Edge Intelligence III: Codesign & Edge Systems

Chapter 4

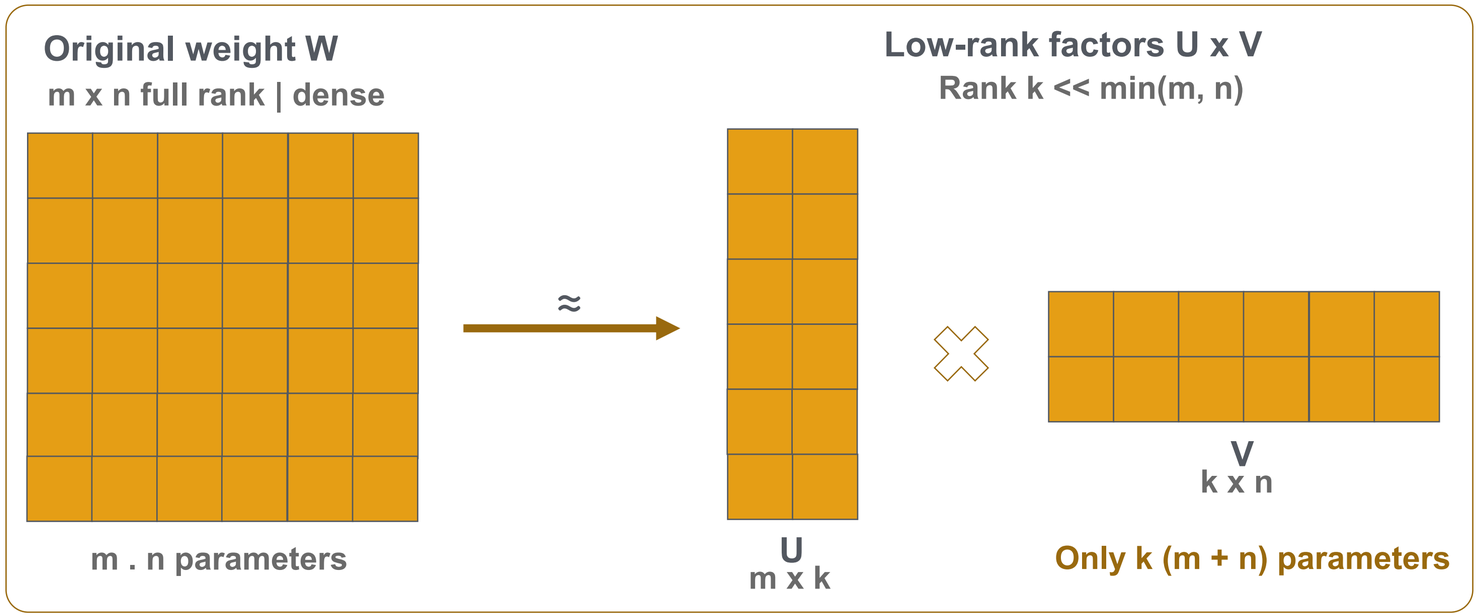
Topic: Low-rank approximation | HW-SW codesign | Lightweight Models | Training & Inference

Outline

- ❑ Low-Rank Approximation
  - Factor big weight matrices
  - SVD & rank intuition
  - Fewer params, less compute
- ❑ HW-SW Codesign
  - Co-develop model + chip
  - CODEBench, TransPIM
  - FPGA & multimodal codesign
- ❑ Lightweight Models
  - MobileNet, SqueezeNet
  - Tiny models: Bonsai, FastGRNN
  - MobileViT hybrids
- ❑ Acceleration Techniques
  - Model simplification
  - Auto-ViT-Acc
  - ViA FPGA accelerator
- ❑ Training & Inference
  - Edge training architectures
  - CLONE collaborative learning
  - TensorRT & collaborative infer
- ❑ Summary & Practice

Low-Rank Approximation

Factor the Weight Matrix



What is rank?

The number of linearly independent rows/columns — a measure of how much unique information a matrix holds. Correlated rows → low rank → compressible.

Decompose one big multiplication into two small ones

Correlated rows mean W is effectively low-rank- SVD or factorization recovers it, cutting compute.

Denton et al. SVD on FC layers → 3× conv speedup, <1% precision loss.

Denil et al. Low-rank cuts the number of dynamic parameters.

Sainath et al. Low-rank factorization on the final DNN layer for acoustic models.

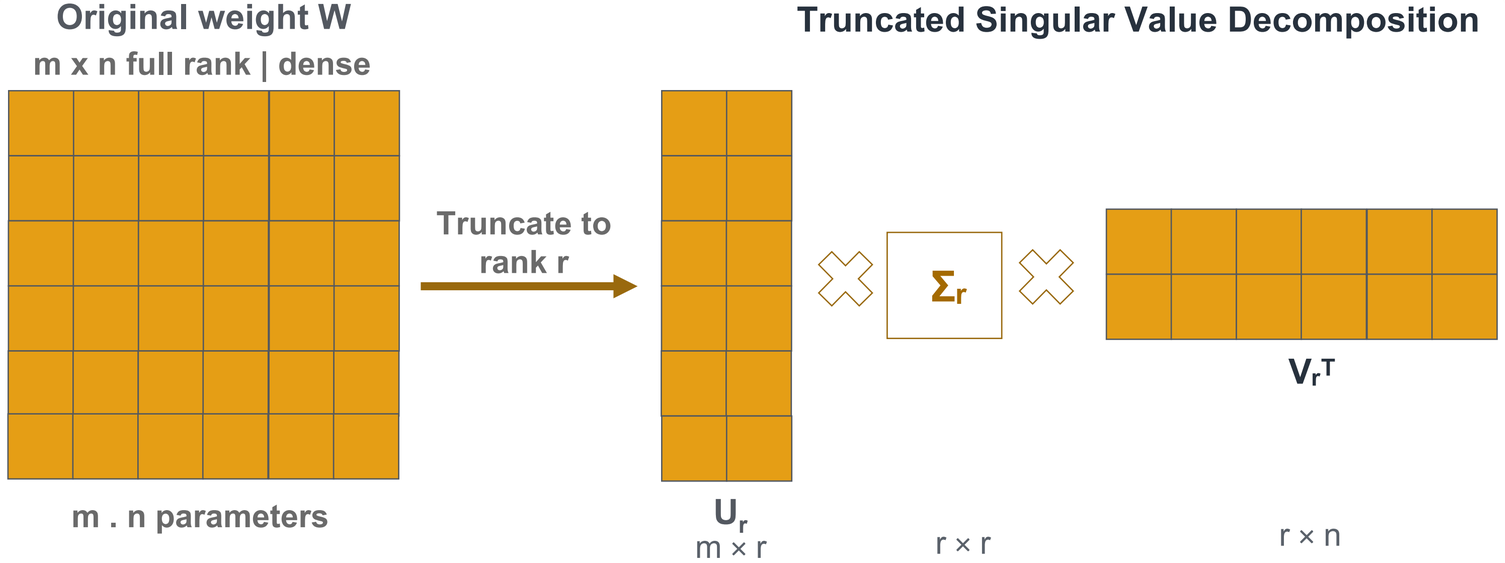
Emily L Denton et al. "Exploiting linear structure within convolutional networks for efficient evaluation". In: Advances in Neural Information Processing Systems 27 (2014).

