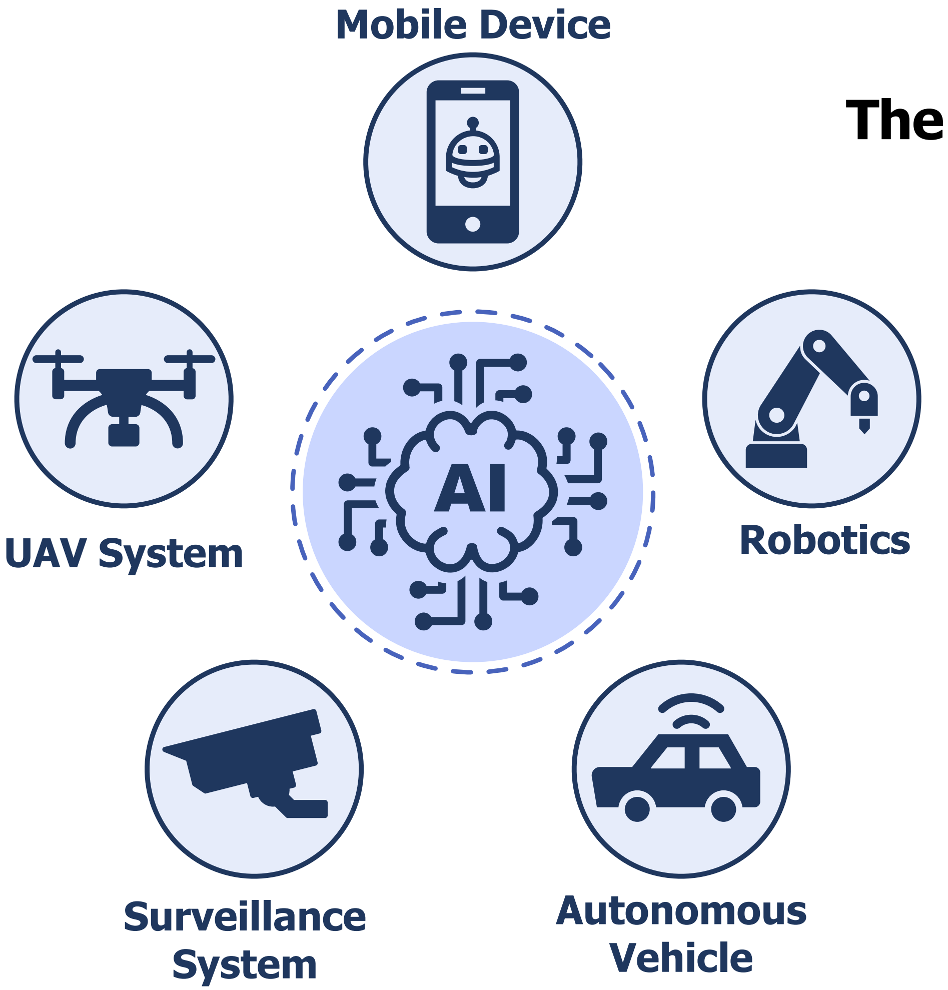
*This work was done at Google

KAIST

†*Meta

‡Google DeepMind

On-Device AI:
Local Intelligence for Autonomous Systems



The autonomous system market is expected to grow to $28.5 billion by 2028 with an annual growth rate of 43.0%*

On-Device AI Chips in industry

SAMSUNG

Qualcomm

*MarketsandMarkets™ Autonomous AI and Autonomous Agent Market

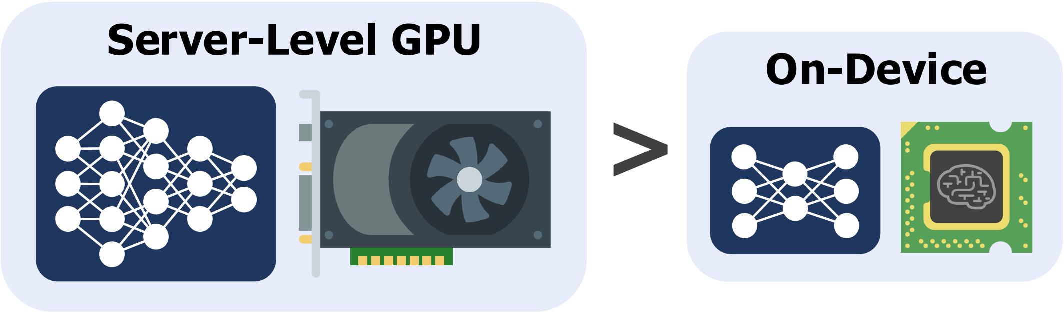
Deployment of On-Device AI

Considerations of Inherent Features: Data Drift and Model Capacity

- **Data drift: Changes of input data distribution**
- **On-device DNNs are lightweight due to constrained resources**

