

Data-Model-Circuit Tri-Design for Ultra-Light Video Intelligence on Edge Devices

Yimeng Zhang¹, **Akshay Kamath**², Qiucheng Wu³, Zhiwen Fan⁴,
Wuyang Chen⁴, Zhangyang Wang⁴, Shiyu Chang³, Sijia Liu¹, Cong Hao²

¹Michigan State University, USA

³University of California at Santa Barbara, USA

²Georgia Institute of Technology, USA

⁴University of Texas at Austin, USA

Speaker Bio



2021 - 2023

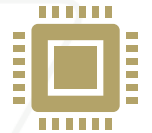
ECE graduate student at Georgia Tech, USA

Thesis Advisor : Prof. Cong (Callie) Hao

Research Area : Hardware accelerators for deep learning



Akshay Kamath



Summer '22

Interned in the SoC Design and Integration team at Apple

Silicon Engineering Group, Cupertino, USA



2017 - 2021

RTL Design Engineer at Samsung Semiconductor India Research

Cryptographic accelerator IPs, SerDes IPs, and test chips.








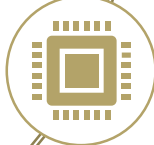
2013 - 2017

Bachelor's degree from NITK Surathkal in India

Electronics and Communication Engineering

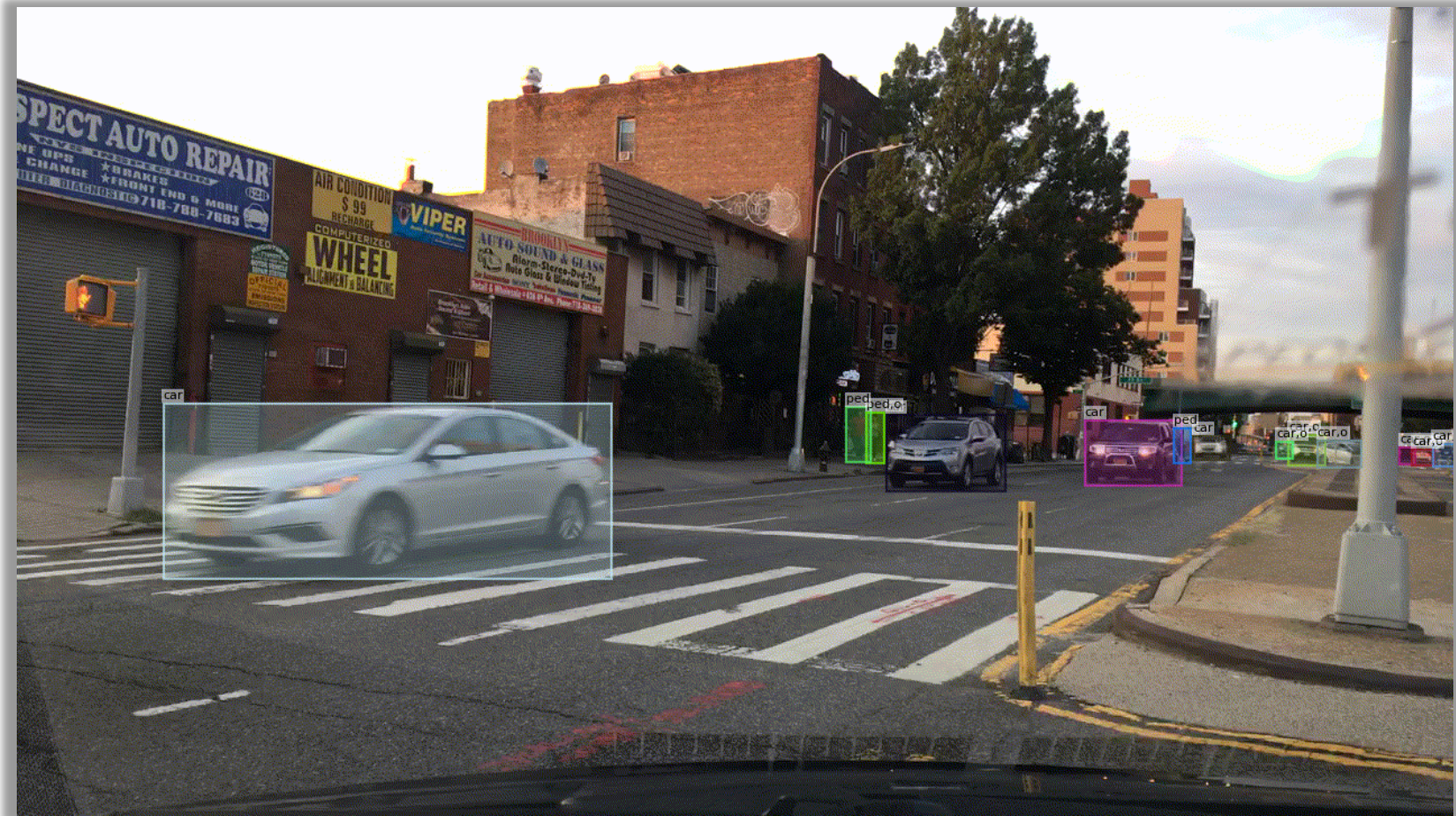
MITACS Research Intern at University of Toronto in Summer 2016

Overview

-  Multiple Object Tracking (MOT) on HD Video Stream
-  Challenges for MOT deployment on Edge Devices
-  Data-Model-Circuit Tri-Design Framework and its Efficacy
 -  Temporal Frame Filtering & Spatial Saliency Focusing
 -  Hardware-friendly Sparsity Pattern-aware Pruning
 -  Scalable Dataflow-style FPGA-based Accelerator

Multiple Object Tracking (MOT)

- Basis for video intelligence in
 - Autonomous driving
 - Virtual reality
 - Medical imaging
- Involves object
 - Detection
 - Localization
 - Association