Misha Denil et al. "Predicting parameters in deep learning". In: Advances in Neural Information Processing Systems 26 (2013).

Tara N Sainath et al. "Low-rank matrix factorization for deep neural network training with high-dimensional output targets". In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE. 2013, pp. 6655–6659.

Low-Rank Approximation: Singular Value Decomposition (SVD)

SVD provides fewer parameters with less compute



For deployment, absorb Σ_r into either factor: $A = U_r\Sigma_r$, $B = V_r^T$

One large multiply becomes two smaller multiplies

Efficiency Gain

Parameters

Before **mn** After **r(m +n)**

Example

W = 1024 × 1024 **1,048,576**

r = 128 **262,144**

≈ 75% fewer parameters

Compute

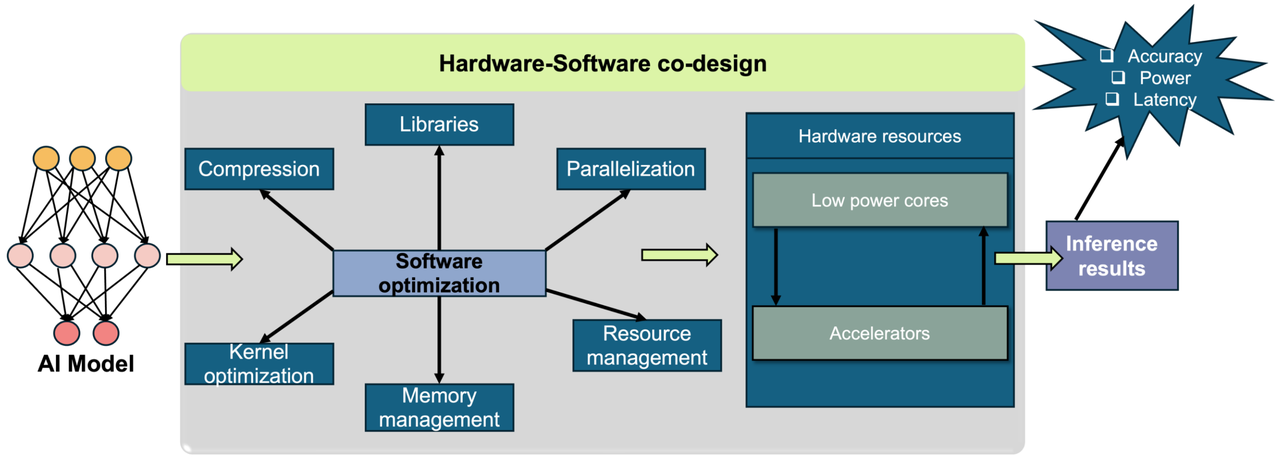
O(mn) → O[r(m+n)]

SVD preserves the dominant structure of W while discarding low-information directions.

Less memory • Fewer MACs • Faster inference

Hardware-Software (HW-SW) Codesign

Model and hardware should be designed together for edge



Overview of hw-sw codesign

The idea.

Software and hardware are developed simultaneously to maximize an edge device's performance, efficiency and scalability.

Software part

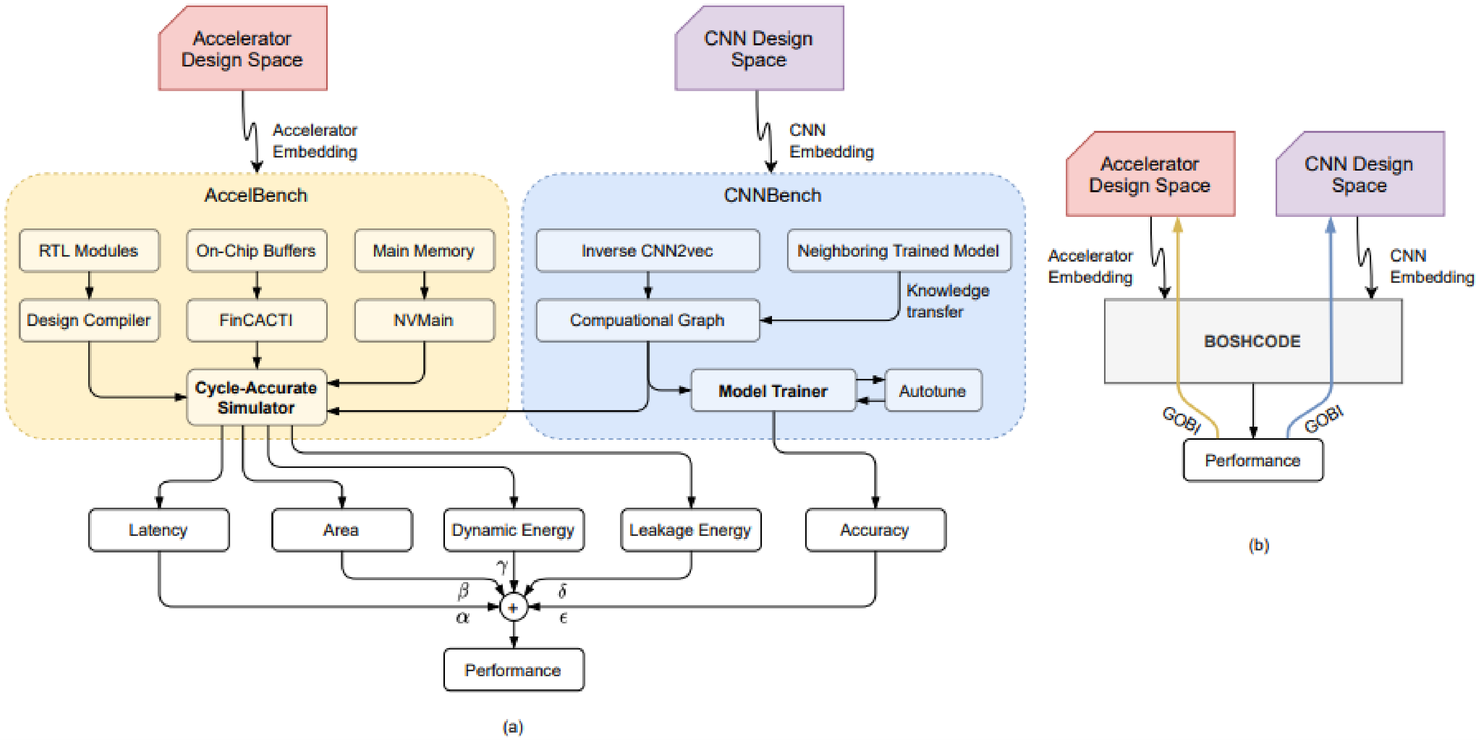
Compression, kernel & memory optimization, parallelization, resource management, libraries.