Input data distribution changes over time

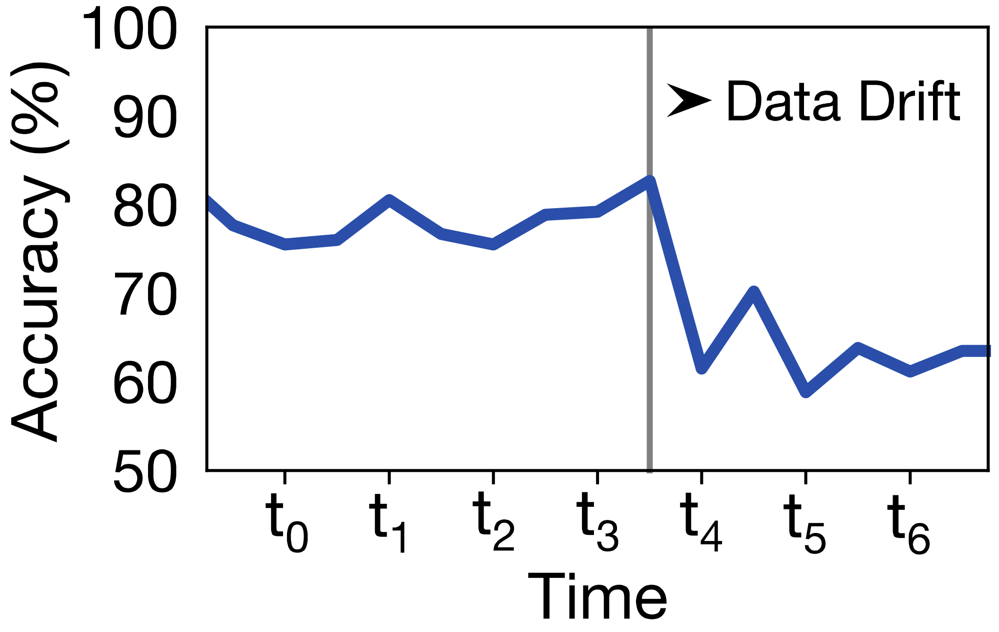
On-device DNNs have low model capacity



On-device DNNs are **sensitive to data drift** due to **low model capacity**

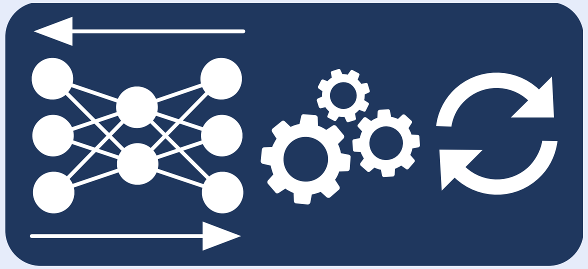
On-Device DNNs Suffer from Data Drift

Accuracy degradation from data drift



Solution

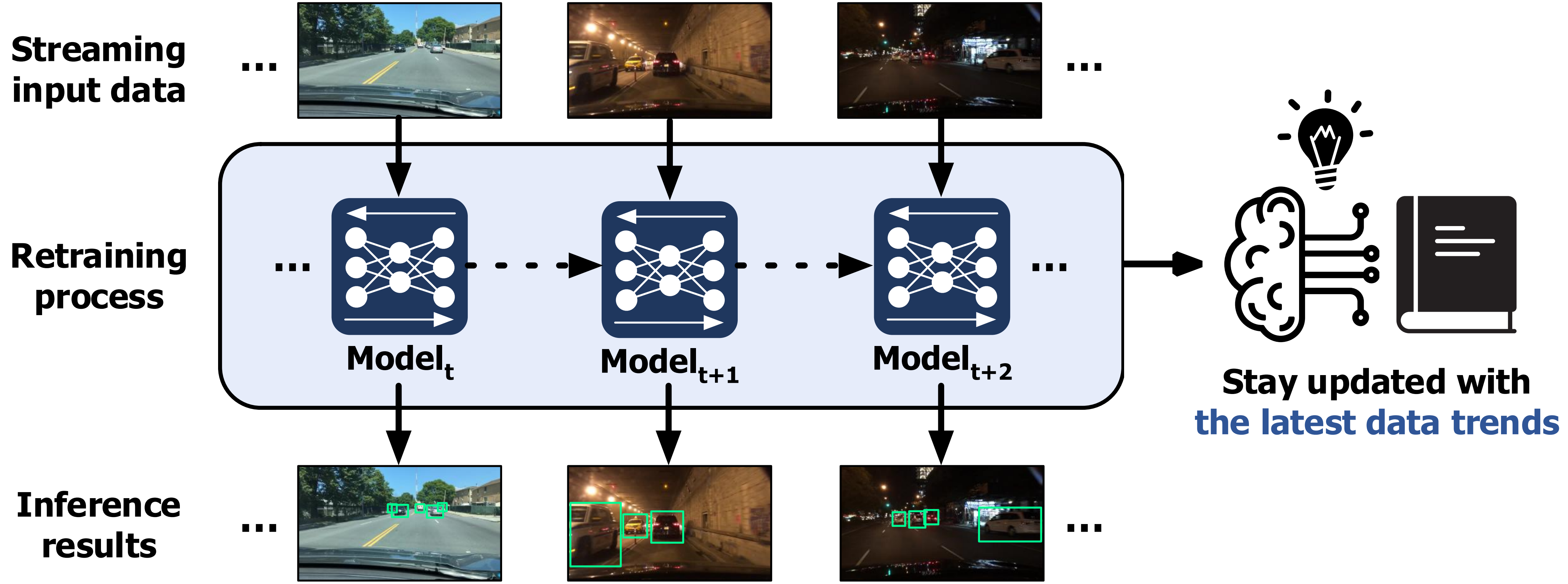
Runtime DNN adaptation



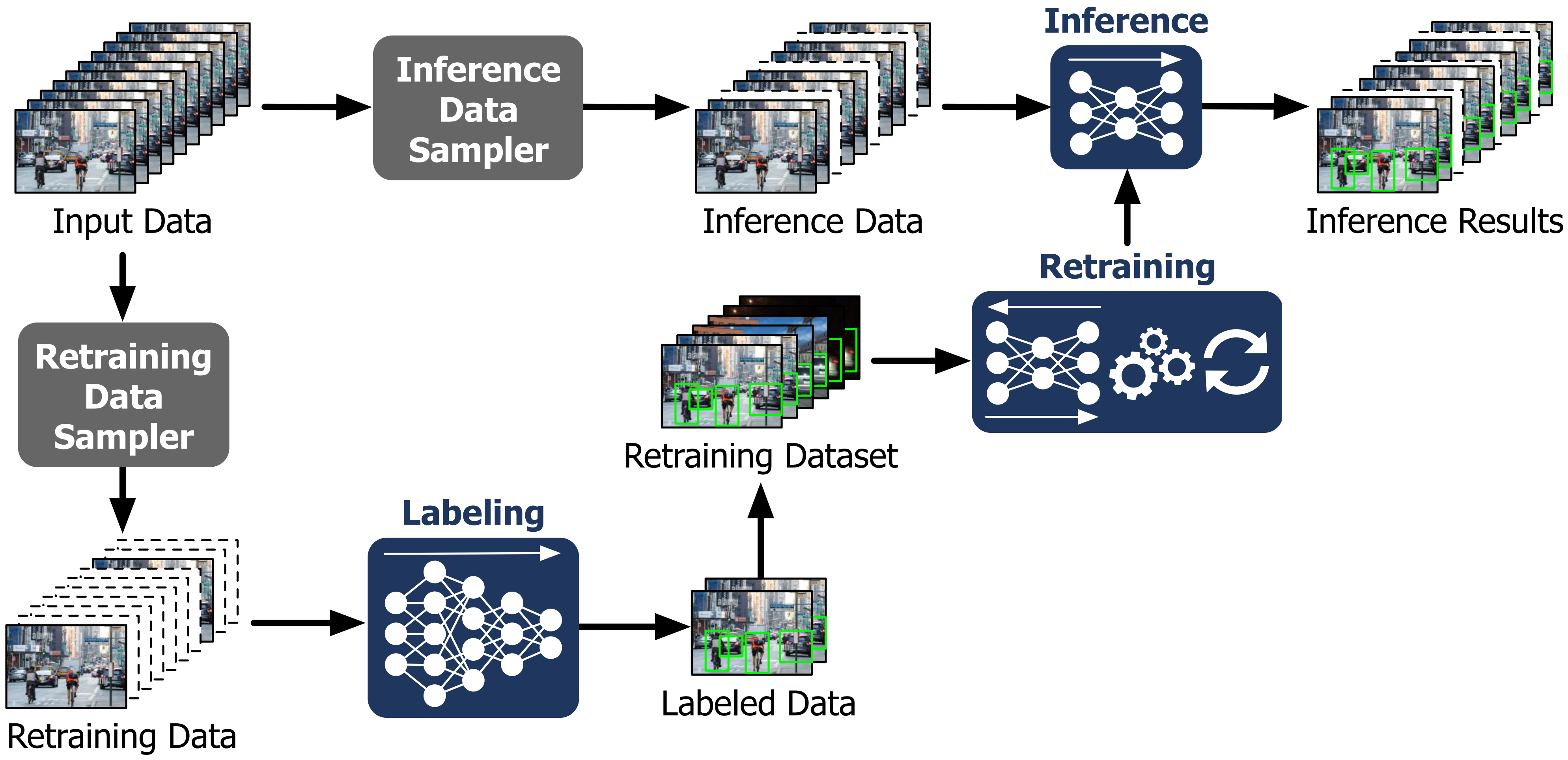
Retrain DNN over time

Existing Solution: Continuous Learning

- **Continuous learning (CL): Keep retraining DNN model over time**
  - Recent advances in systems and architecture [NSDI'22,23, MM'23, ASPLOS'24, HPCA'24]



Workflow of Continuous Learning



Workflow of Continuous Learning