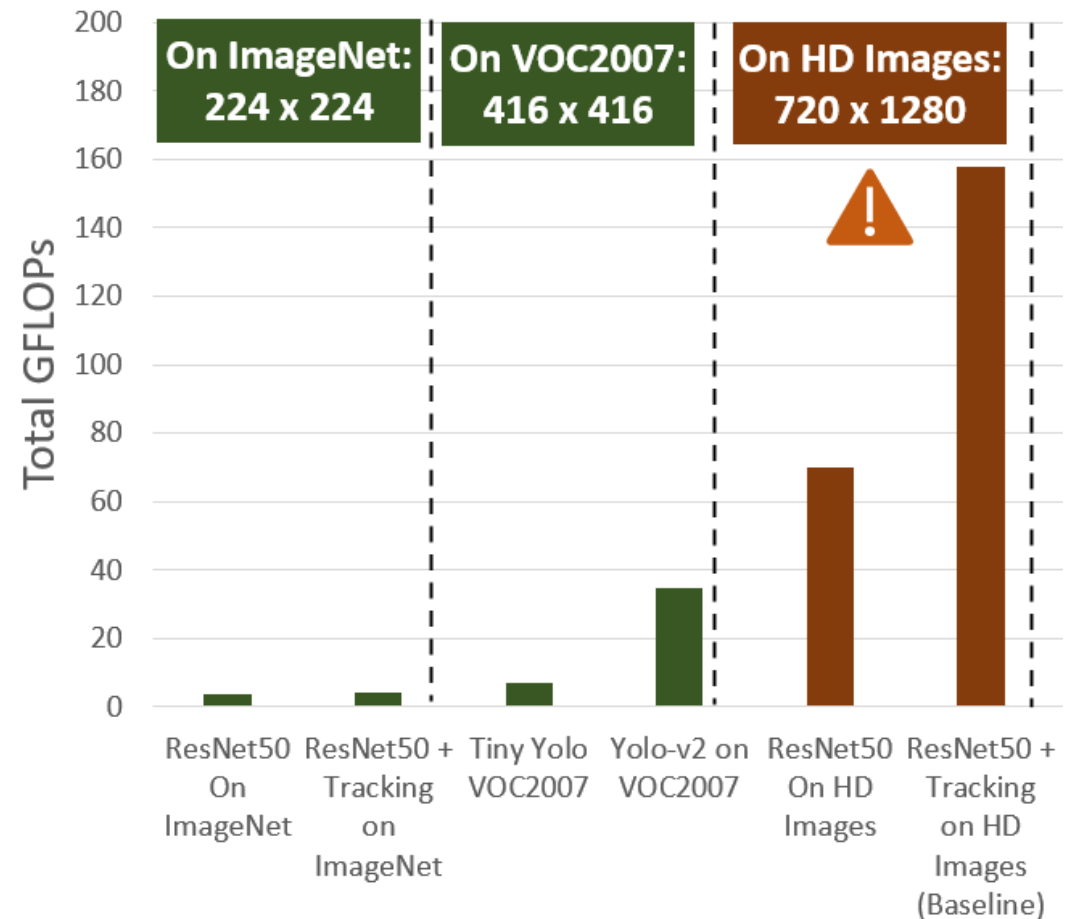
Source: [BDD100K](#) Driving Dataset Example for Object Tracking

Problem & Motivation

Accurate real-time High-Definition (HD) video processing for MOT

Critical Limitations:

- Computationally-expensive ML models
 - High in accuracy but not hardware friendly
- Massive size of video data as input
 - Further adds to computational complexity
- Poor scalability of hardware devices
 - Limited computing resources and power
 - Low degree of parallelism by design tools



Challenges & Approaches

1

MOT implementation prohibitively expensive in energy and latency due to large data size



Eliminate temporal and spatial redundancies in video frames to reduce data complexity

2

Gigantic number of parameters in state-of-the-art MOT models



Hardware-friendly model compression necessary

3

Lack of software-hardware full-stack design approach



Realistic implementation and evaluation on hardware

No prior work addresses ①-③ in a single unified MOT implementation pipeline!

Proposed Data-Model-Circuit Tri-Design Framework

State-of-the-art MOT Model: QDTrack¹ with BDD100K dataset

- Faster R-CNN with Feature Pyramid Network (FPN) backbone
- Contrastive Learning to optimize backbone parameters
- Bi-directional Softmax for object association and tracking

