Lecture 13:

Optimizations for Processing Video

Visual Computing Systems
Stanford CS348V, Winter 2018

Follow up from last time

- Recall "fancy instruction" approach to DNN acceleration

- Idea: reduce overhead of instruction stream processing, register access, and control by designing a single instruction that performs a significant amount of work

Recall Volta GPU

Single instruction to perform 2x4x4x4 + 4x4 ops

Each SM core has:

64 fp32 ALUs (mul-add)

32 fp64 ALUs

8 "tensor cores"

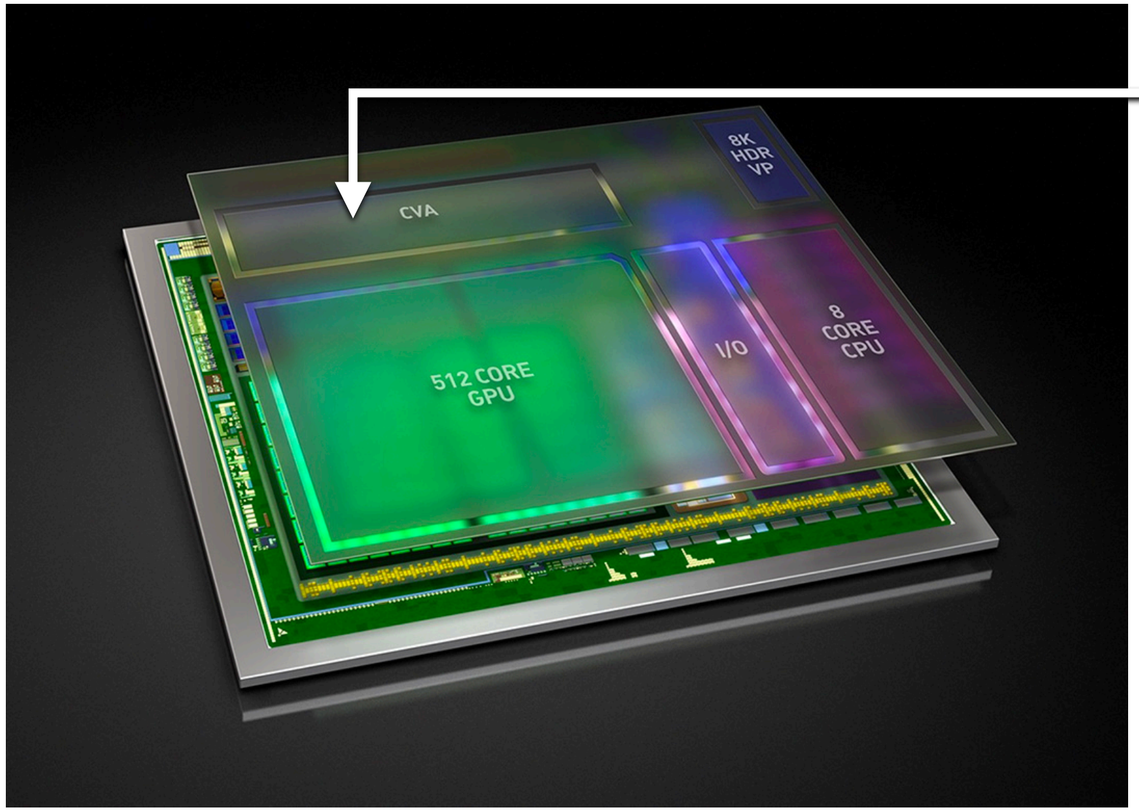
Execute 4x4 matrix mul-add instr

A x B + C for 4x4 matrices A,B,C

A, B stored as fp16, accumulation with fp32 C

Efficiency estimates *

- Estimated overhead of programmability (instruction stream, control, etc.)
  - Half-precision FMA (fused multiply-add) 2000%
  - Half-precision DP4 (vec4 dot product) 500%
  - Half-precision MMA (matrix-matrix multiply + accumulate) 27%



NVIDIA Xavier (SoC for automotive domain)

Features a Computer Vision Accelerator (CVA), a custom module for deep learning acceleration (large matrix multiply unit)

But only 2x more efficient than Volta MMA instruction despite being highly specialized component. (includes optimization of gating multipliers if either operand is zero)