Problems of CL with GPU server:

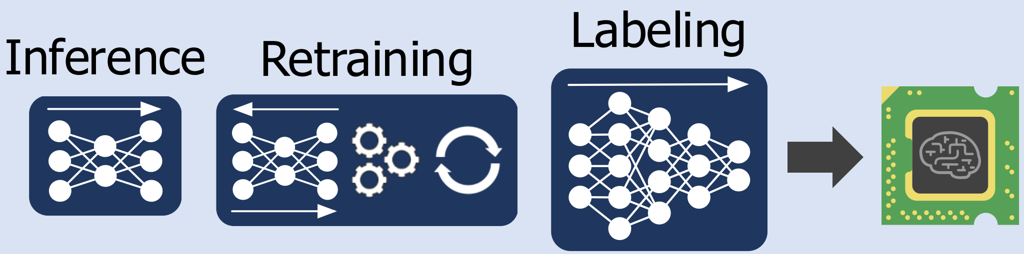
(1) Privacy, (2) network availability, and (3) latency concerns

We aim to build on-device continuous learning system

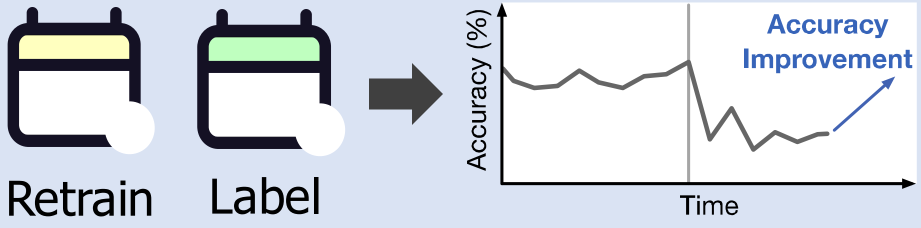
[1] Bhardwaj et al., "Ekya: Continuous Learning of Video Analytics Models on Edge Compute Servers," NSDI 2022
[2] Khani et al., "RECL: Responsive Resource-Efficient Continuous Learning for Video Analytics," NSDI 2023
[3] Kong et al., "Edge-Assisted On-Device Model Update for Video Analytics in Adverse Environments," MM 2023

Challenges of On-Device CL System

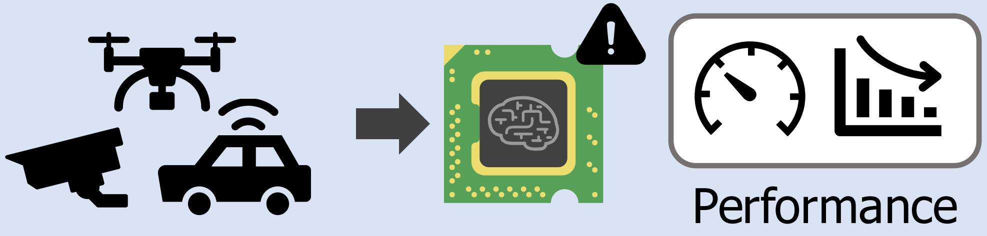
Challenge 1:
How to share resource across kernels?



Challenge 2:
How to schedule retraining & labeling?