A synergized software/hardware co-optimization framework for MOT

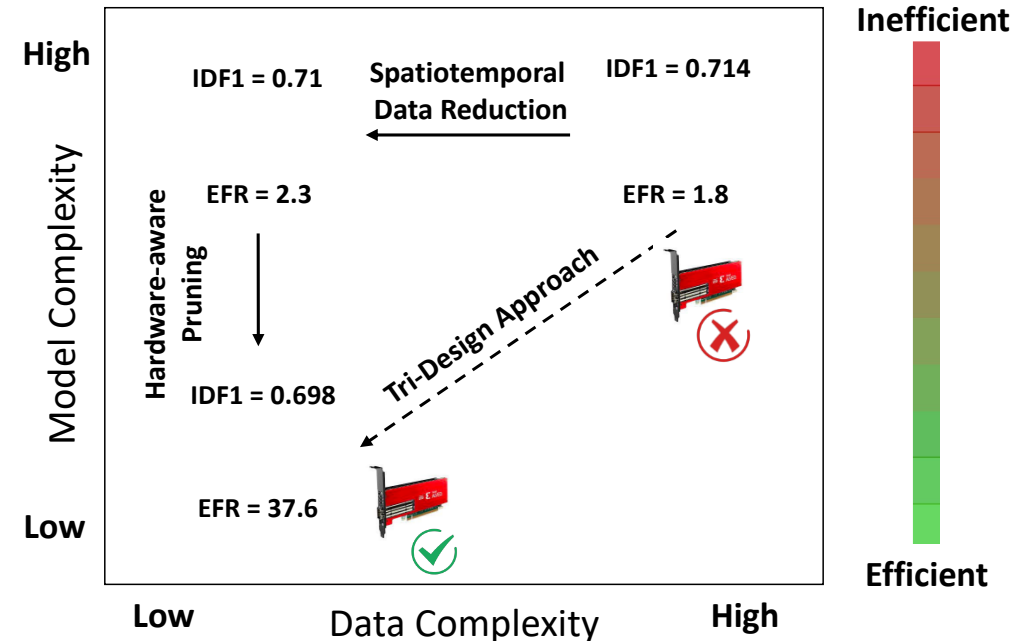
High-throughput

Low-cost

High-accuracy

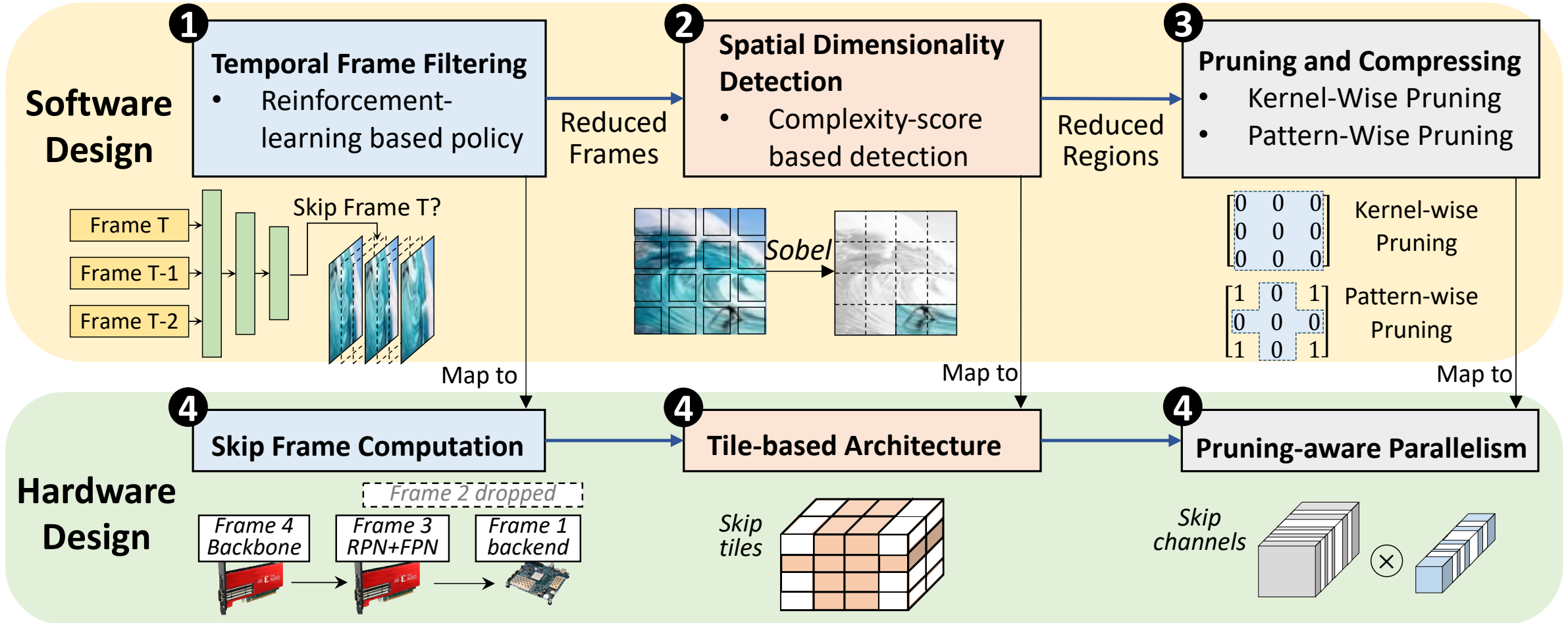
Real-time performance

High-energy efficiency



¹Pang et al., *Quasi-Dense Similarity Learning for Multiple Object Tracking*, CVPR 2021.