* Estimates by Bill Dally using academic numbers, SysML talk, Feb 2018

The visual data world in 2030

8.5 billion people (61% urban)
[UN estimate]

70% smartphone penetration
[Statista, 50% in 2020]

25% turned on → **1.5B**

---

2 billion cars
[Sperling 2009]

8 cameras/car

25% load → **4B**
[currently 2%]

1.1B streaming security cameras
Extrapolation from 245M in 2014, 10% annual growth [IHS]

Assume 8K video resolution (33 megapixel)

Total capture capability across the world

~6.5B video streams = 2.1 x 10^{17} pixels x 30 fps = 6 exapixels/sec

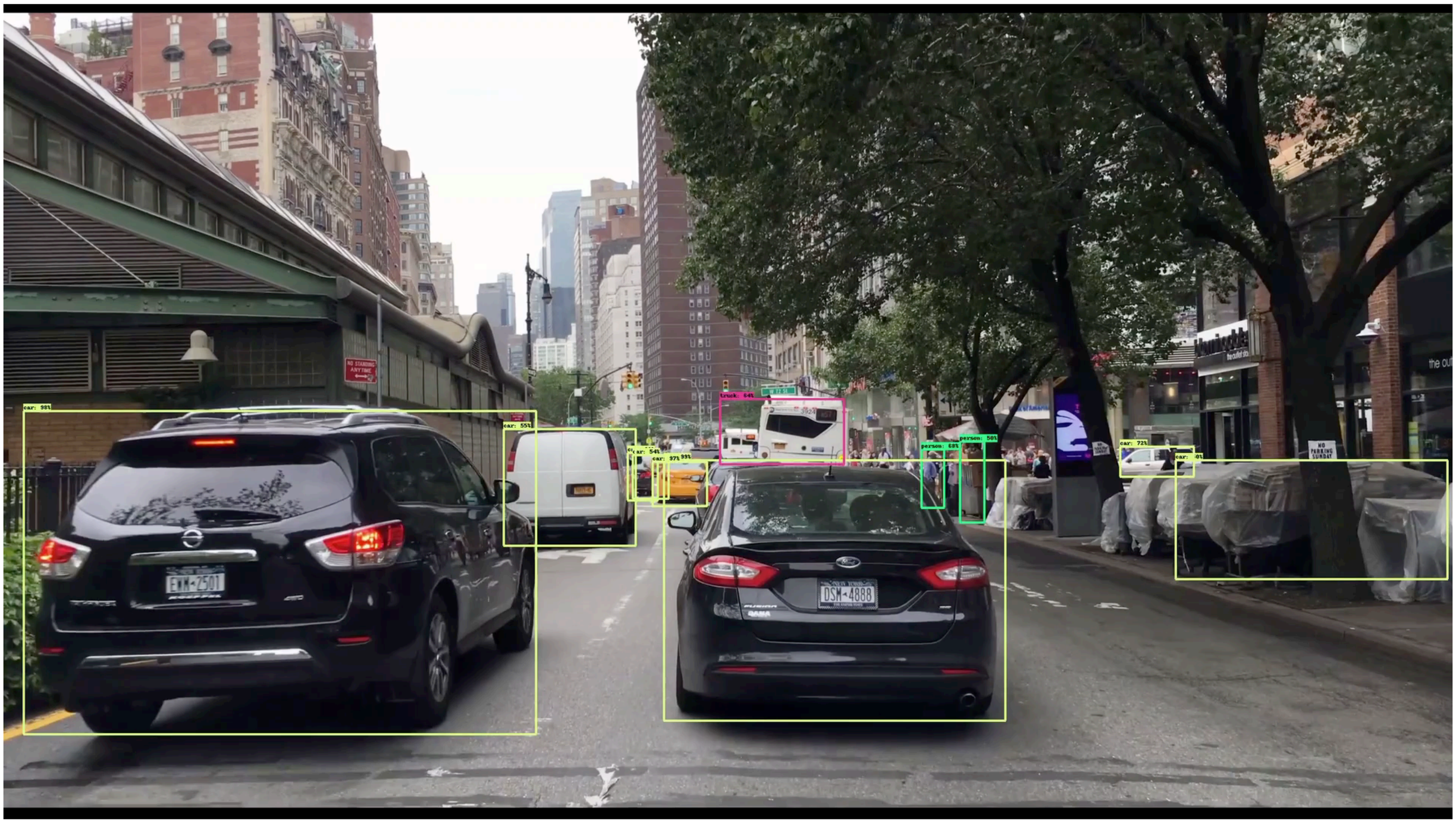
Other considerations: home health care robotics, survey science, infrastructure monitoring…