Hardware part

Low-power cores, accelerators & hardware resources tuned to the workload.

HW-SW Codesign: Frameworks

CODEBench



CODEBench

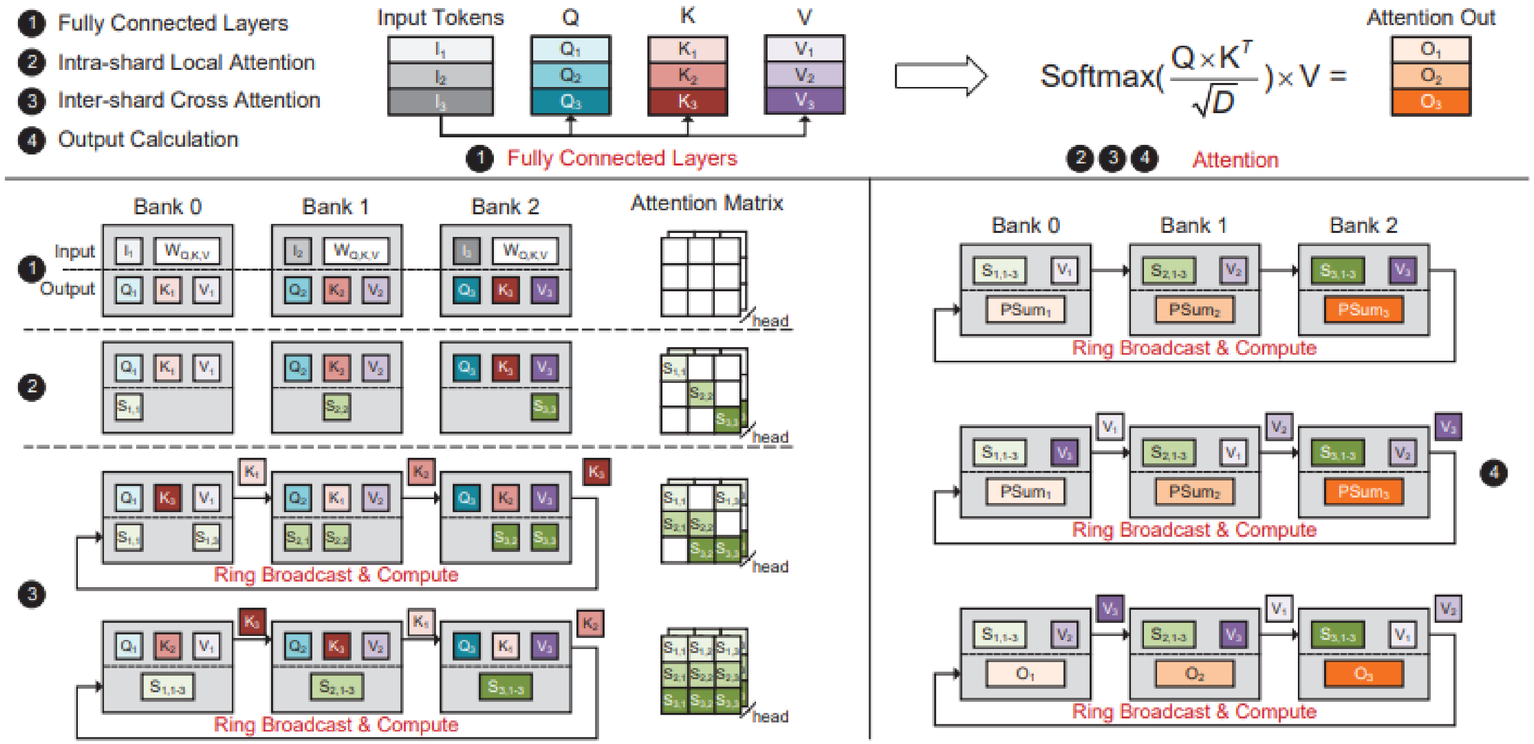
Pairs CNNBench (Bayesian architecture search) with AccelBench (cycle-accurate accelerator sim). Beats SOTA on ImageNet & Cifar for accuracy, latency and energy.

CODEBench pipeline that includes CNNBench, AccelBench, and a novel co-design method, BOSHCODE: (a) The CNN and accelerator design spaces are sampled for novel CNN-accelerator pairs that are simulated using the CNNBench and AccelBench frameworks. (b) BOSHCODE learns a surrogate function from these design spaces to predict the performance of each CNN-accelerator pair, implementing GOBI to predict the next pair to be simulated in the active learning framework.

Shikhar Tuli et al. “CODEBench: A neural architecture and hardware accelerator co-design framework”. In: ACM Transactions on Embedded Computing Systems 22. 3 (2023), pp. 1–30.

HW-SW Codesign: Frameworks

TransPIM



Token-based data sharding scheme and the dataflow of Transformer encoder in TransPIM.

TransPIM

software–hardware codesign for transformers via processing-in-memory. Token-based dataflow + light HBM tweaks give 2.0× the throughput of ASIC accelerators.

Minxuan Zhou et al. “TransPIM: A memory-based acceleration via software-hardware codesign for transformer”. In: 2022 IEEE International Symposium on High-PerformanceComputer Architecture (HPCA). IEEE. 2022, pp. 1071–1085.

HW-SW Codesign: Frameworks

Additional

FPGA codesign

Custom PEs (systolic arrays) map models onto reconfigurable fabric. Anupreetham et al. reach 8× detection throughput; Sparse-YOLO targets YOLOv2 on CPU+FPGA.

Multimodal & AV

Li et al. weigh BEVDet (camera) vs. PointPillars (LiDAR) on FPGA/GPU. Hao & Chen survey codesign for multimodal multitask autonomous systems.

