AdaptiveNet: Post-deployment Neural Architecture Adaptation for Diverse Edge Environments

MobiCom '23, October 2–6, 2023, Madrid, Spain

Visit https://snspace.top/2024/03/12/AdaptiveNet/

Reporter : Sun Hao

2024.3

Salute to Authors

Institute for AI Industry Research (AIR), Tsinghua University, focuses on research geared towards the internationalization, intelligentization, and industrialization of the fourth technological revolution. Our mission is to fuel the industrial upgrade and propel social advance with AI technologies. With university-enterprise as double engines for innovation, we aim to make breakthroughs in core AI technologies, develop future industry-leaders, and achieve leapfrog progress with the industry. AIR was founded in 2020 by professor Zhang Ya-Qin, a world-renowned scientist and entrepreneur in the fields of multimedia and artificial intelligence.

Yuanchun Li
Institute for AI Industry Research (AIR), Tsinghua University

An Assistant Researcher at Institute for AI Industry Research (AIR)

Interests: Edge Systems/Applications with AI, EdgeLLM (large language models at the edge), and MobileAgent (intelligent personal agents on mobile devices) powered by EdgeLLM

Yunxin Liu
Institute for AI Industry Research (AIR), Tsinghua University

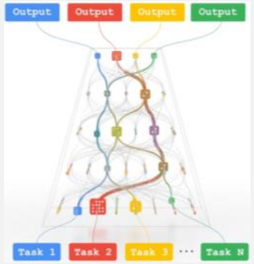
A Guoqiang Professor at Institute for AI Industry Research (AIR), Tsinghua University

Interests: Mobile computing and edge computing, including power management, security and privacy, sensing, and intelligent edge/mobile systems

AI is Transforming the World, with Cloud + Edge

Cloud AI

Multi-domain, multi-task, general-purpose services

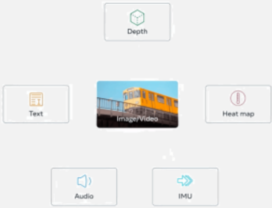


Pathways

Google

ChatGPT

OpenAI

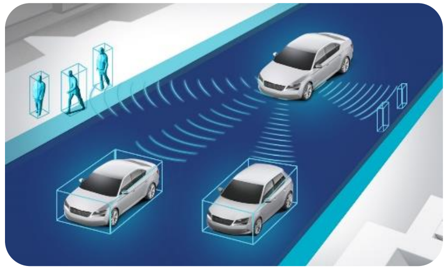


ImageBind

Meta

Edge AI

Domain-specific, real-time, privacy-sensitive applications



Auto Pilot

UAV Delivery

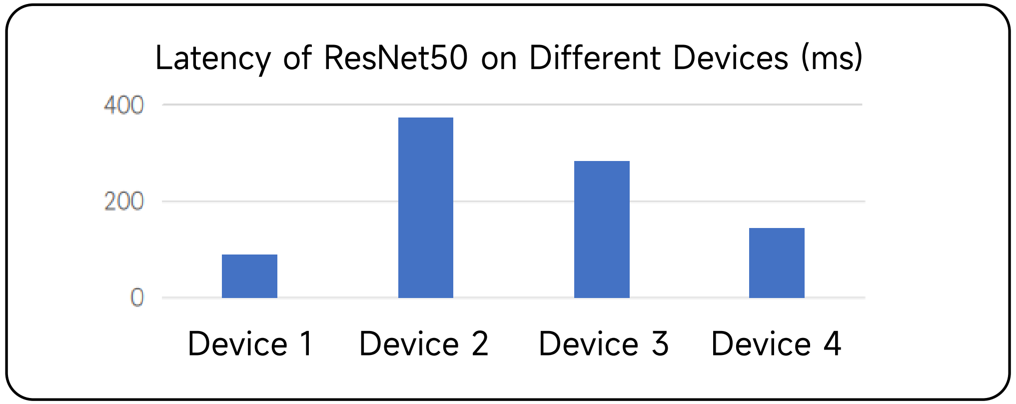
Intelligent Manufacturing

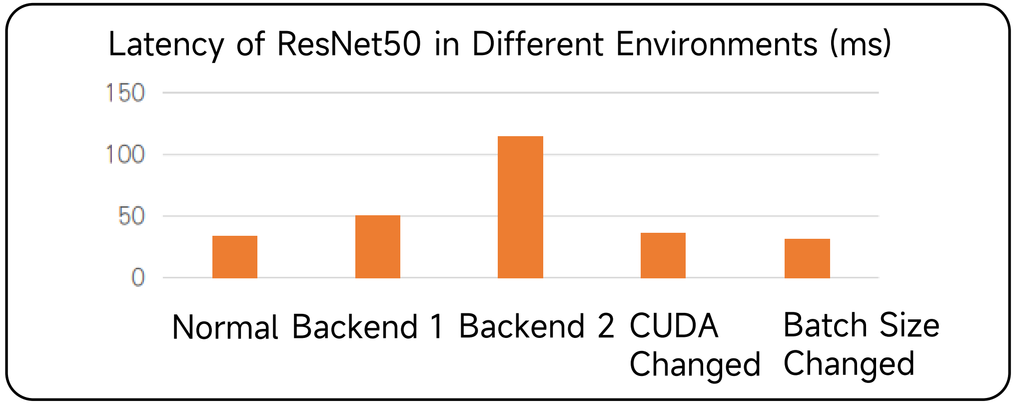
Intelligent Robot

Environment Diversity is a Main Challenge in Edge AI

- **Device diversity is a main challenge.**
  a) hardware diversity
  b) Intra-device diversity (backend number, software version, temperature)
  c) data distribution diversity