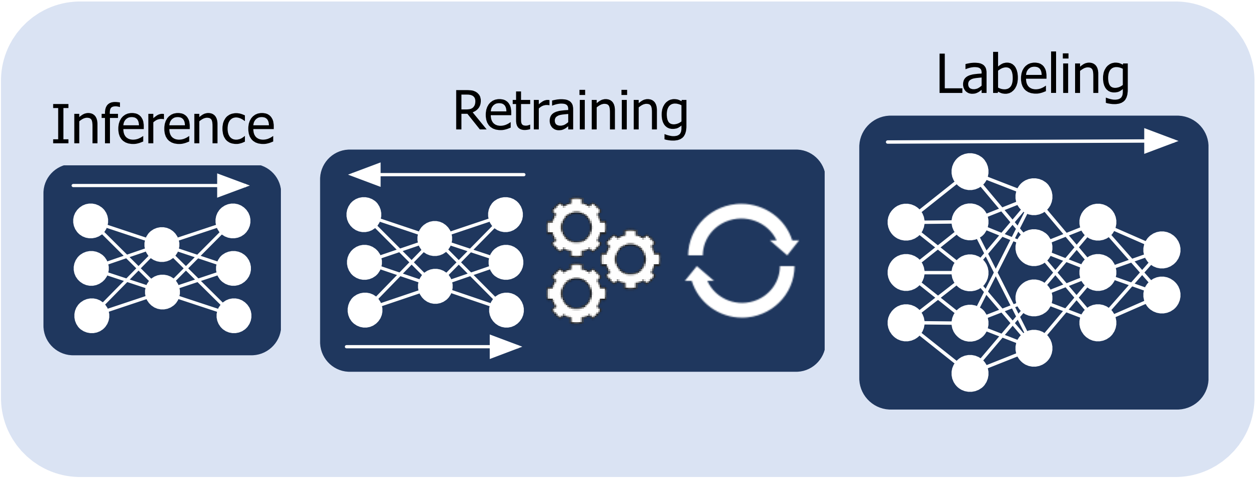
Challenge 3:
How to achieve efficient CL system?



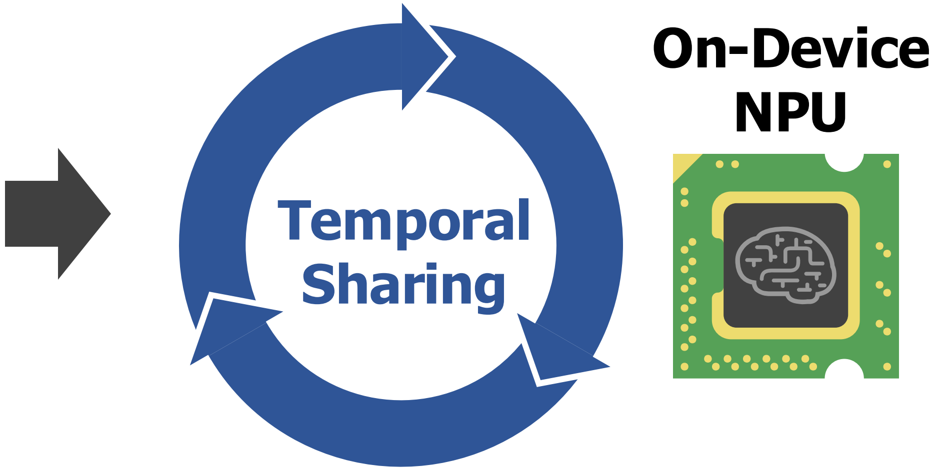
Challenge 1: Resource Sharing

Executing Three Key Kernels Using On-Device Resources

- **Naïve solution: Time-sharing across kernels**
- **Problem: Different computational demands between kernels**



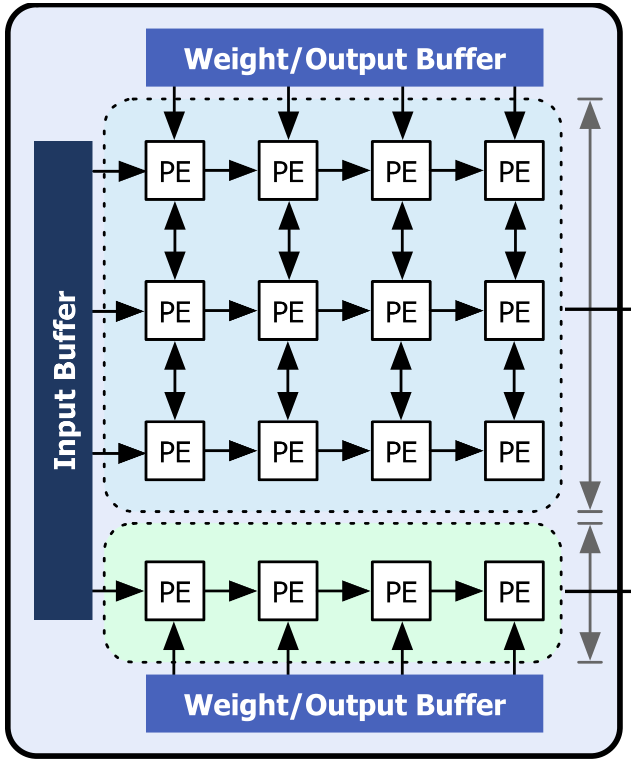
Three main CL kernels



Resource inefficiency occurs

Solution: Spatial Partitioning

Vertically Partitionable Systolic Array

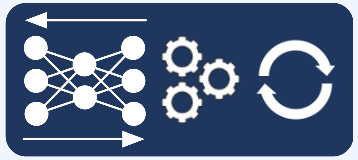


Top-Sub Accelerator (T-SA)

Bottom-Sub Accelerator (B-SA)