Data-Model-Hardware Tri-Design Overview



RL-based Model for Temporal Frame Filtering

Performance Loss

Minimize ID-swap/frame

Training Objective



Amount of Filtering

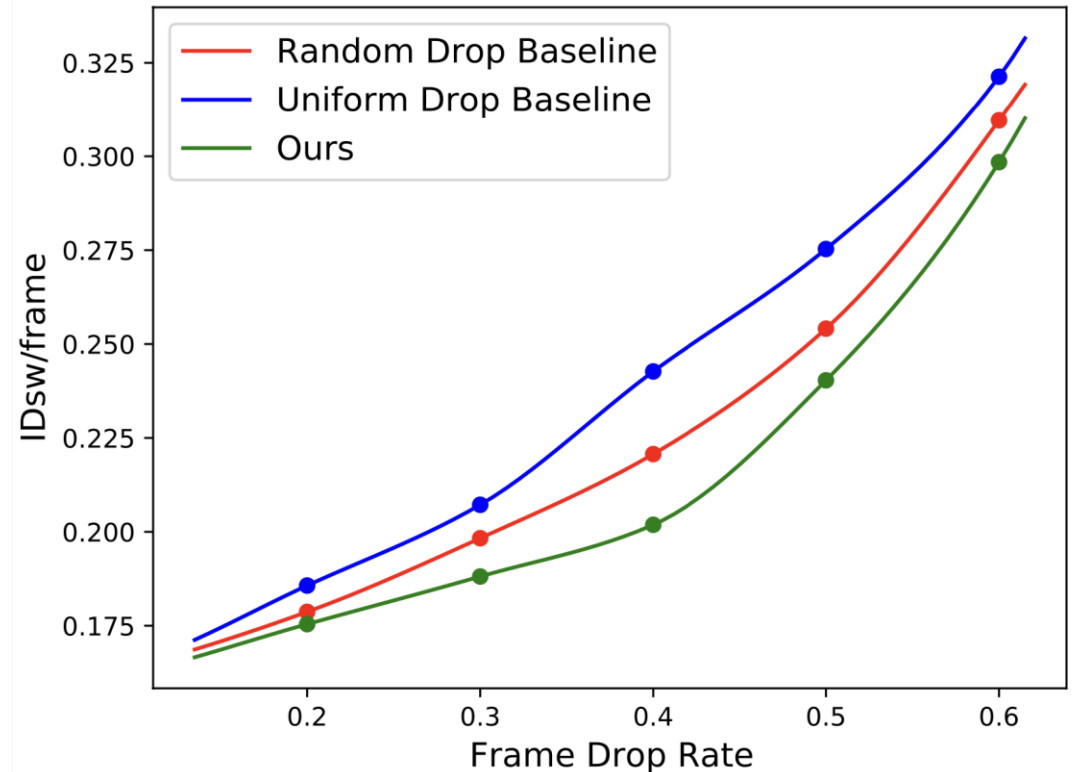
Maximize frame drop rate

Baselines

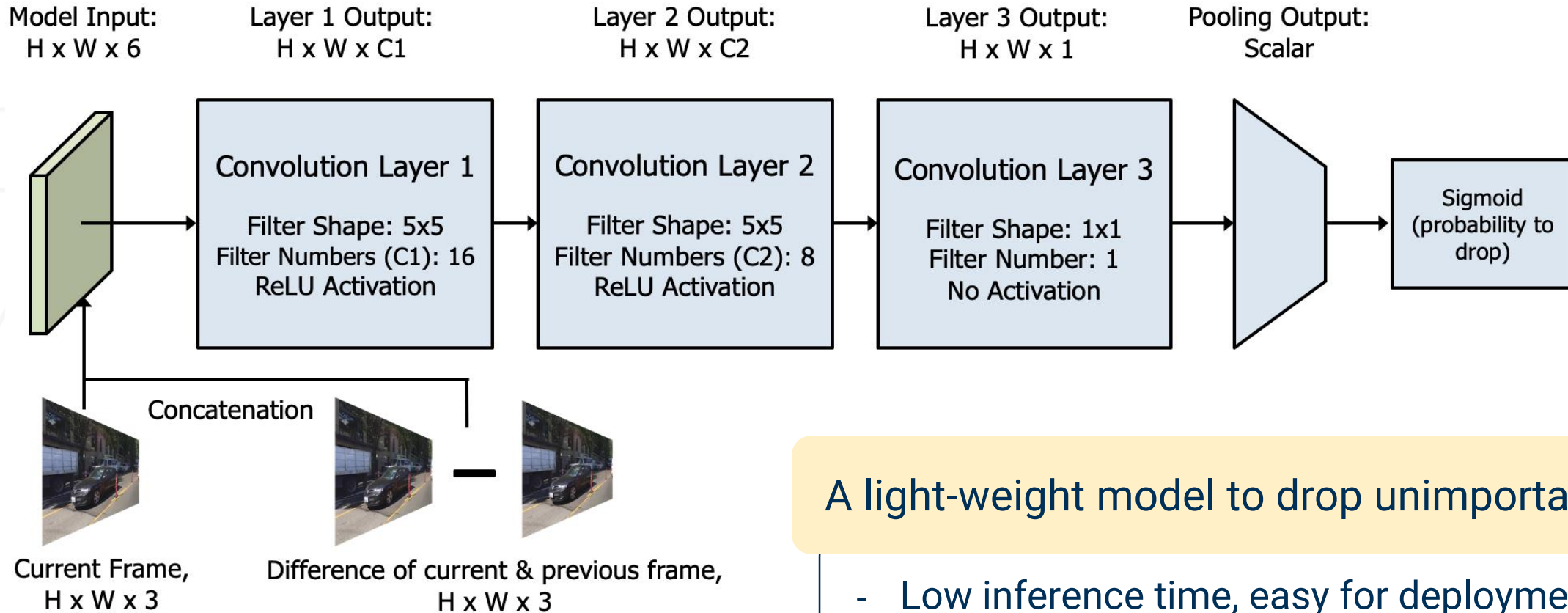
- Drop every N^{th} frame
- Drop frames randomly

Proposed

- Performs better
- “Sweet” drop rate: 40%



Frame Filtering Model Architecture



A light-weight model to drop unimportant frames

- Low inference time, easy for deployment
- High efficiency for subsequent modules

Temporal Frame Filtering Model Behavior

Before Frame Dropping



After Frame Dropping



Less changes
in surroundings



More frames
filtered

More changes
in surroundings



Less frames
filtered

Spatial Saliency Focusing

Input Frame:
 $N_1 \times H \times W \times 3$



Frame
Filtering

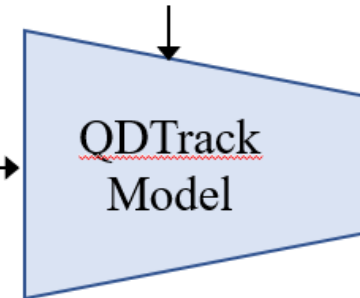
Dropped Frame:
 $N_2 \times H \times W \times 3$



Saliency
Reduction
(image)



Saliency
Reduction (feature)



Prediction



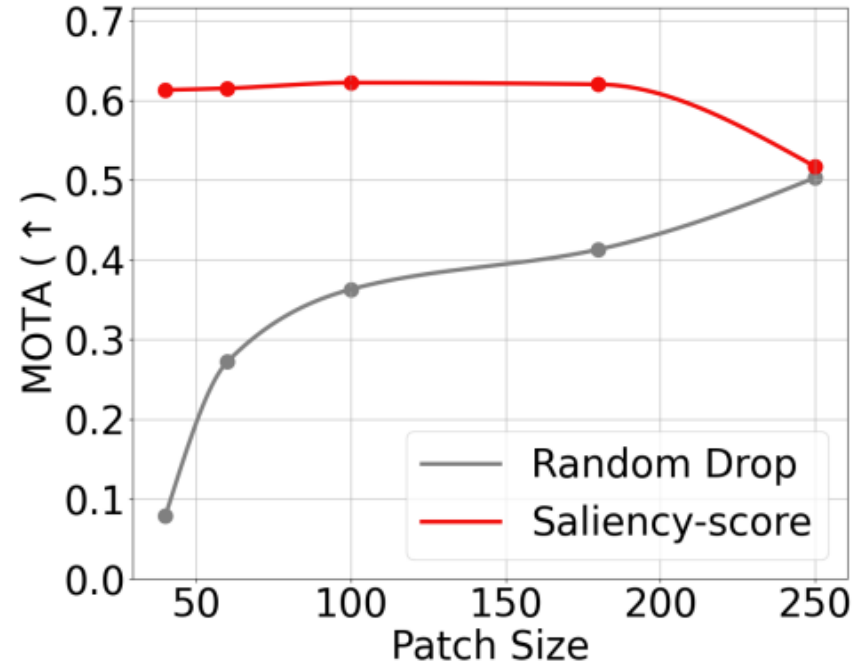
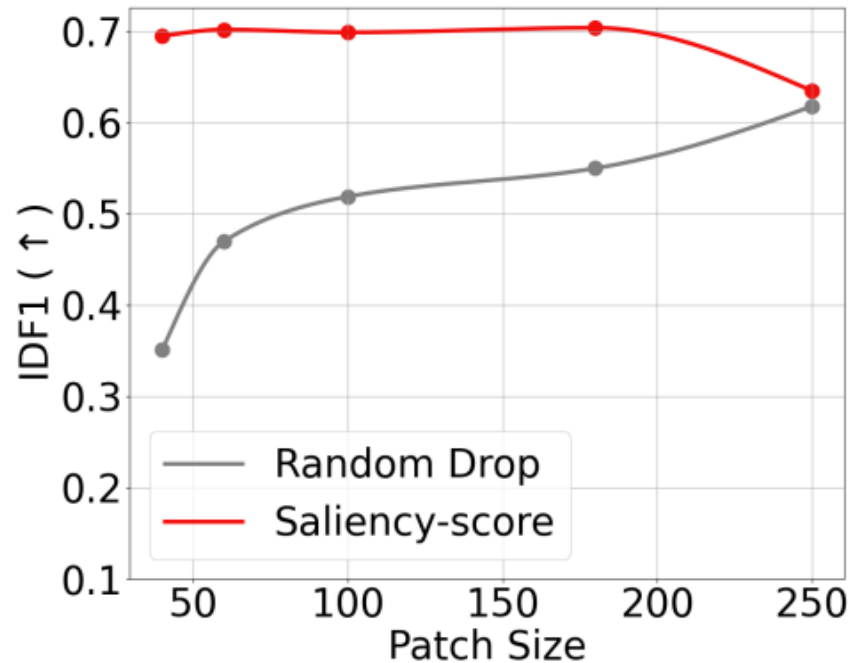
Spatial Saliency Reduction (SSR) Method