Cong Hao and Deming Chen. “Software/hardware co-design for multi-modal multi-task learning in autonomous systems”. In: 2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS). IEEE. 2021, pp. 1–5.

Lightweight Models

Resource-constrained edge devices are limited in compute, memory, storage, energy and bandwidth. So, models must be small by design. Two approaches:

Human-designed

Expert-crafted efficient architectures

- Depth-wise separable convolutions (MobileNets)
- Group convolutions (Xception)
- Proven, but time-consuming to design

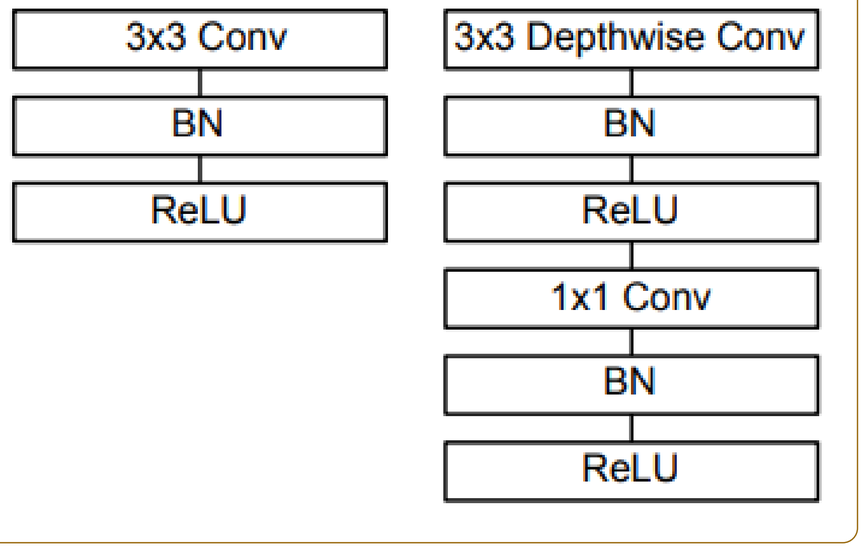
Neural Architecture search

Let AI find the best architecture

- Automated search: NASNet, AmoebaNet
- Competitive — sometimes superior — accuracy
- Limited by heavy hardware requirements