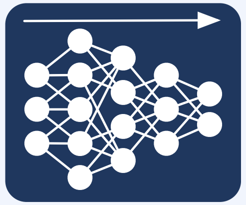
Allocate

Retraining



and

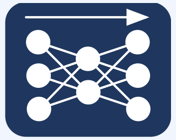
Labeling



✅ Leverage large HW resource for model adaptation kernels

Allocate

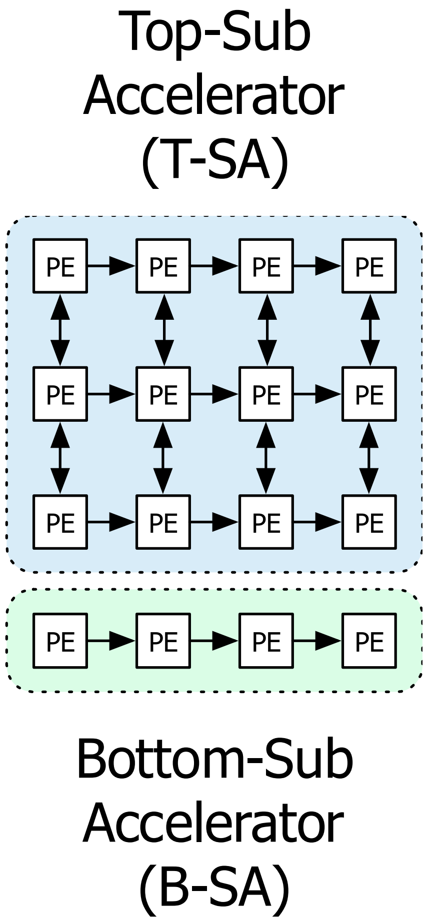
Inference



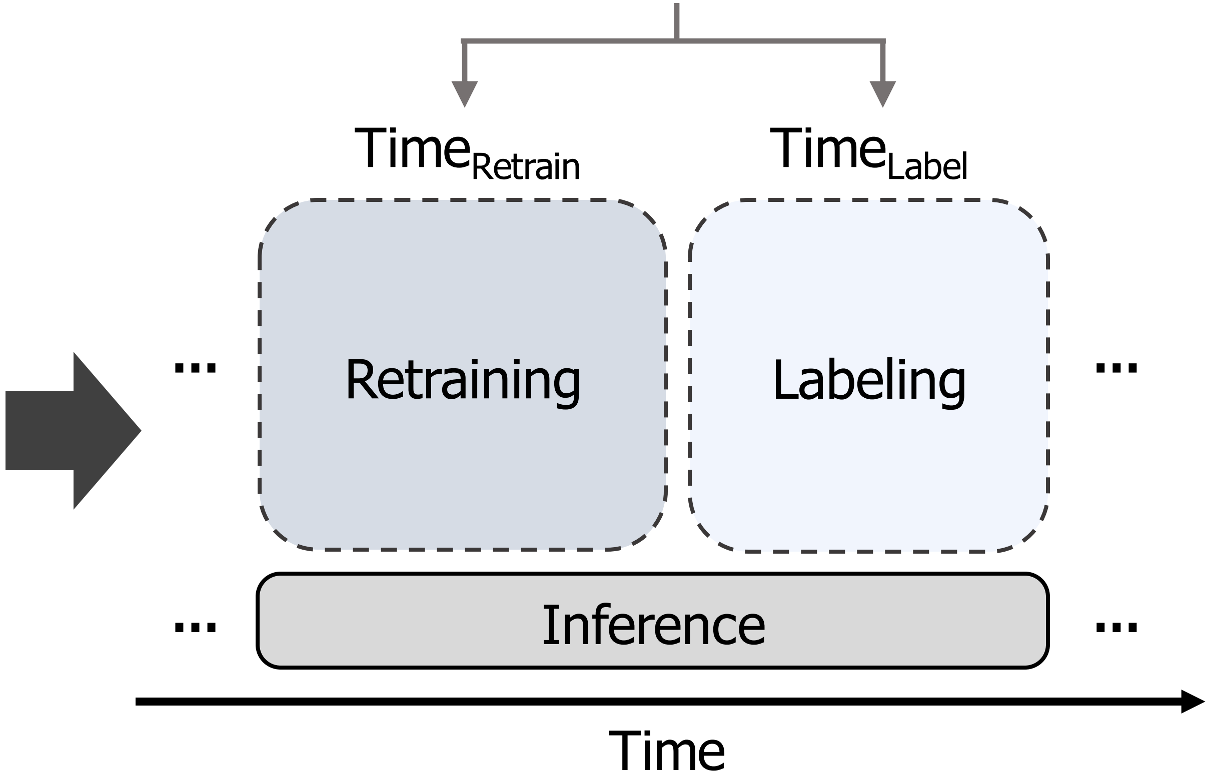
✅ Allocate minimal HW resource to satisfy FPS requirements