- Decompose input image into multiple patches
- Obtain saliency score using Sobel operator
- Create binary masks based on the scores
- Apply to both image and feature spaces

Highlights

- Drop unimportant pixels
- Align with visually spatial saliency, easy for computation
- Eliminate in images and features

Saliency Score and Patch Size Estimation



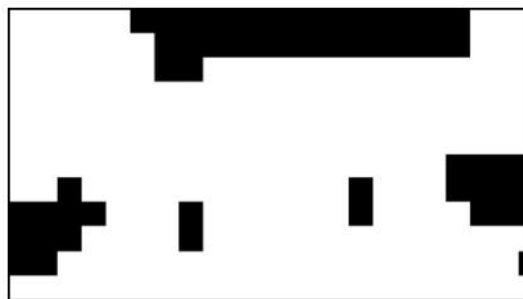
Saliency score performs better than random patch dropping

Hardware-friendly “sweet” patch size: **60 x 60**

Spatial Saliency Focusing Behavior



(a) Input frame



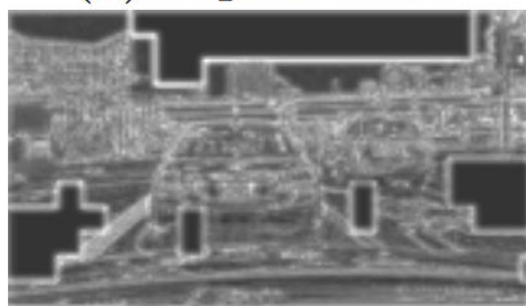
(b) Saliency Mask



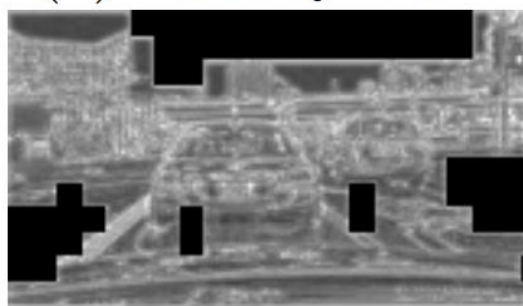
(c) Masked Input



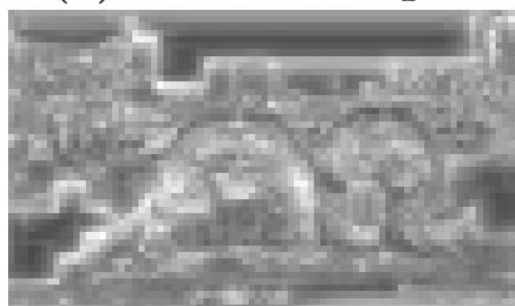
(d) GT



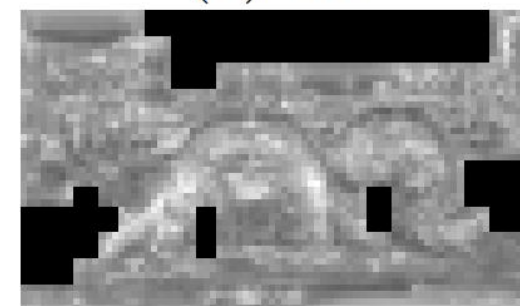
(e) FM(2,0)



(f) FM(2,0) w/ Mask



(g) FM(3,0)



(h) FM(3,0) w/ Mask

Saliency focusing
ratio set to 20%

Discarded pixels
shown by dark patches

GT -> Ground Truth
FM -> Feature Map
(i, j) -> ResNet-50 layer

Hardware-aware Model Pruning

SOTA Iterative Magnitude Pruning (IMP) applied to QDTrack

Metrics	Dense Model	Global Pruning		
	0%	80%	85%	90%
IDF1 ↑	0.714	0.712	0.706	0.703
MOTA ↑	0.637	0.631	0.627	0.624
Sparse Kernel Ratio	0%	36.59%	34.44%	32.53%

Ratio of pruned 3 x 3 kernel weights over total number of pruned weights

Kernel-wise sparsity embedded in irregular weight pruning on QDTrack!

But... IMP is unstructured 😞



Unstructured Pruning

Hinders parallelization

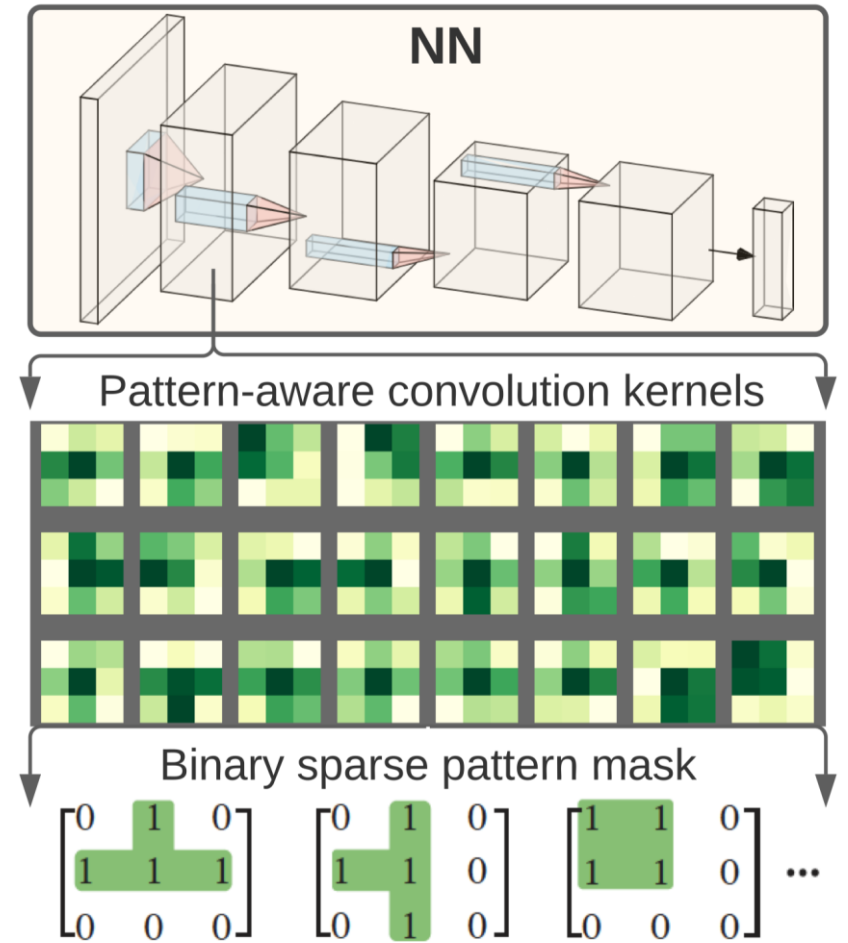


Structured Pruning

Hardware-friendly

Irregular Pattern-aware Pruning

- For uncovered sparse weights
 - From kernel-wise sparse patterns
- Effective area of a convolution kernel,
 - Maintains specific sparse patterns
 - May not yield kernels with all zero weights
- Pre-defined irregular sparse patterns^[1]
 - Leveraged for 3 x 3 kernels
- Fixed number of such sparse patterns
 - Facilitates efficient hardware implementation