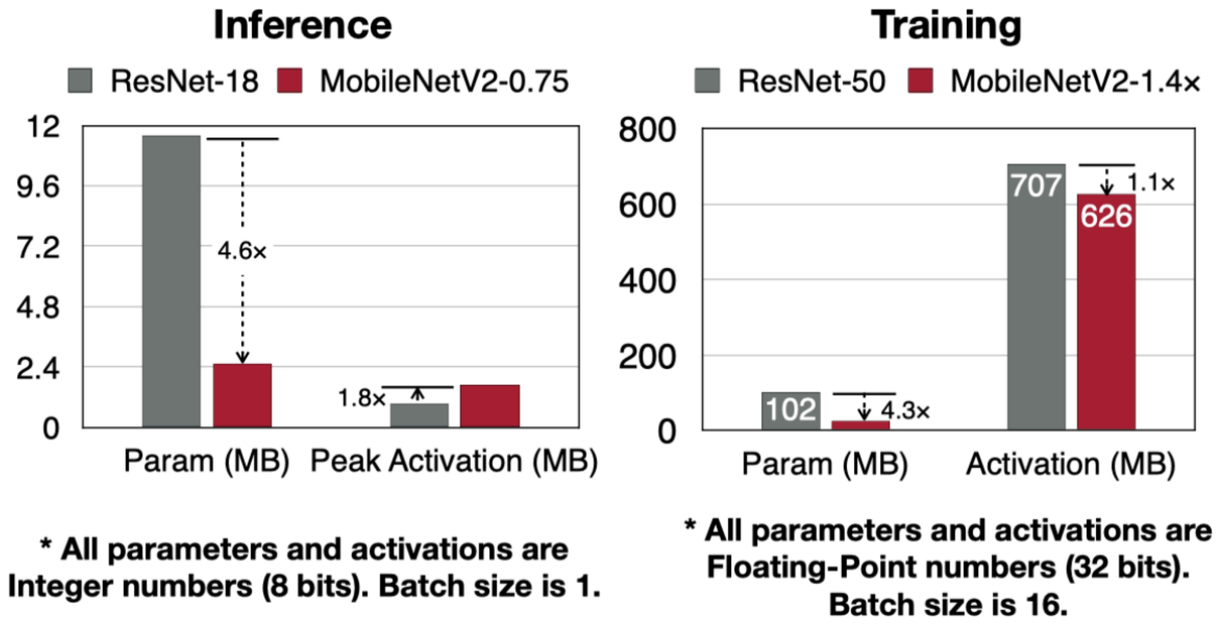
Lightweight Models

Mobilenets



Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

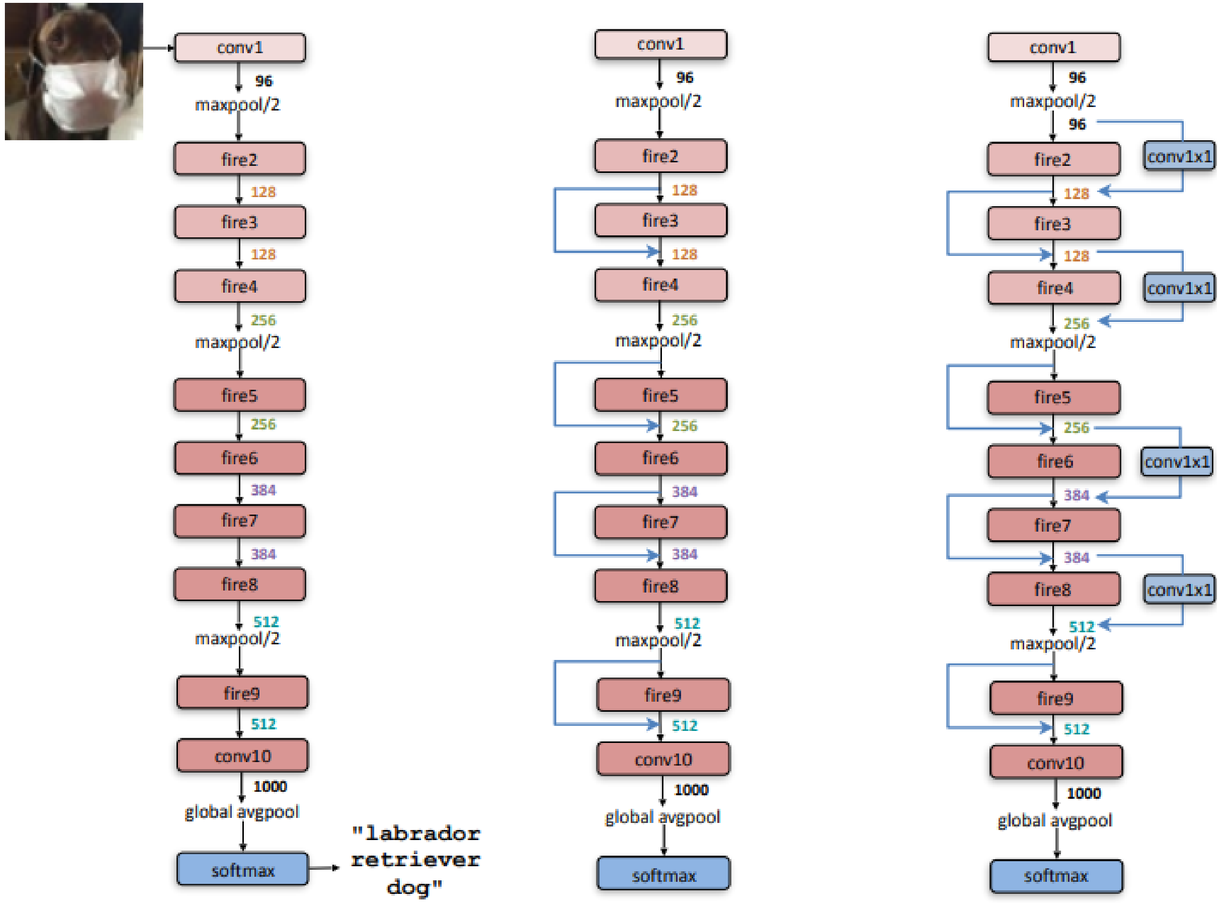
- Problem: compared to normal convolution, depthwise convolution has a much lower capacity
- Solution: Increase the depthwise convolution's input and output channels to improve its capacity
  - convolution's cost only grows linearly, so the cost is still affordable.
  - However, this design is not memory-efficient for both inference and training.



Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).

Lightweight Models

SqueezeNet



(a) SqueezeNet *(b) SqueezeNet with simple bypass* *(c) SqueezeNet with complex bypass*

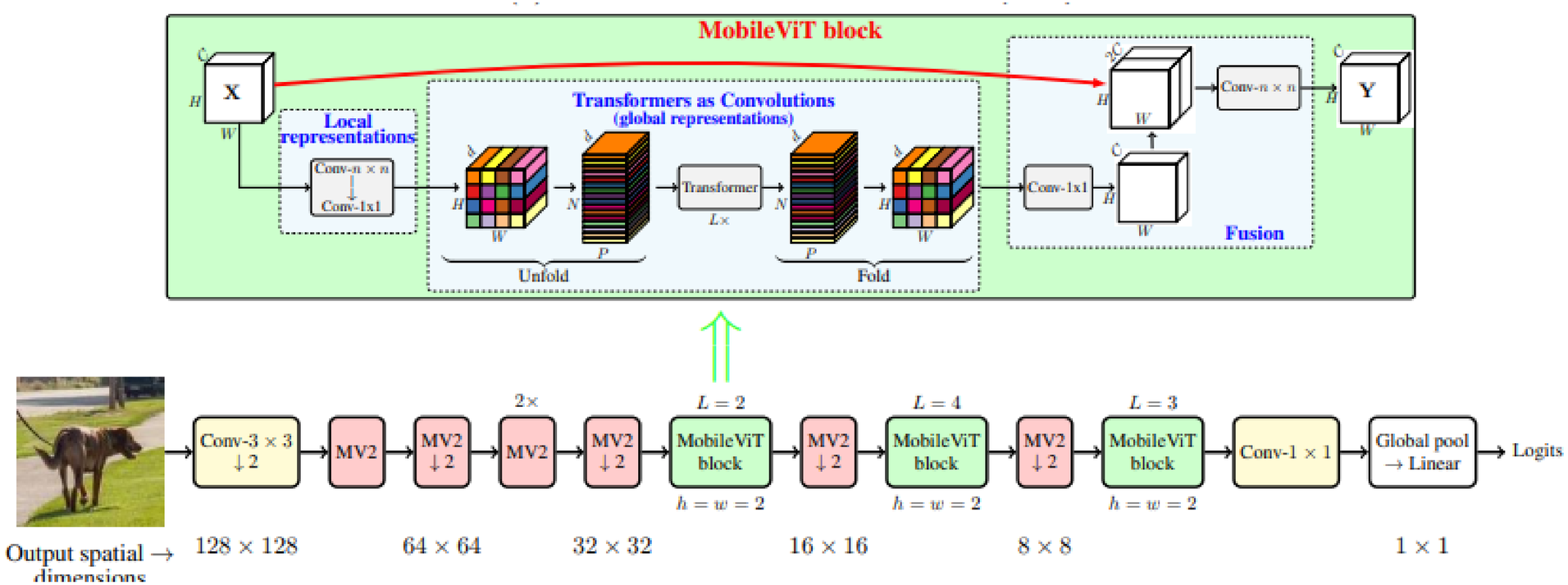
SqueezeNet

AlexNet-level accuracy with 50× fewer parameters (510× smaller).

Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." arXiv preprint arXiv:1602.07360 (2016).

Lightweight Models

MobileViT



MobileViT

MobileViT

Combines CNNs with Transformers to capture both local features and global context efficiently on mobile devices.

MobileViTV2

Improves MobileViT with separable self-attention, reducing computational cost and improving efficiency while maintaining strong accuracy.

Mehta, Sachin, and Mohammad Rastegari. "Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer." arXiv preprint arXiv:2110.02178 (2021).

Lightweight Models

Additional studies

Xception / Flattened

Depth-wise separable & 1-D filters for fast, compact feedforward nets.

Bonsai / ProtoNN

Microsoft — tree & KNN models that fit tiny IoT devices (2 KB RAM Arduino).

EMI-RNN / FastGRNN

Kilobyte-sized RNNs — 72× less compute than LSTM, +1% accuracy.

Acceleration Techniques

Overparameterization

Training needs many parameters to capture model fluctuations — but inference needs far fewer. So the model can be simplified after training, before edge deployment.