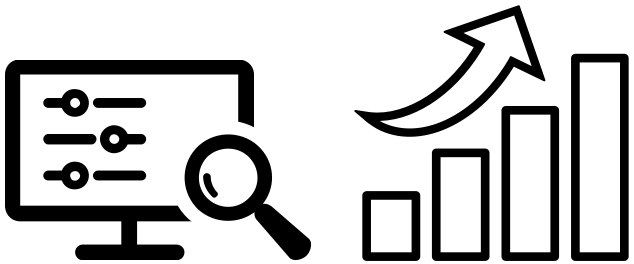
Challenge 2: Resource Scheduling

Scheduling Retraining and Labeling Kernels for Model Adaptation



Schedule time resources for kernels

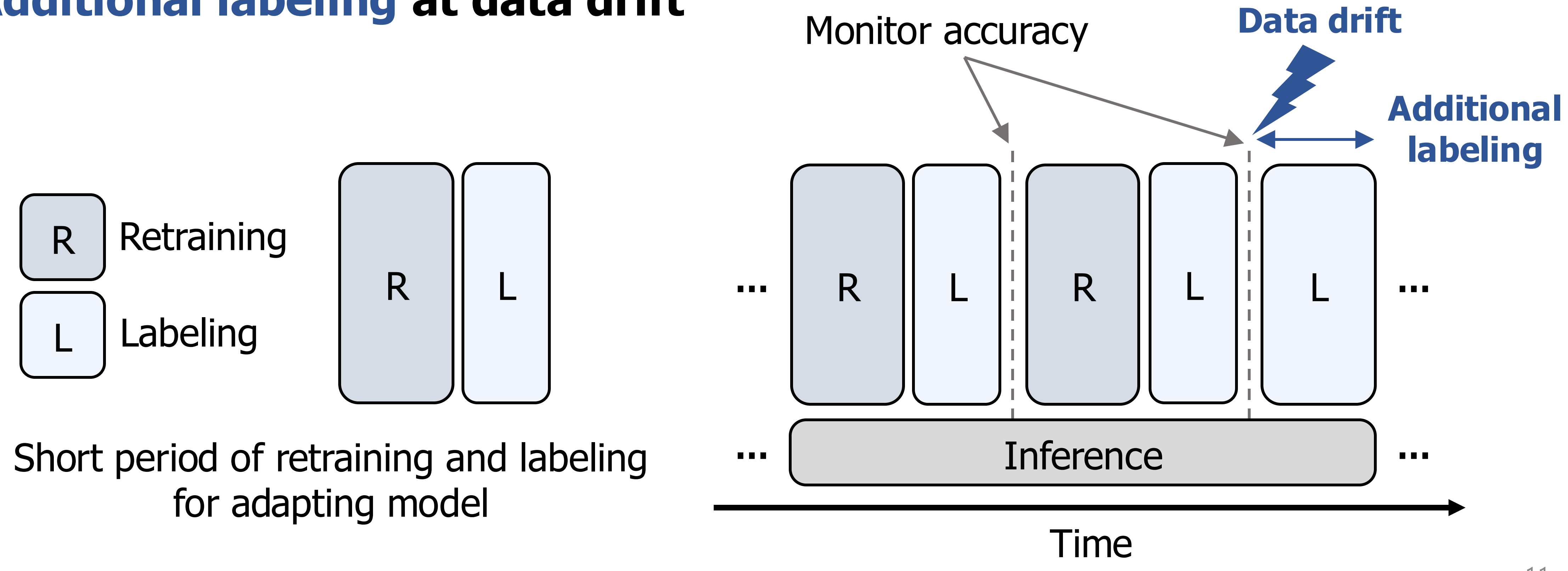




Resource scheduling affects **accuracy improvement**

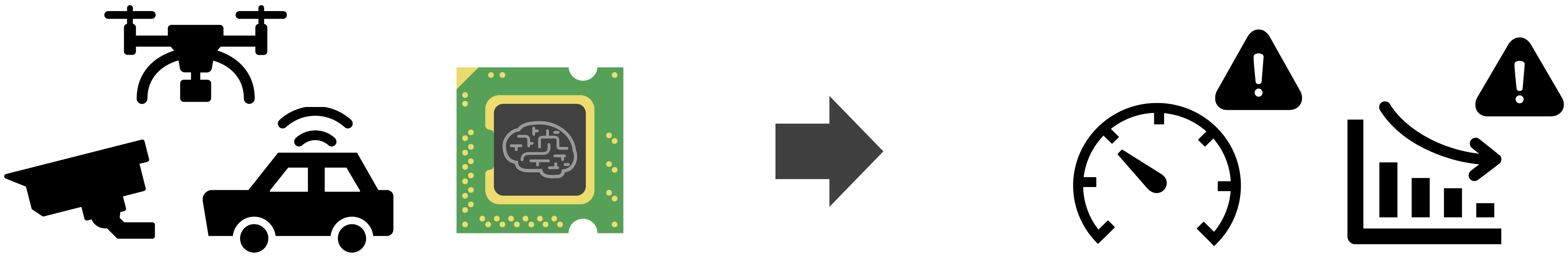
Solution: Fine-Grained Retraining and Labeling

- **Frequent model adaptation intervals**
- **Data drift detection by monitoring accuracy**
- **Additional labeling at data drift**



Challenge 3: Resource-Constrained System

- **On-device resources hinders optimal performance of CL system**
- **Inefficiency of CL system degrades adaptability to data drift**



CL system on **limited resources**

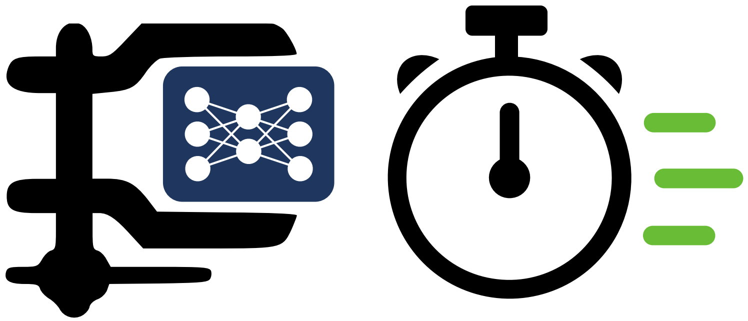
Low **performance** and **adaptability**

We need to design **performant** and **effective** on-device CL system

Solution: Flexible Low-Precision Arithmetic

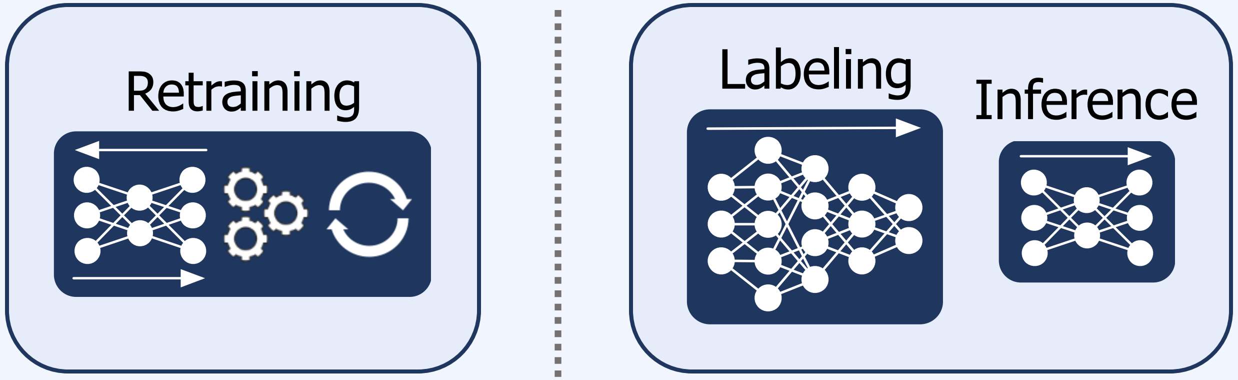
Dynamic Quantization Using Block Floating Point (BFP) Format