- **DNNs are expected to meet certain constant latency requirements.**





Challenge: Generate Models for Diverse Edge Environments

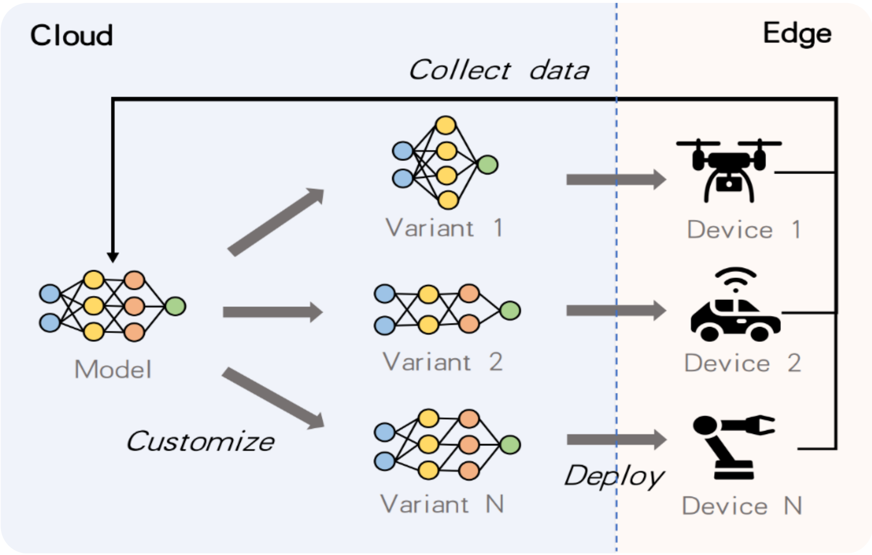
Conventional: Pre-deployment Model Generation

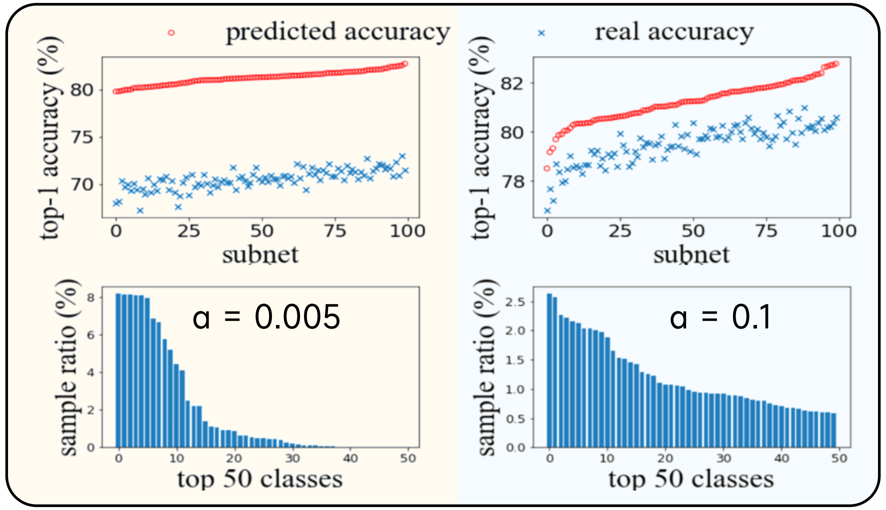
- **Most popular techniques**: Neural Architecture Search (NAS), Model Pruning, etc.

- **Limitations:**

  1. **Requires collecting privacy information** about computational resources, runtime conditions, data distribution, etc.
  2. **High maintenance cost.** Less practical in many edge/mobile scenarios where the model execution environments may be very diverse and dynamic.
  3. **Modeling the edge environment may be difficult.**

  - The cloud-based model generation relies on **accuracy and latency predictor**.
  - The unified accuracy predictor may not perform well for edge devices with **data distribution shifts**.

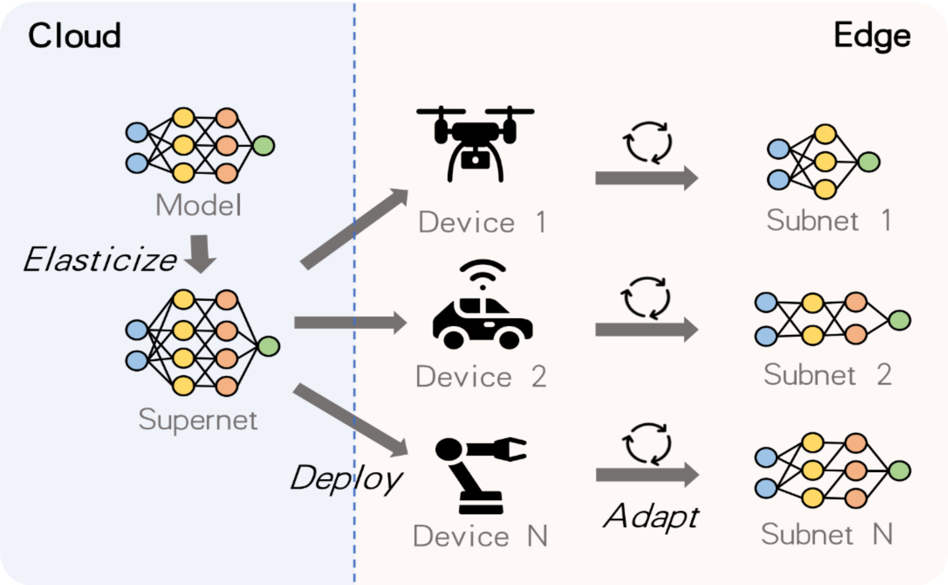




Solution: Post-deployment Neural Architecture Adaptation

Benefits:

- Directly evaluate the given DNN **<u>without accuracy predictor</u>**, which is more precise.
- A plug-and-play process, **<u>reduces the computation overhead</u>** of the cloud.
- **<u>Protect user privacy</u>**.