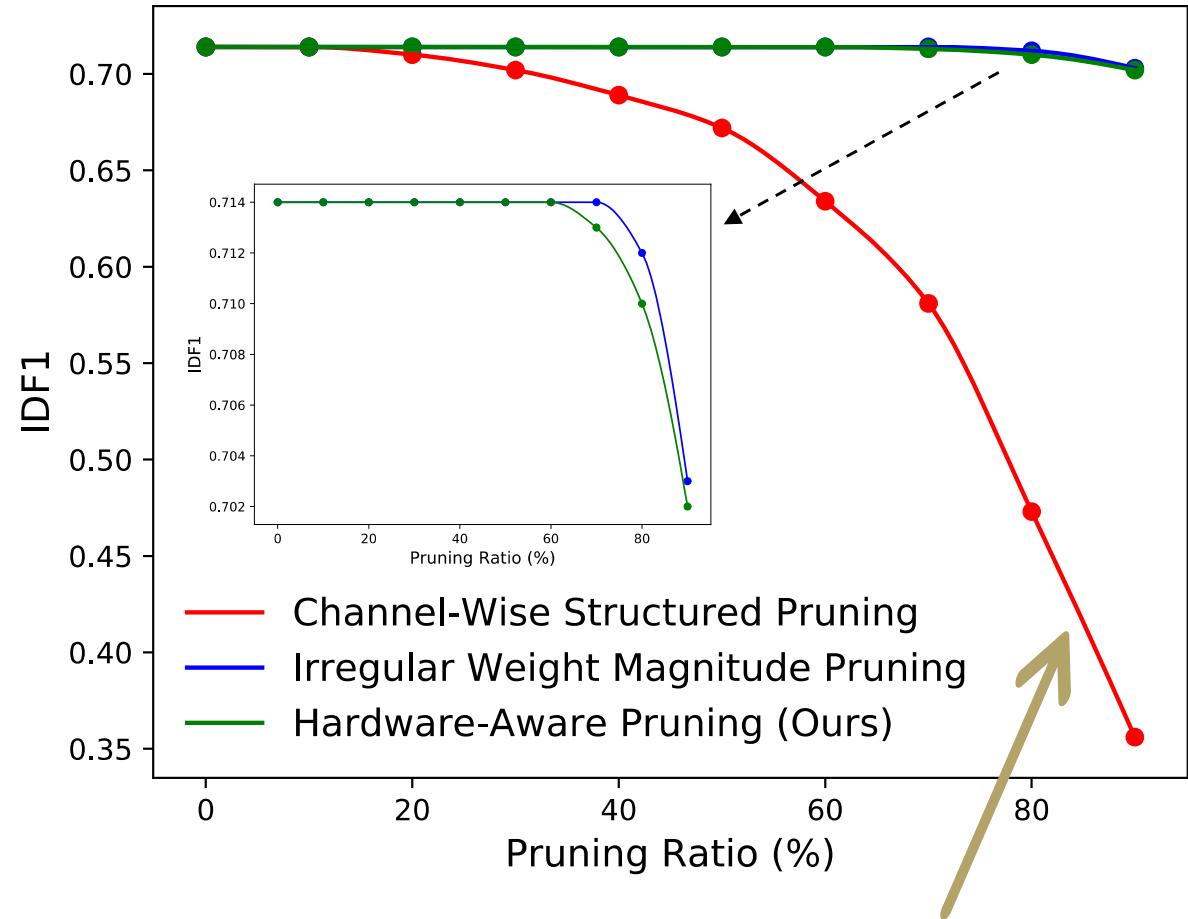
Pattern Pruning

^[1]Xiaolong Ma et al., *An image enhancing pattern-based sparsity for real-time inference on mobile devices*, ECCV 2020.

Proposed Pruning Method

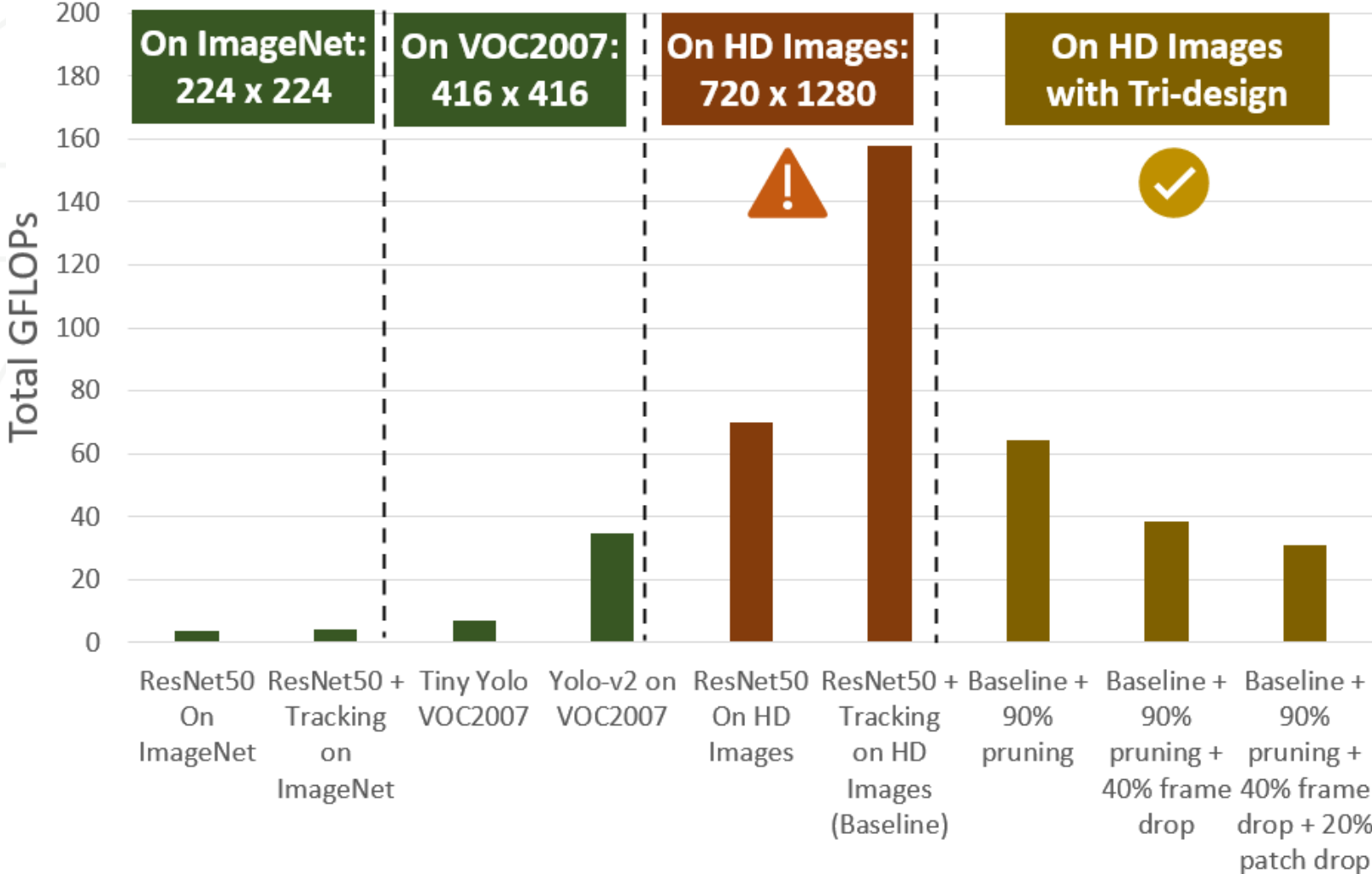
- Find **irregular** weight sparse mask \mathbf{M}_0
 - As per model weight magnitudes
- Extract **kernel-wise** sparsity mask \mathbf{M}_k from \mathbf{M}_0
- Find irregular **pattern-aware** mask \mathbf{M}_p
 - On remaining weights identified by $(1 - \mathbf{M}_k)$
- **Retrain** non-zero model weights
 - Under fixed pruning mask $(\mathbf{M}_k + \mathbf{M}_p)$