Quantization



Achieving **faster responses** from retraining and labeling kernels

Different precision levels suitable for each kernel



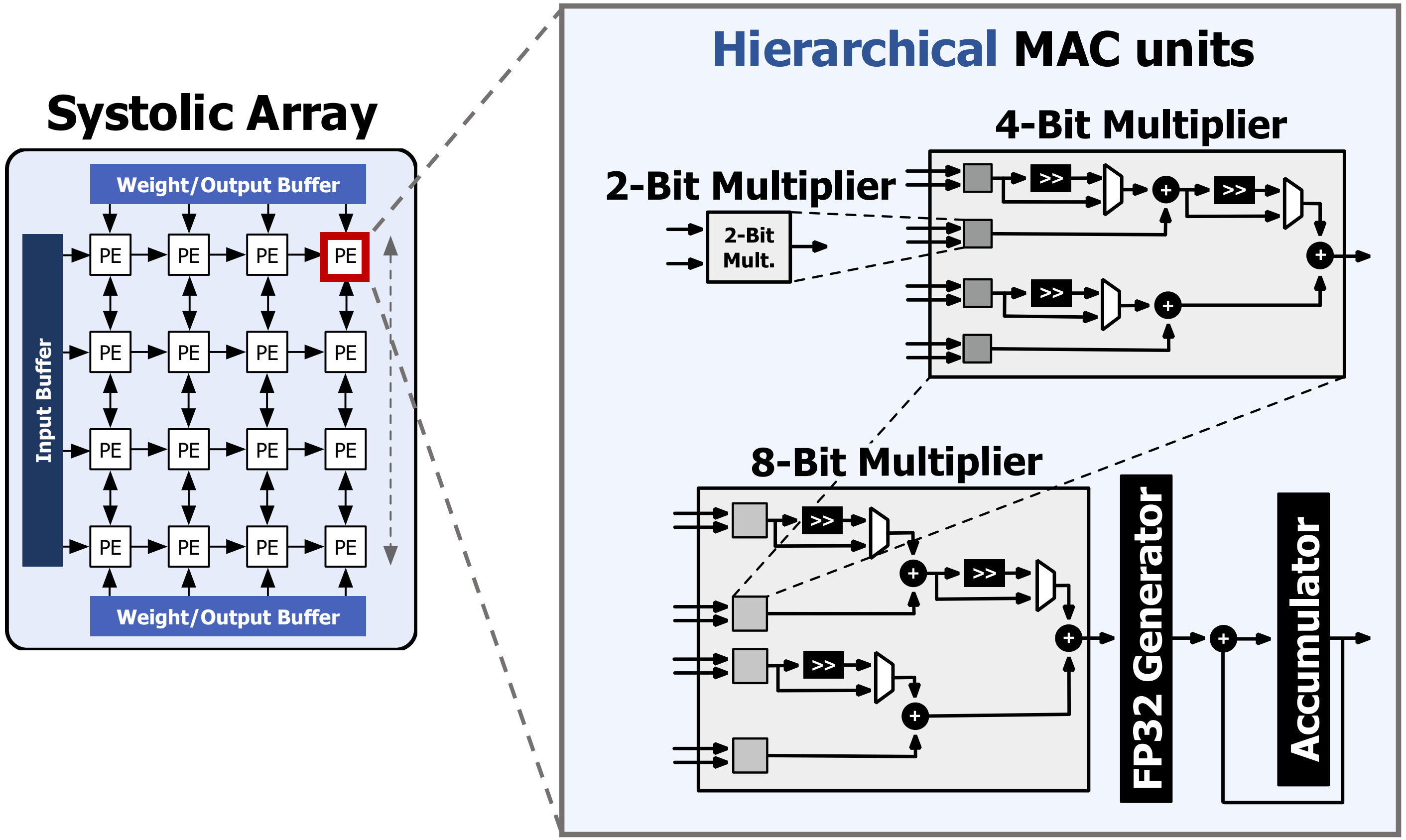
Higher precision (MX9)

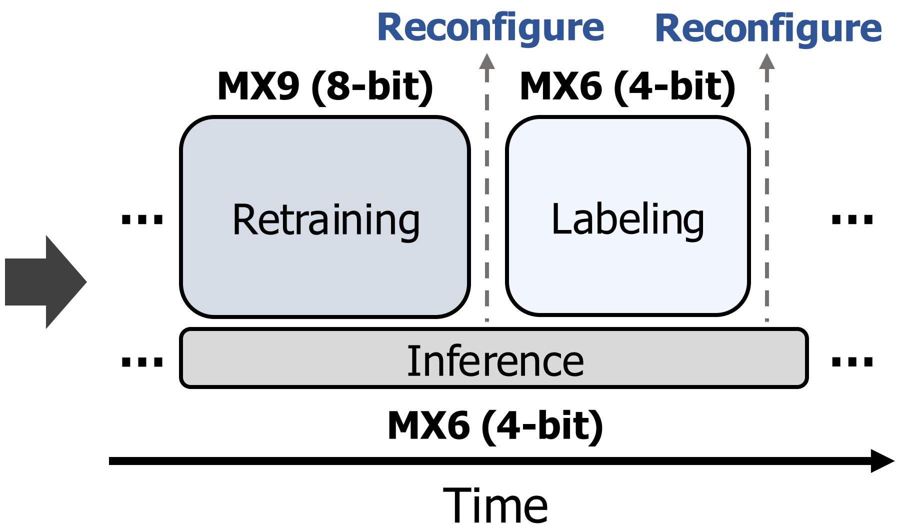
Lower precision (MX6)

*Use **Microsoft MX**, a variant of BFP formats

Microarchitecture for Dynamic Precisions

- **Reconfigurable PEs supporting two different precisions**





*More details in paper

Evaluation Methodology

Dataset

- BDD100K driving dataset

BDD100K

Baselines

- Ekya[1]
- EOMU[2]