Related work in mobile community: on-device model scaling

LegoDNN (MobiCom'21)
NestDNN (MobiCom'18)
…

- **Limited model space**
- **Still relying on performance predictors**

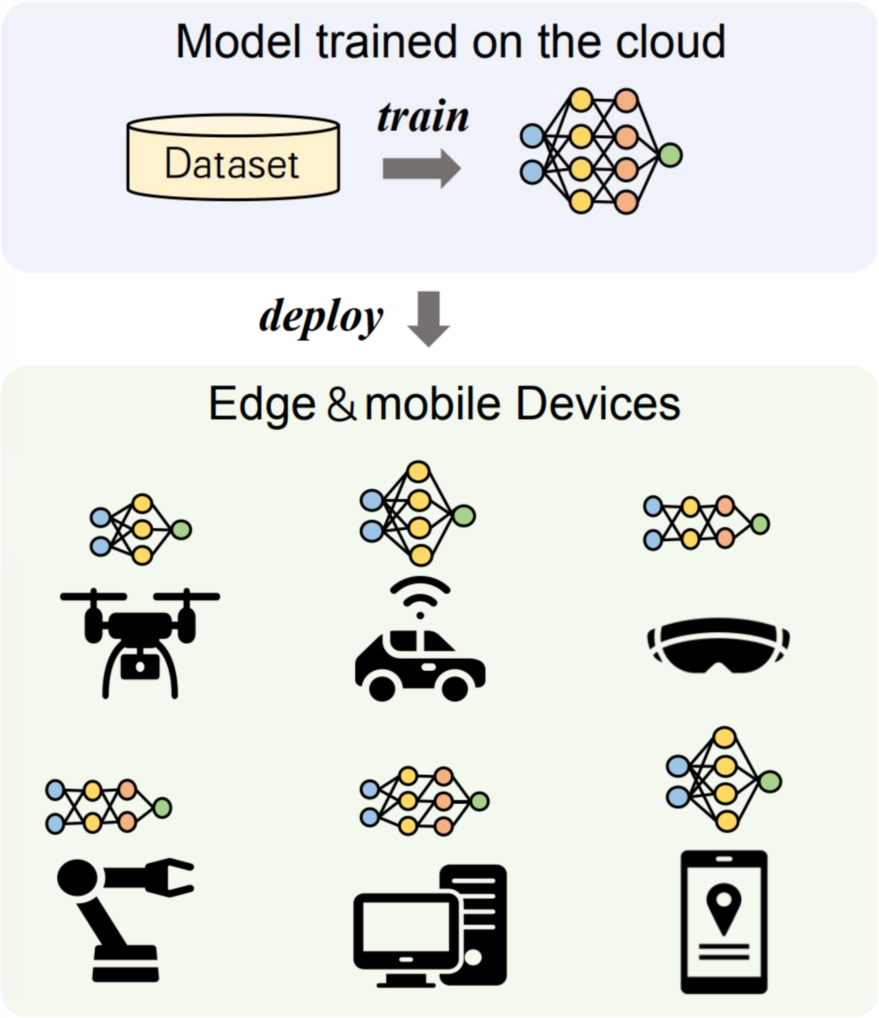
Challenges

Generating the model search space for edge devices is difficult.

- The search space should be **large** and **flexible** enough.
- Should contain **high-quality candidate models** for edge devices.

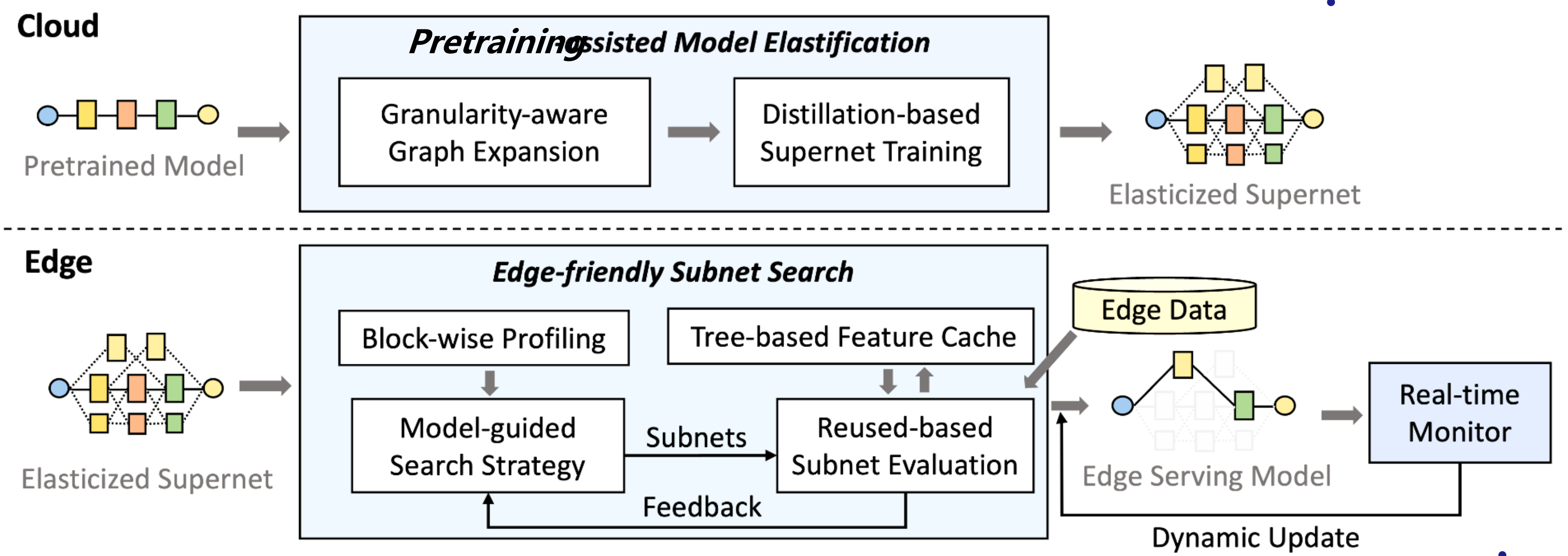
The model search process can be time-consuming at the edge.