- Less computation → lower power & time
- Smaller footprint → lower-end devices
- Complementary to hardware acceleration

ViT acceleration on FPGA

Auto-ViT-Acc

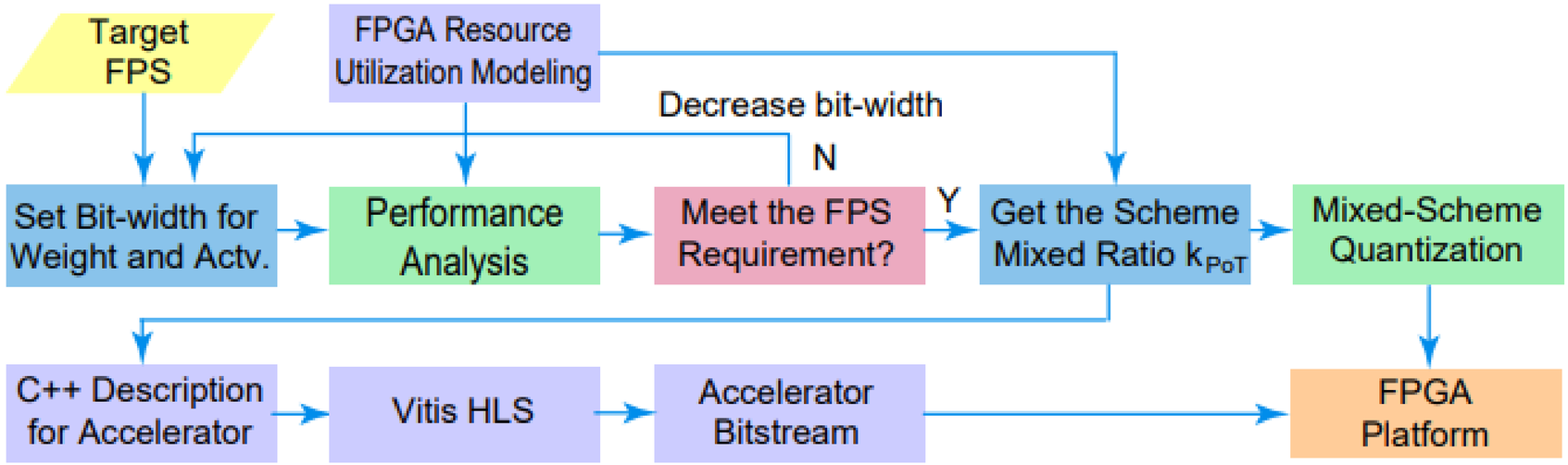
Mixed precision across transformer blocks; 0.47–1.36% higher Top-1 at the same bit-width.

ViA

Half-layer mapping + reuse engines beat NVIDIA Tesla V100 on energy efficiency.

Acceleration Techniques : ViT acceleration on FPGA

Auto-ViT-Acc



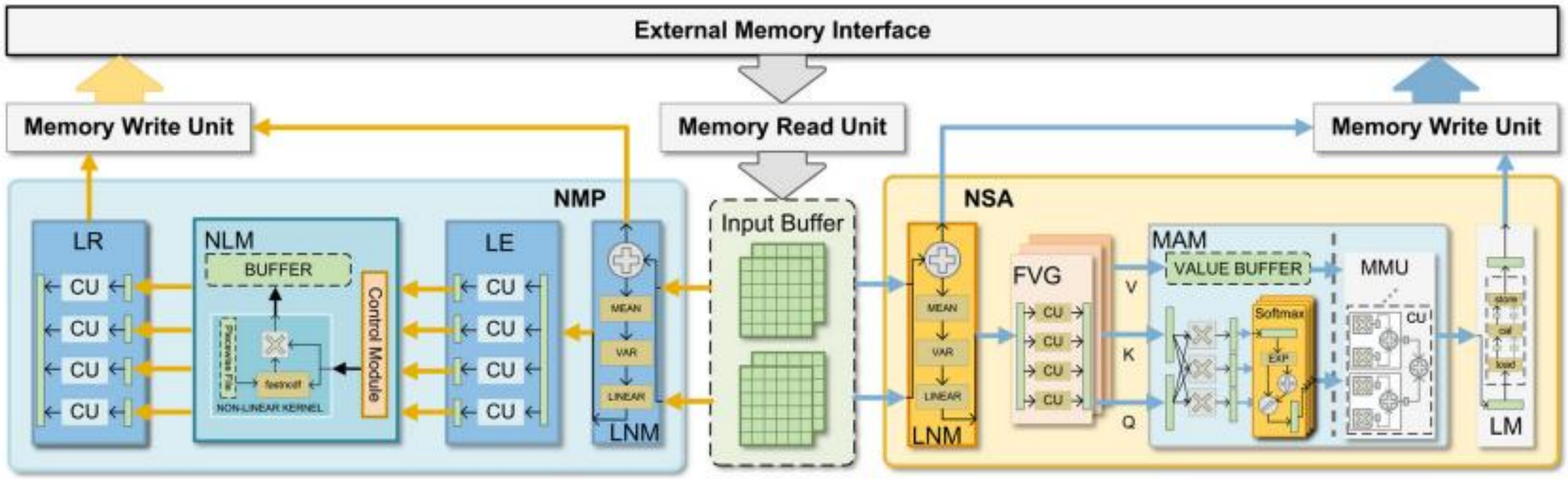
Overview of Auto-ViT-Acc framework

- Uses mixed-scheme quantization combining **fixed-point + Power-of-Two (PoT)** quantization.
- Achieves **56.8 FPS vs. 10 FPS** (~5.6× higher) than the FP32 FPGA baseline for DeiT-Base, with only 0.71% accuracy drop on ImageNet.

Li, Zhengang, et al. "Auto-vit-acc: An fpga-aware automatic acceleration framework for vision transformer with mixed-scheme quantization." *2022 32nd International Conference on Field-Programmable Logic and Applications (FPL)*. IEEE, 2022.

Acceleration Techniques : ViT acceleration on FPGA

ViA



Overview of Auto-ViT-Acc framework

- ViT-specific FPGA accelerator designed around the actual computation flow of Vision Transformers.
- Reports about **5.2× higher energy** efficiency than NVIDIA Tesla V100 and 4–10× improvement over prior FPGA accelerators.

T. Wang et al., "ViA: A Novel Vision-Transformer Accelerator Based on FPGA," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 41, no. 11, pp. 4088-4099, Nov. 2022, doi: 10.1109/TCAD.2022.3197489

Training & Inference on Edge

Where and How Models Learn

➢ Architecture

Centralized

Client edges request services from a central server / cloud (client–server).

Decentralized

Peer-to-peer — edges share and consume resources directly, no central server.