*Both baselines target high-performance GPU systems

Scenario

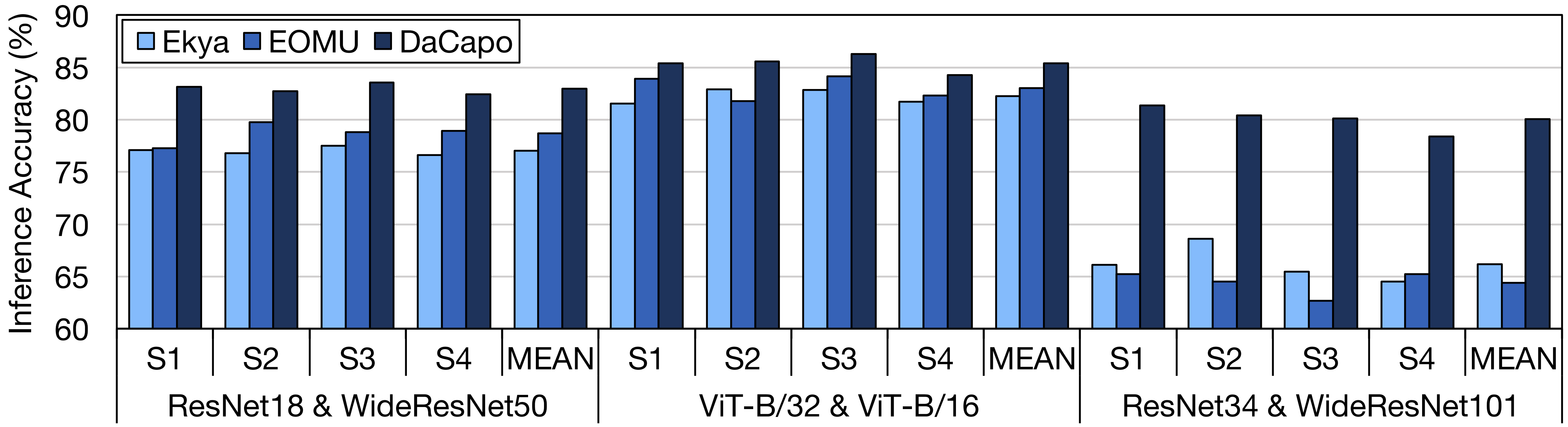
- A series of frames from BDD100K
- Real-world datasets with data drifts

Cycle-accurate simulator

- DaCapo system simulator: modified SCALE-Sim
- RTL synthesis and verification
    - Using Synopsys Design Compiler and CACTI

[1] Bhardwaj et al., "Ekya: Continuous Learning of Video Analytics Models on Edge Compute Servers," NSDI 2022
[2] Kong et al., "Edge-Assisted On-Device Model Update for Video Analytics in Adverse Environments," MM 2023

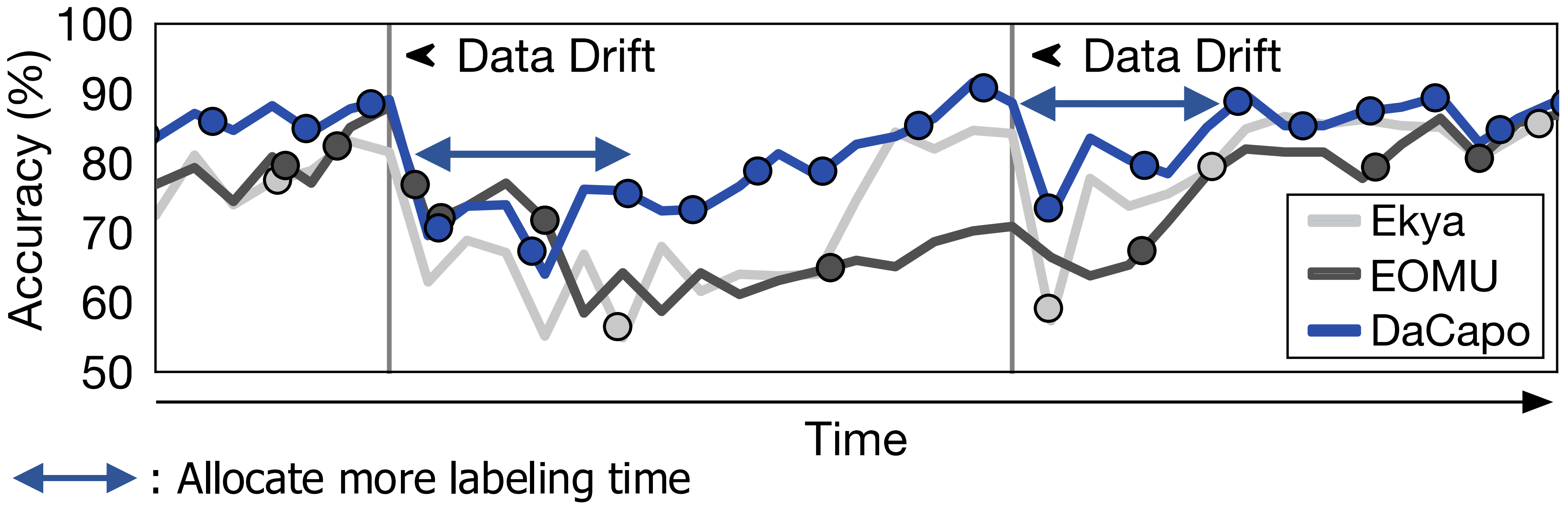
End-to-End Accuracy



- **DaCapo achieves optimal performance under on-device resources**
- **6.5% and 5.5% higher accuracy than Ekya and EOMU, respectively**

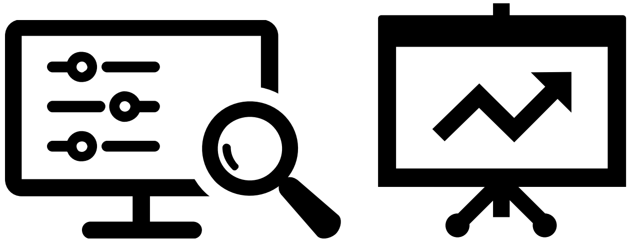
Inference Accuracy Over Time

ResNet18 & WideResNet50 Comparing to Baselines



- **Baselines struggle with data drifts, showing low accuracy trends**
- **DaCapo recovers accuracy from data drifts by adequate scheduling**

Additional Results in Paper

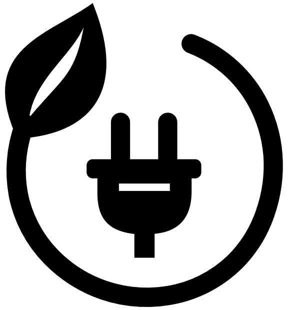


Analysis of Scheduling Decision

5.9% accuracy improvement

Extreme Data Drift Evaluation

7.6% higher accuracy



Power/Area Analysis

256x less power consumption

Conclusion

- **DaCapo**
  - On-device CL acceleration solution for autonomous systems
- **Contributions**
  - Spatially partitionable systolic array architecture
  - Fine-grained resource scheduling to handle data drift
  - PE microarchitecture using flexible low-precision arithmetic
- **Result**
  - **6.5%** and **5.5%** higher accuracy than GPU-based CL solutions, Ekya and EOMU, respectively

DaCapo is available!