- **Limited computing resources** and tight deadline of model initialization.
- The edge environment is **dynamic**.



AdaptiveNet: System Design

Model search space Generating



Model search

Cloud Stage: Graph Expansion

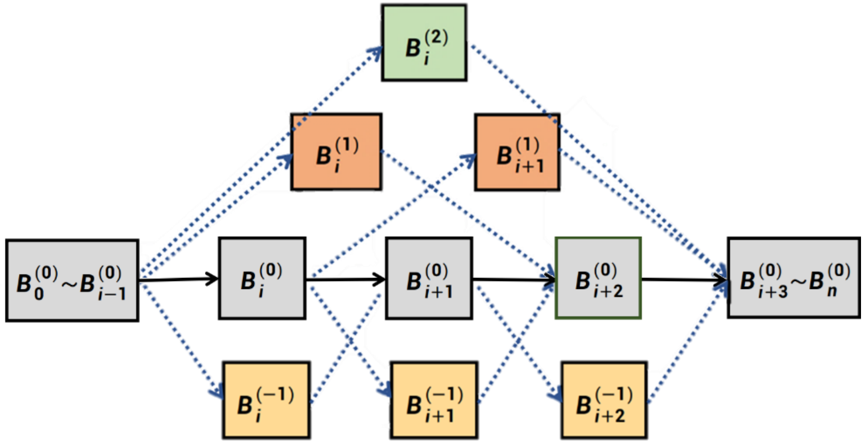
1. Given an arbitrary pre-trained DNN, AdapativeNet discovers the repeating **basic blocks ($B_0^{(0)}$~$B_n^{(0)}$)** in the DNN.

2. AdapativeNet converts the given pre-trained DNN into a **supernet** by adding **merged blocks ($B_i^{(1)}$, $B_i^{(2)}$)** and **pruned blocks ($B_i^{(-1)}$)**. The supernet encompasses a large search space of **subnets**.

Block Partitioning Strategy:

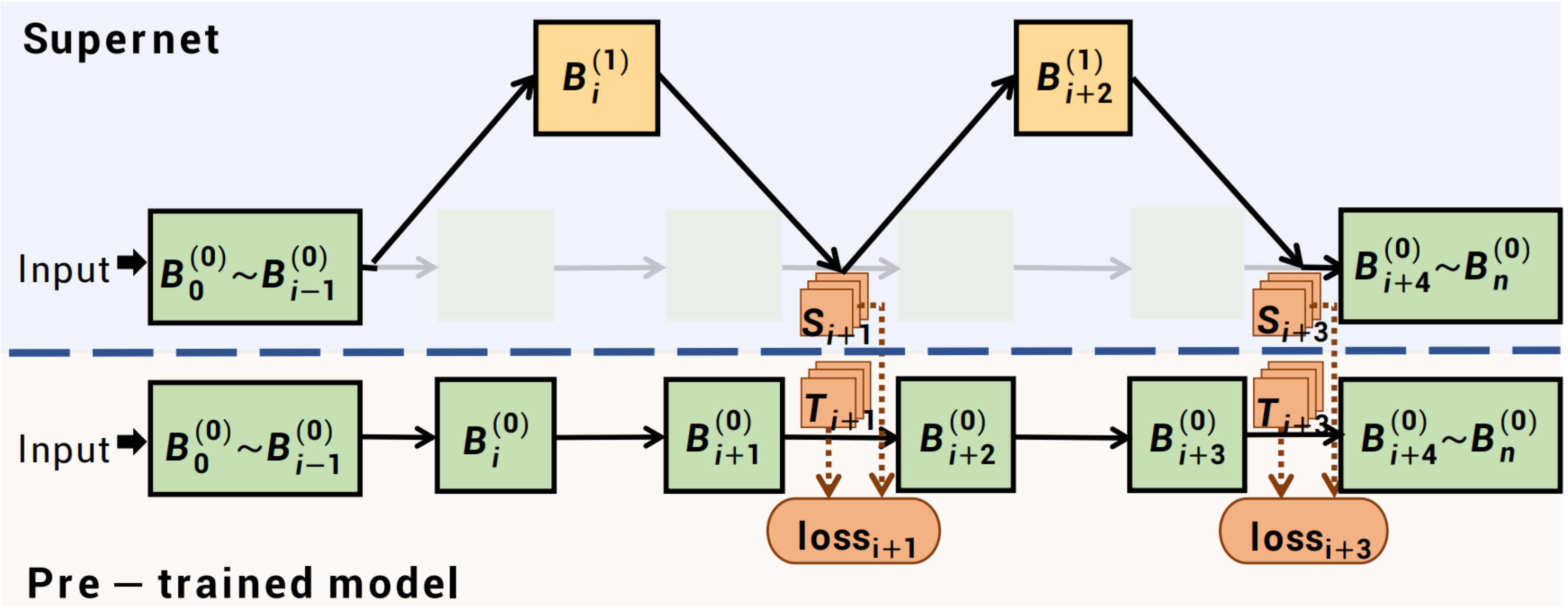
1. Limit the block parameter size.
2. The blocks should not span fusion layers.
3. Each basic block should be single-input and single-output.