Our method achieves MOT performance like IMP across different pruning ratios!



Channel-wise pruning sensitive to pruning ratio

Computation Complexity Comparison



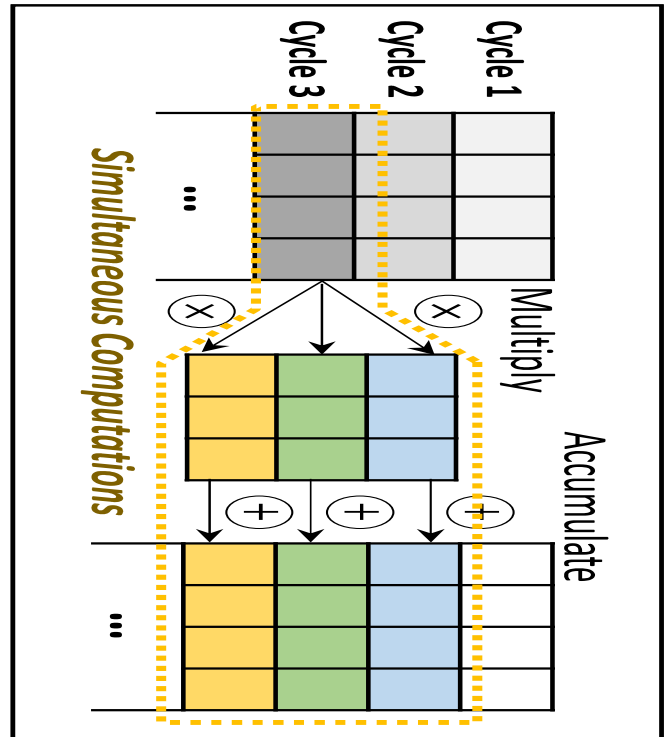
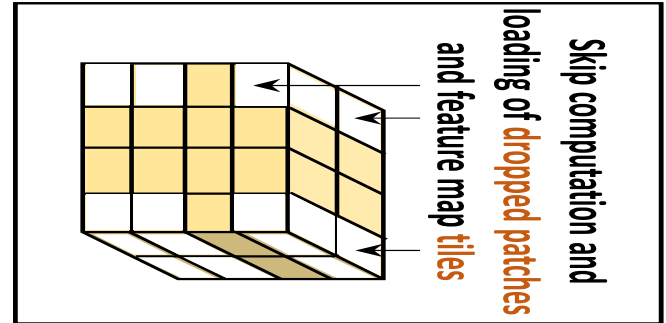
Baseline models and our tri-design approaches

Significant reduction in GOPs achieved with software optimizations

Hardware design must leverage frame filtering, saliency focusing, and pruning.

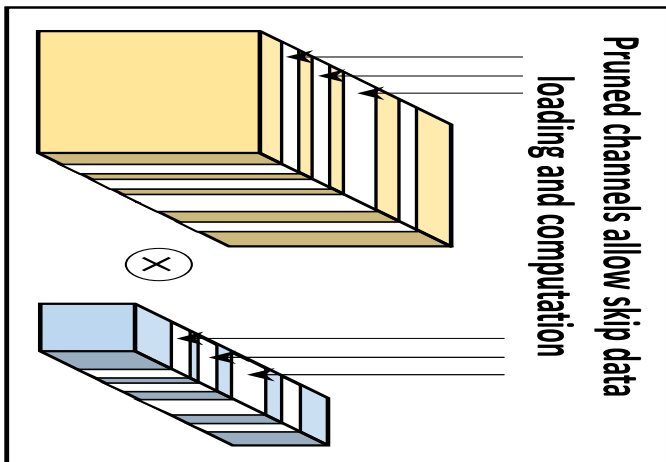
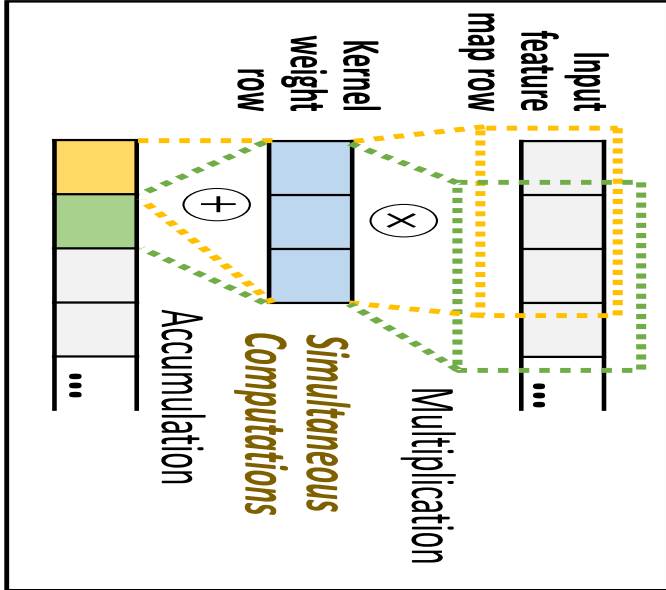
Back-end Hardware Accelerator Overview