➢ Optimization goals

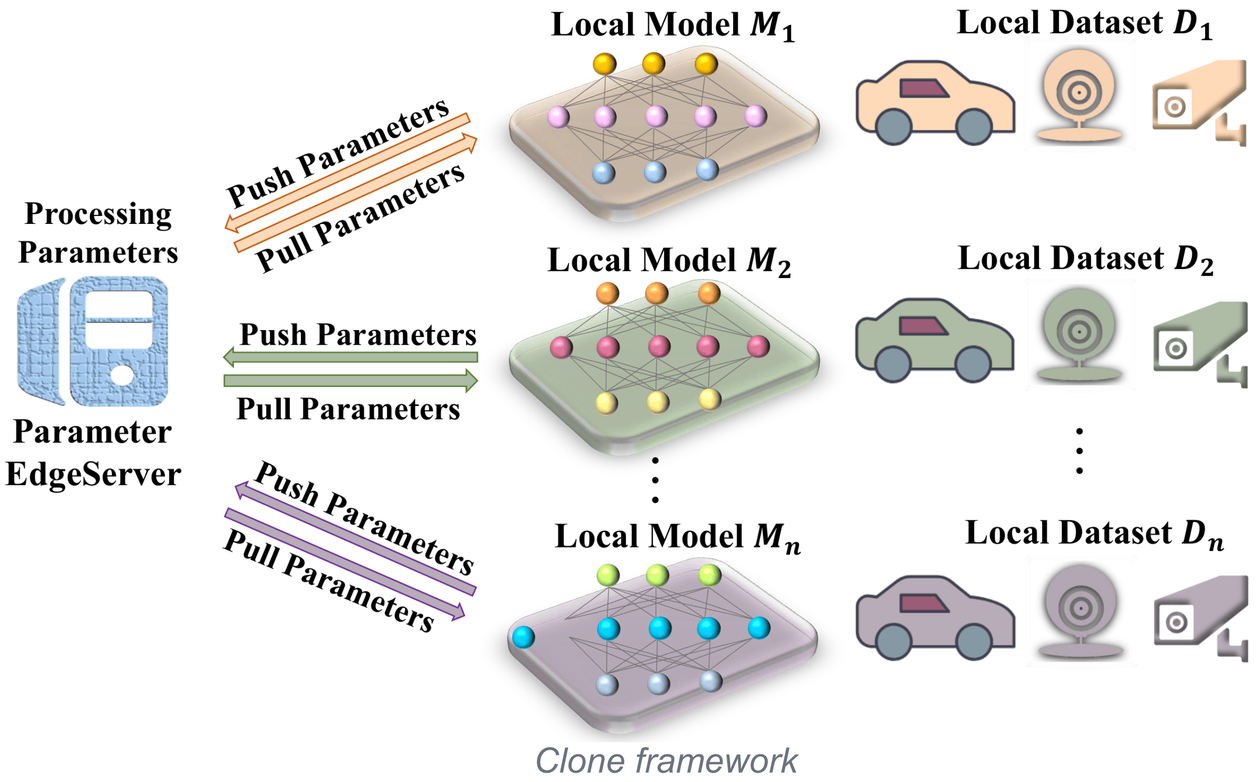
- Processing efficiency — transfer learning, cache features, learn from peers
- Communication efficiency — fewer, smaller updates; gradient quantization & sparsification

Why train at the edge?

- **Privacy** Data stays local instead of going to a central server.
- **Personalization** Models adapt to the data each device actually sees.
- **Robustness** Less dependence on the network; resilient services.

Training & Inference on Edge: Collaborative Training

Clone: learning on the edges



Clone framework

The idea. Each edge node trains on its own private data, then sends parameters to a central Parameter EdgeServer that aggregates and returns them.

Local data stays put
Nodes train on private datasets $D_1 \ldots D_n$ — raw data never leaves the device.

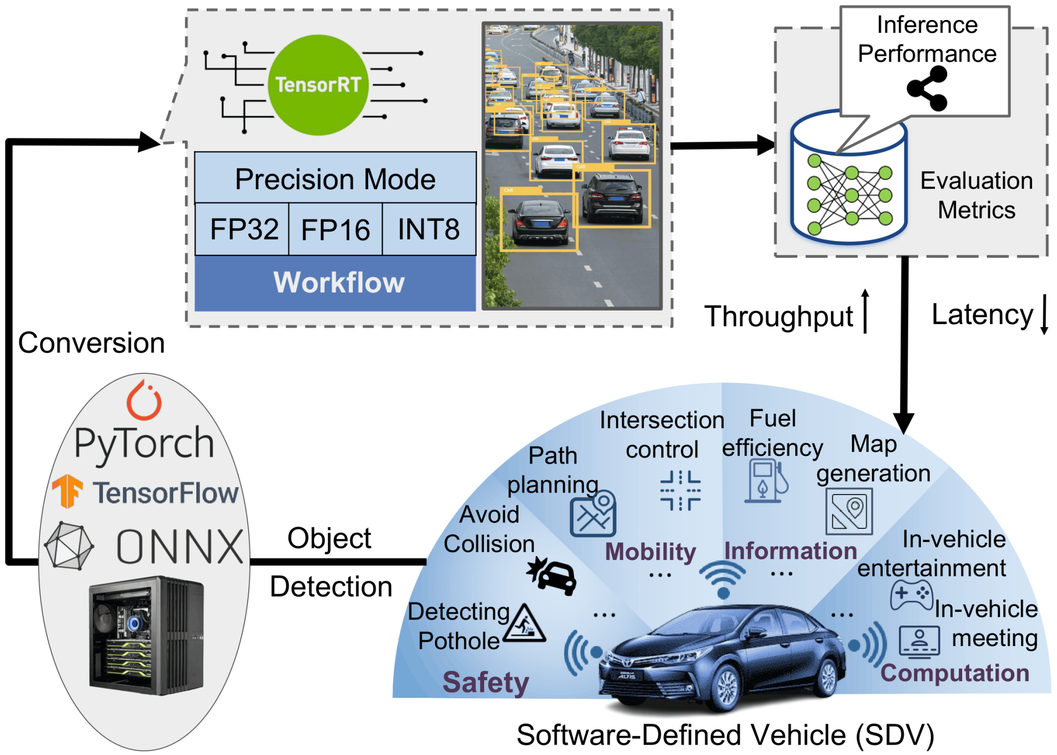
Push / pull parameters
Only model parameters travel to and from the EdgeServer.

Lower latency & privacy
Demonstrated on connected vehicles — faster and more private than cloud training.