Merged Blocks



Pruned Blocks

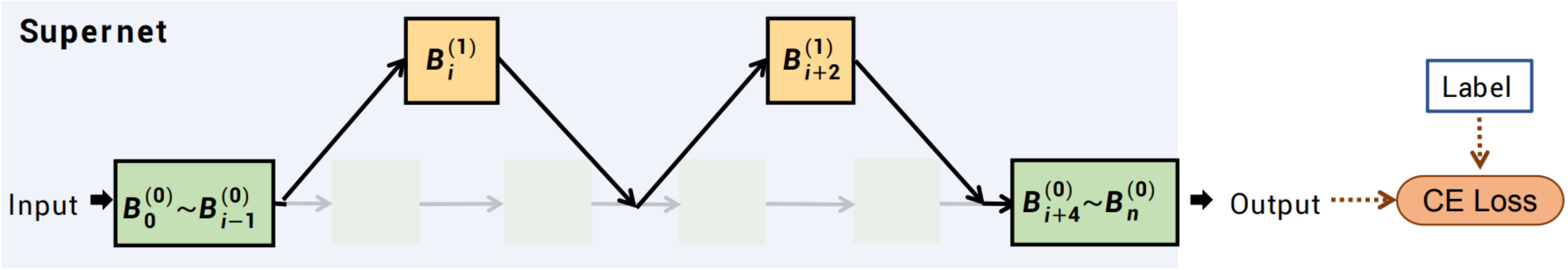
Cloud Stage: Distillation-based Training



Branch distillation phase:

- Adopt feature-based knowledge distillation (Pre-trained model as the teacher).
- In each iteration, randomly sample a subnet from the supernet and use the pre-trained model as the teacher model to train the new branches in the subnet.

Cloud Stage: Distillation-based Training



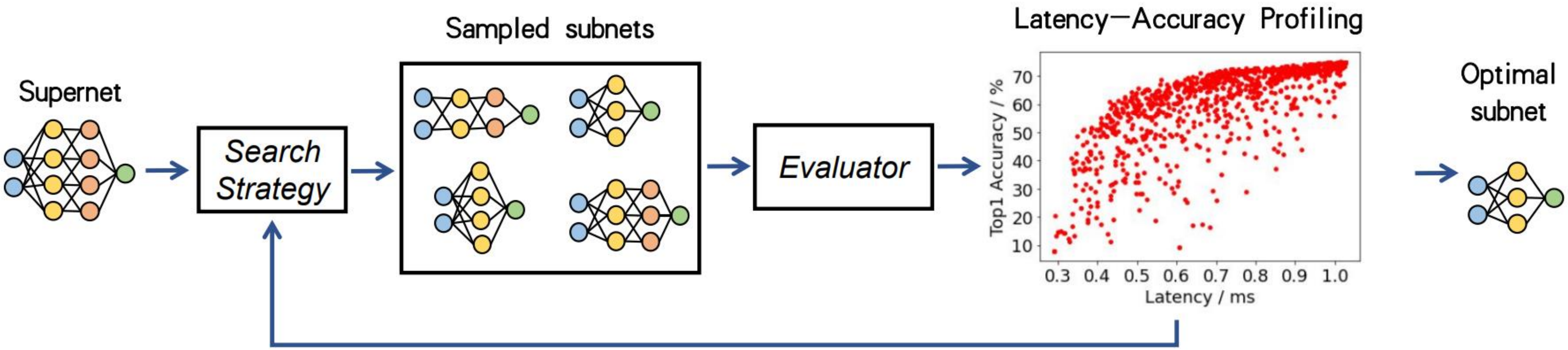
Further tuning phase:

- Further train the supernet using labelled data.
- In each iteration, randomly sample a subnet and forward a batch of samples, compute the Cross-Entropy loss and update the parameters of the new branches.

Edge Stage: Overview

Edge Stage

is to obtain the optimal architecture adaptively in the target environment by searching the subnet space.



Challenge: **Using a normal search method as in NAS can cost more than 10 hours on edge devices.** Most of the searching time is spent on **evaluating the subnets.**

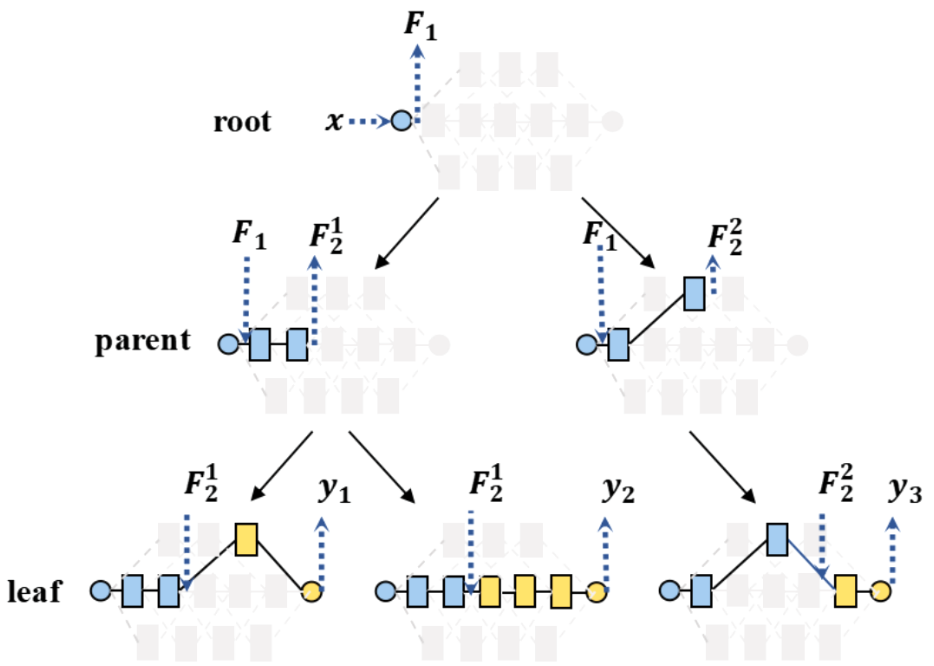
Edge Stage: Search Strategy

An example of **Genetic Algorithm (GA)** - based search strategy:

1. Build **Latency Table** $\boldsymbol{T} = \{\boldsymbol{t}_i^j\}$ ($\boldsymbol{t}_i^j$ is the latency of $\boldsymbol{B}_i^j$). Thus, the latency of a chosen subnet is the sum of all its blocks.
2. Generate the **initial candidate subnets** by randomly sampling a group of subnets whose latencies are near the **latency budget**.
3. In each iteration, mutate subnets by replacing branches. **(Make sure the mutated subnets are also near the latency budget)**.

Edge Stage: Evaluator

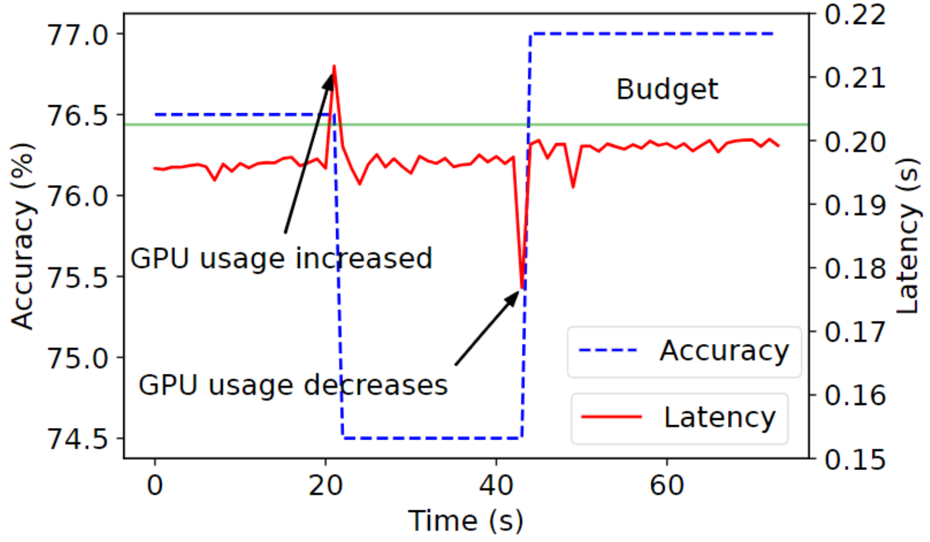
- In each iteration, it is usually needed to evaluate **hundreds of candidate subnets** with the edge data to find the most accurate ones.
- The candidate subnets usually share common prefix substructures, so it is possible to reuse common intermediate features across subnets.
- Introduce a **tree-based feature cache** to schedule the evaluation (Right Figure).



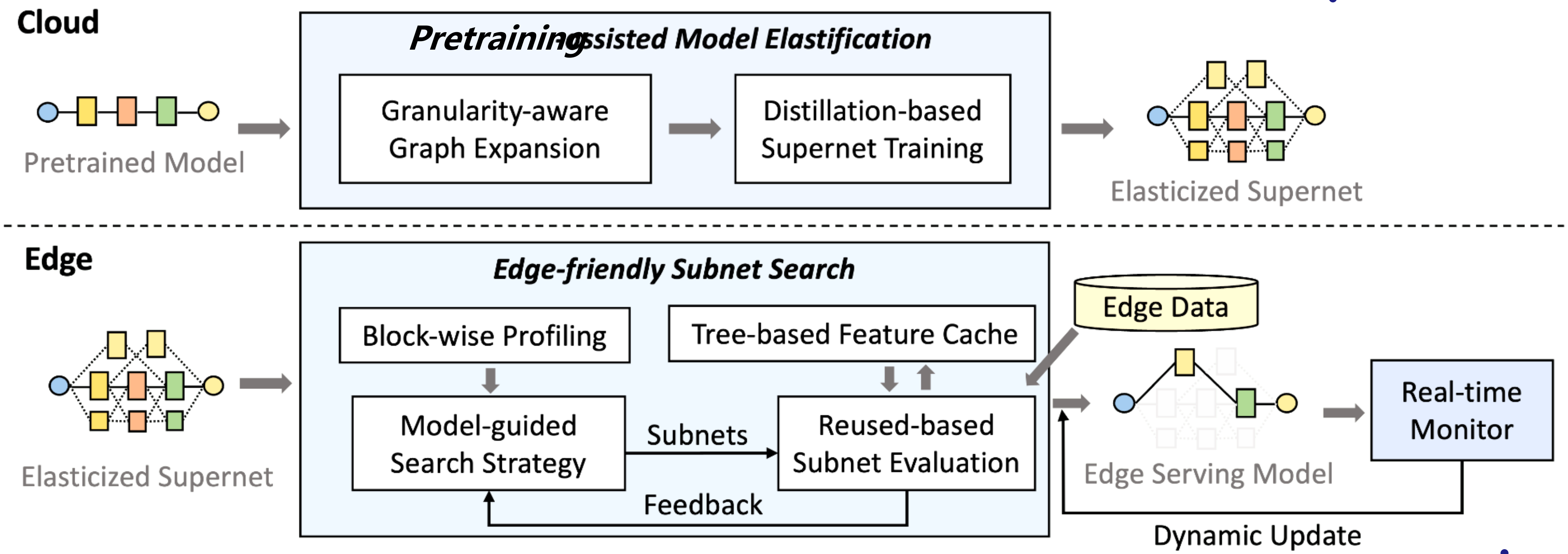
Tree-based Feature Cache

Edge Stage: Dynamic Model Update

- After searching, the subnets achieving the highest accuracy at different levels of latency are saved.
- AdaptiveNet **dynamically pages in and pages out alternative blocks** when the environment changes.