- Feature maps partitioned
 - Limited on-chip memory
 - Tile size same as patch
- Parallel computations
 - Unrolling each row
- Overlapping operations
 - Effectively single-cycle
- Pruning-aware design
 - Skip pruned channels



(a) Tile-based Architecture

(b) Row-level Parallelism



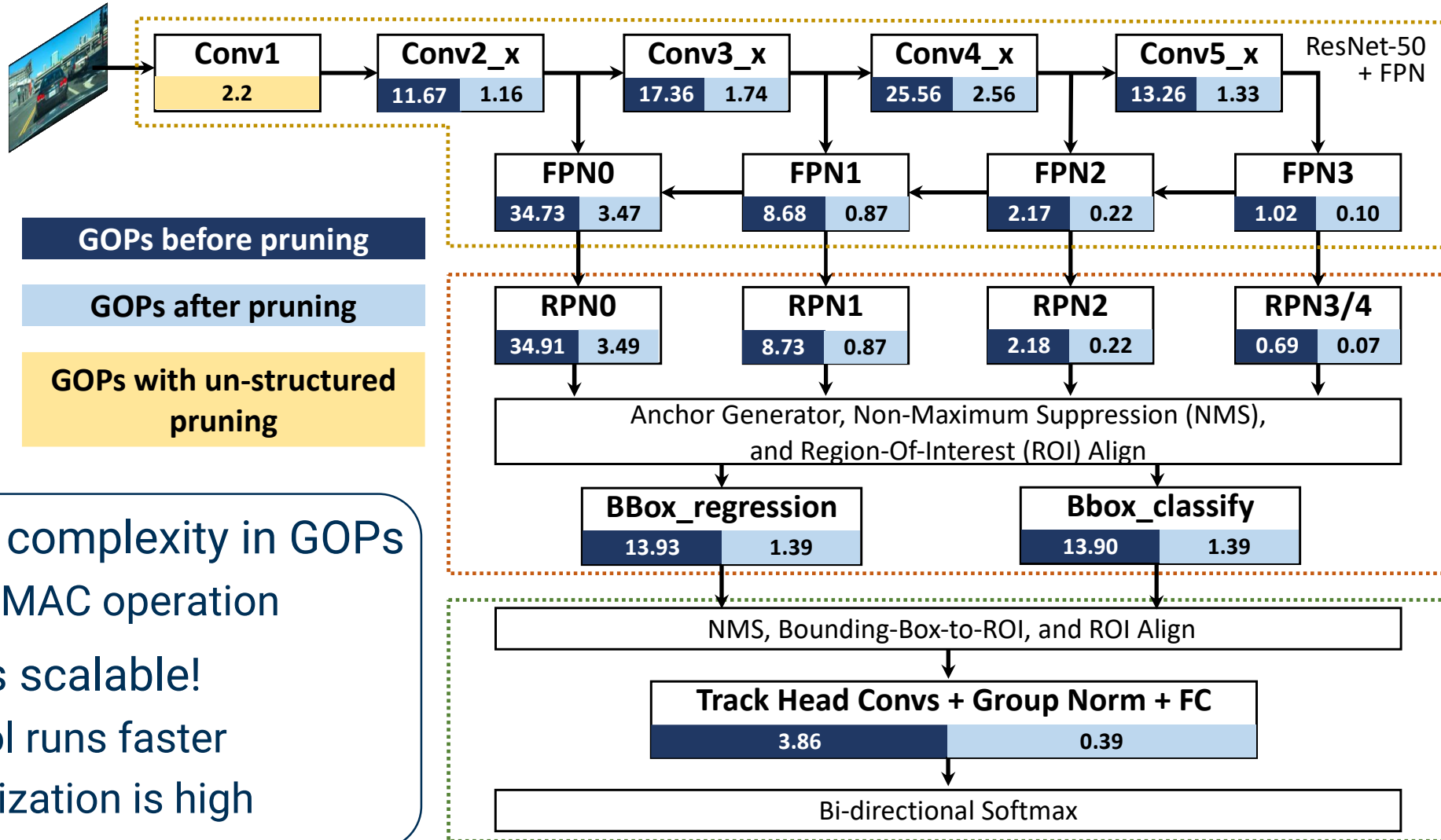
(c) Column-level Parallelism

(d) Structured-pruning

Scalable Multi-FPGA Dataflow Architecture

QDTrack Deployment on FPGA Cluster

- Computation complexity in GOPs
 - 1 GOP = 1 MAC operation
- Multi-FPGA is scalable!
 - Design tool runs faster
 - Device utilization is high



FPGA 1
Xilinx Alveo
U50@75W



FPGA 2
Xilinx Alveo
U50@75W



FPGA 3
Xilinx
ZCU104@5W



Performance Comparison

Methods	Metrics					
	IDF1 ↑	MOTA ↑	Latency ↓ (ms)	EFR ↑ (fps)	Power ↓ (W)	Energy Efficiency ↓
QDTrack (GPU Baseline)	0.714	0.637	60.9	22.5	296	13.2 J/frame
QDTrack on FPGA	0.714	0.637	554.7	1.8	50.8	28.2 J/frame
Tri-design (proposed)	0.704	0.617	44.4	37.6	50.8	1.35 J/frame

- Standard MOT metrics: IDF1 score and Multi-Object Tracking Accuracy (MOTA).
- IDF1 emphasizes association accuracy while MOTA concerns with object detection accuracy.
- EFR (Effective Frame Rate) is indicative of throughput in processing video frames

Tri-Design Summary w.r.t. State-of-the-Art Baseline

12.5x

Latency Reduction

20.9x

Effective Frame Rate (EFR)

Similar Accuracy!

5.8x

Lower power

9.8x

Better Energy Efficiency

Summary

- Highly-efficient video processing algorithm/hardware pipeline
 - DNN-based state-of-the-art MOT model
 - BDD100K dataset contents as inputs
 - FPGA cluster with Alveo U50 and ZCU104
- Data-model-circuit tri-design for MOT implementation on edge
 - Aggressive data reduction techniques
 - Hardware-friendly model compression
 - SW optimization-aware dataflow accelerator

Thank You!

Akshay Kamath

akshay.k2@gatech.edu

Dr. Callie Hao

callie.hao@gatech.edu