Sidi Lu, Yongtao Yao, and Weisong Shi. “CLONE: Collaborative learning on the edges”. In: IEEE Internet of Things Journal 8. 13 (2020), pp. 10222–10236.

Training & Inference on Edge: Inference on Edge

Running Models Under Tight Budgets



TensorRT-enabled framework for a Software-Defined Vehicle

Optimization tool: TensorRT.

Converts PyTorch / TF / ONNX models and runs them in FP32, FP16 or INT8 precision modes to accelerate inference on edge devices.

➢ Bridging task needs and device limits

Model design Lightweight nets (human-designed or searched) that fit edge hardware.

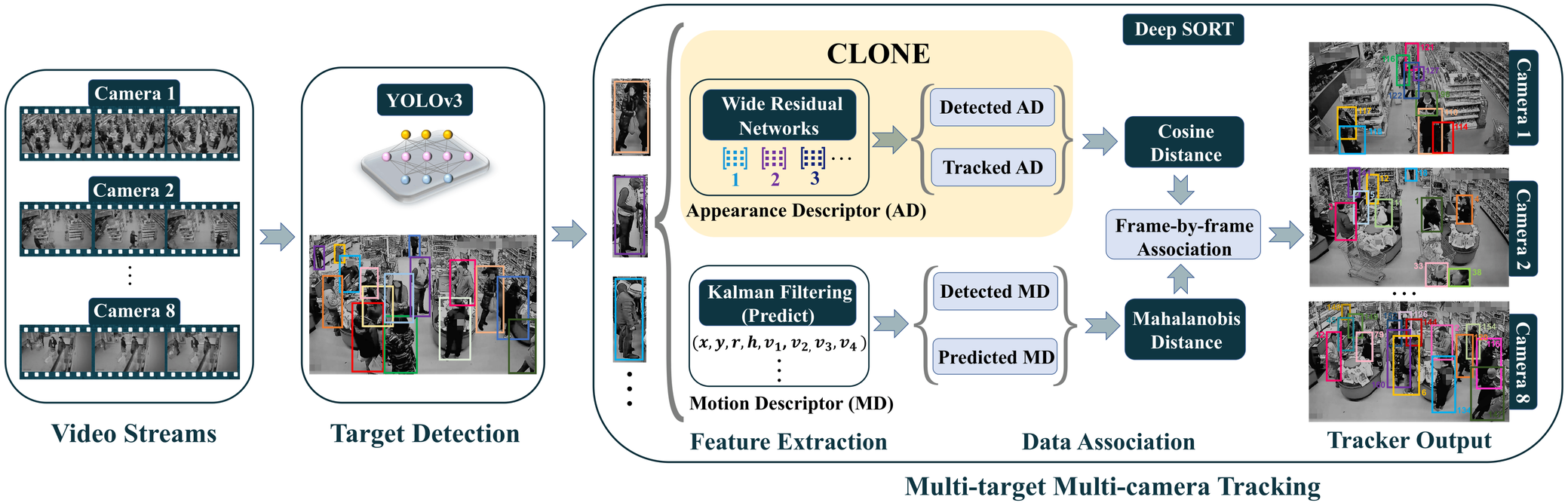
Model compression Prune, quantize, distill, low-rank — reuse of Lecture 5 techniques.

Optimization tools TensorRT precision modes trade accuracy for throughput & latency.

Sumaiya et al. “Enhancing real-time inference performance for time-critical software-defined vehicles”. In: 2024 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST). IEEE. 2024, pp. 101–113.

Training & Inference on Edge: Collaborative Inference

Many Cameras, One Tracker



Collaborative inference pipeline for multitarget multicamera tracking

Capture
Multiple camera video streams

Detect
YOLOv3 finds & boxes targets

Describe
Appearance (AD) + motion (MD) descriptors

Associate
Cosine + Mahalanobis distance match IDs

Figure 4.19 — Collaborative inference pipeline for multitarget multicamera tracking (Lu et al.).

Sidi Lu, Yongtao Yao, and Weisong Shi. “CLONE: Collaborative learning on the edges”. In: IEEE Internet of Things Journal 8. 13 (2020), pp. 10222–10236.

Summary

- Low-rank factorization reduces redundant weights and computation with minimal accuracy loss.
- Model–hardware codesign balances performance requirements with available hardware resources.
- Lightweight model design creates compact architectures suitable for edge deployment.
- Edge training and inference keep data local while enabling efficient and fast execution.

Practice

1. How does integrating hardware accelerators impact the design of machine learning models for edge deployment?

2. Discuss the role of software-hardware codesign in optimizing resource-constrained edge computing environments.

References

- Emily L Denton et al. "Exploiting linear structure within convolutional networks for efficient evaluation". In: Advances in Neural Information Processing Systems 27 (2014).
- Misha Denil et al. "Predicting parameters in deep learning". In: Advances in Neural Information Processing Systems 26 (2013).
- Tara N Sainath et al. "Low-rank matrix factorization for deep neural network training with high-dimensional output targets". In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE. 2013, pp. 6655–6659.
- Shikhar Tuli et al. “CODEBench: A neural architecture and hardware accelerator co-design framework”. In: ACM Transactions on Embedded Computing Systems 22. 3 (2023), pp. 1–30.
- Minxuan Zhou et al. “TransPIM: A memory-based acceleration via software-hardware codesign for transformer”. In: 2022 IEEE International Symposium on High-PerformanceComputer Architecture (HPCA). IEEE. 2022, pp. 1071–1085.
- Cong Hao and Deming Chen. “Software/hardware co-design for multi-modal multi-task learning in autonomous systems”. In: 2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS). IEEE. 2021, pp. 1–5.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).
- Mehta, Sachin, and Mohammad Rastegari. "Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer." arXiv preprint arXiv:2110.02178 (2021).
- Li, Zhengang, et al. "Auto-vit-acc: An fpga-aware automatic acceleration framework for vision transformer with mixed-scheme quantization." 2022 32nd International Conference on Field-Programmable Logic and Applications (FPL). IEEE, 2022.
- T. Wang et al., "ViA: A Novel Vision-Transformer Accelerator Based on FPGA," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 41, no. 11, pp. 4088-4099, Nov. 2022, doi: 10.1109/TCAD.2022.3197489.
- Sidi Lu, Yongtao Yao, and Weisong Shi. “CLONE: Collaborative learning on the edges”. In: IEEE Internet of Things Journal 8. 13 (2020), pp. 10222–10236.
- Sumaiya et al. “Enhancing real-time inference performance for time-critical software-defined vehicles”. In: 2024 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST). IEEE. 2024, pp. 101–113.