Model search space Generating



Model search

Evaluation: Experimental Setup

Task	Model	Dataset
Image classification	MobileNetV2, ResNet	ImageNet2012
Object detection	EfficientDet	COCO2017
Semantic segmentation	FPN	CamVid

Edge Devices

- **Android Smartphone** (Xiaomi 12) with Snapdragon 8 Gen 1 CPU and 8 GB memory
- **Jetson Nano** with 4 GB memory
- **Edge server with NVIDIA 3090 Ti** with 24 GPU memory

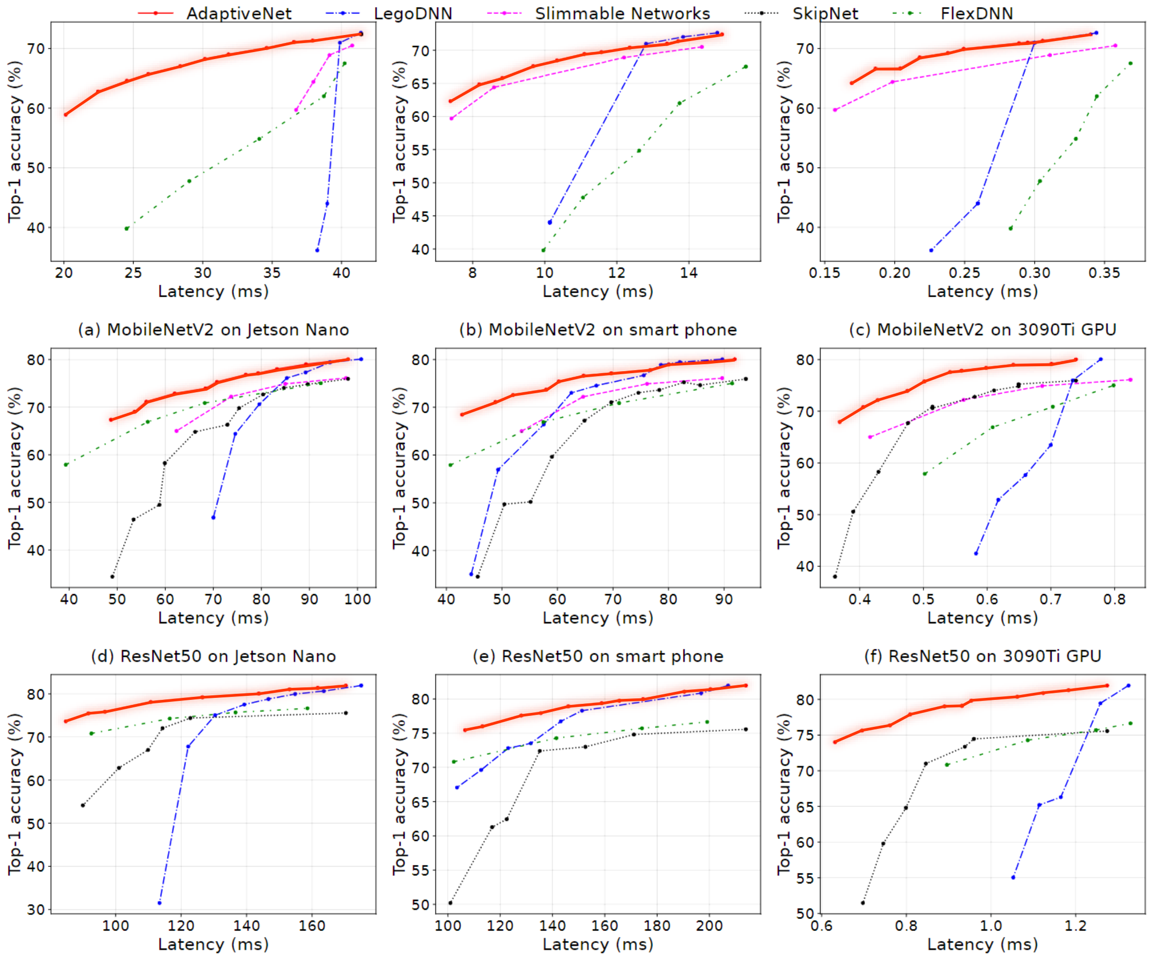
Baselines

- **LegoDNN** [1]: a pruning based, block-grained technique for model scaling.
- **Slimmable Networks** [2], **FlexDNN** [3], **SkipNet** [4]: dynamic neural networks with flexible widths, depths, and layers.

[1] Han et al. LegoDNN: Block-Grained Scaling of Deep Neural Networks for Mobile Vision. (MobiCom 2021)
[2] Yu et al. Slimmable Neural Networks. (ICLR 2019)
[3] Fang et al. FlexDNN: Input-Adaptive On-Device Deep Learning for Efficient Mobile Vision. (SEC 2020).
[4] Wang et al. SkipNet: Learning Dynamic Routing in Convolutional Networks. (ECCV 2019).

Evaluation: Model Scaling

- AdaptiveNet achieves **higher accuracy** than baseline approaches **at almost every latency budget**.
- Increases accuracy by 10.44% and 28.03% on average compared to LegoDNN with 90% and 70% latency budget respectively.
- AdaptiveNet outperforms the baseline models more at a lower latency budget thanks to the merging blocks.

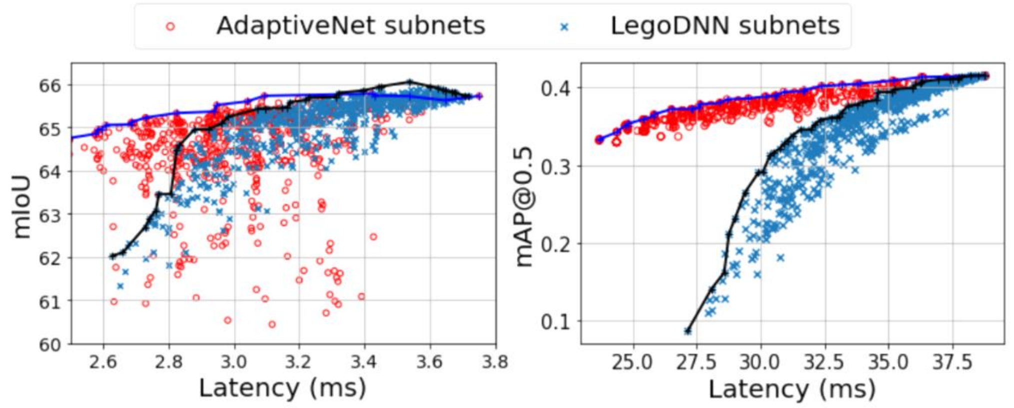


(a) MobileNetV2 on Jetson Nano (b) MobileNetV2 on smart phone (c) MobileNetV2 on 3090Ti GPU
(d) ResNet50 on Jetson Nano (e) ResNet50 on smart phone (f) ResNet50 on 3090Ti GPU
(g) ResNet101 on Jetson Nano (h) ResNet101 on smart phone (i) ResNet101 on 3090Ti GPU

↑ The higher, the better

Evaluation: Model Scaling

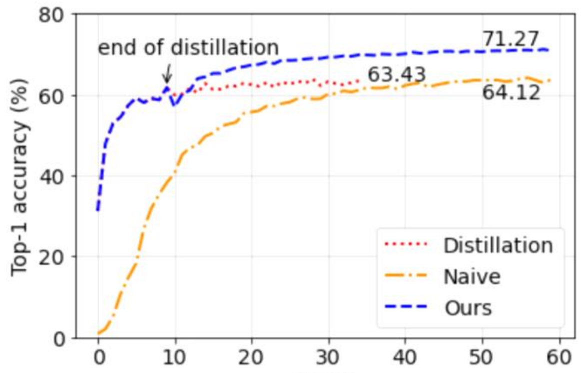
Quality of models generated for detection and segmentation tasks



(a) Semantic Segmentation (b) Object Detection

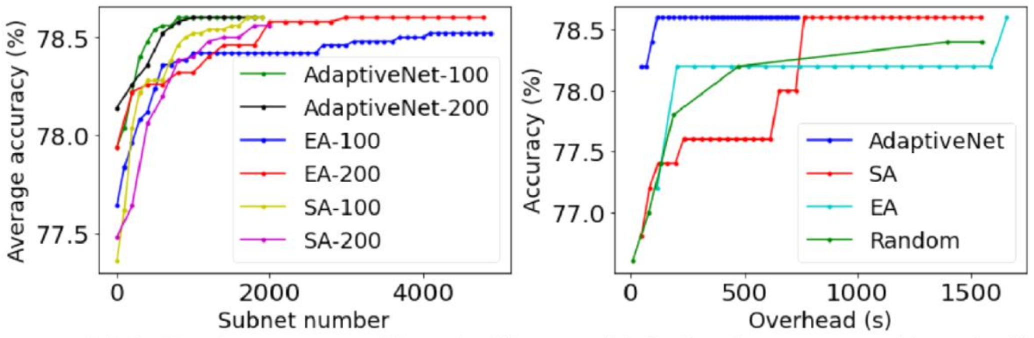
↑ The higher, the better

Training efficiency of on-cloud elastification



↑ The higher, the better

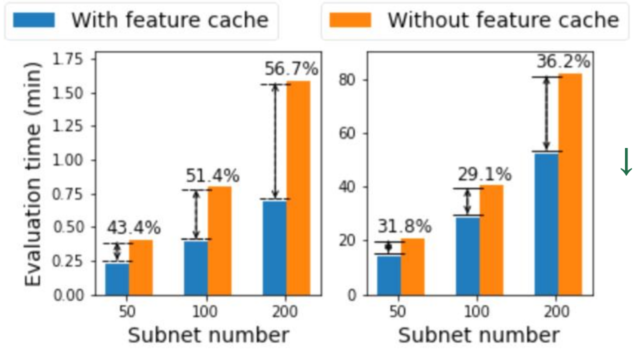
Comparison of search efficiency between different methods



(a) Optimal accuracy achieved with different num of subnets (b) Optimal accuracy achieved with different search time

↑ The higher, the better

Speed of evaluating a group of subnets



(a) 2080Ti (b) Jetson Nano

↓ The lower, the better

Summary

Topic

Generate models for diverse and dynamic edge environments.

Key Idea

Post-deployment on-device neural architecture adaptation.

Challenges

1. Generating the search space of model architectures is non-trivial.
2. The model performance evaluation process is time-consuming.

Techniques

- On-cloud model elastification method
- Model-guided Search Strategy
- Reuse-based Model Evaluation

Thanks for your attention
Q & A

AdaptiveNet: Post-deployment Neural Architecture Adaptation for Diverse Edge Environments

Reporter : Sun Hao

2024.3