Thought experiment

- Imagine we wanted to detect people/cars/bikes in a video stream

Thought experiment

- Imagine we wanted to detect people/cars/bikes in a video stream



Object detection performance

600x600 input images

COCO-trained models {#coco-models}

Model name	Speed (ms)	COCO mAP[^1]	Outputs
ssd_mobilenet_v1_coco	30	21	Boxes
ssd_inception_v2_coco	42	24	Boxes
faster_rcnn_inception_v2_coco	58	28	Boxes
faster_rcnn_resnet50_coco	89	30	Boxes
faster_rcnn_resnet50_lowproposals_coco	64		Boxes
rfcn_resnet101_coco	92	30	Boxes
faster_rcnn_resnet101_coco	106	32	Boxes
faster_rcnn_resnet101_lowproposals_coco	82		Boxes
faster_rcnn_inception_resnet_v2_atrous_coco	620	37	Boxes
faster_rcnn_inception_resnet_v2_atrous_lowproposals_coco	241		Boxes
faster_rcnn_nas	1833	43	Boxes
faster_rcnn_nas_lowproposals_coco	540		Boxes

[Credit: Tensorflow detection model zoo]

Exploiting coherence in video for efficiency

- Benefits in datacenter:
  - Lower cost/frame enables processing of more streams (e.g., thousands of webcams)

- Benefits on the edge:
  - Cheaper per frame costs, real-time performance on cheaper/lower energy computing hardware
  - Lower latency per frame
    - Example: automated breaking systems target ~40ms sense to brake

Temporal differencing

- **Use background image label if similar to background image**

(a) empty frame **(b)** frame with a car **(c)** subtracted frames

- **Use same result as previous frame if sufficiently similar**
  - **How to define sufficiently similar? (thresholds)?**
  - **Differences in feature space more robust than over pixels**

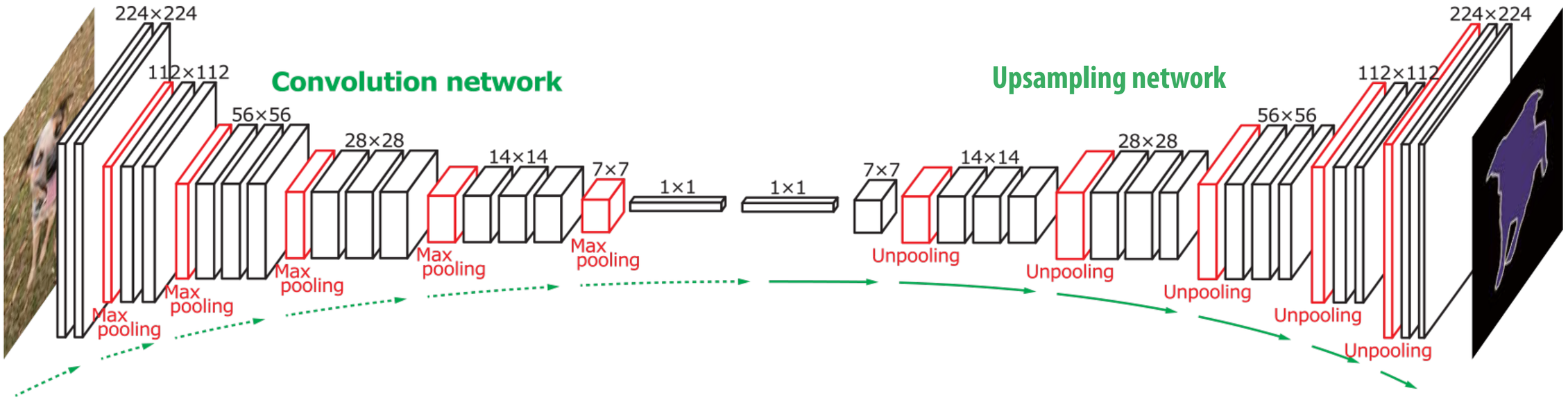
Tracking

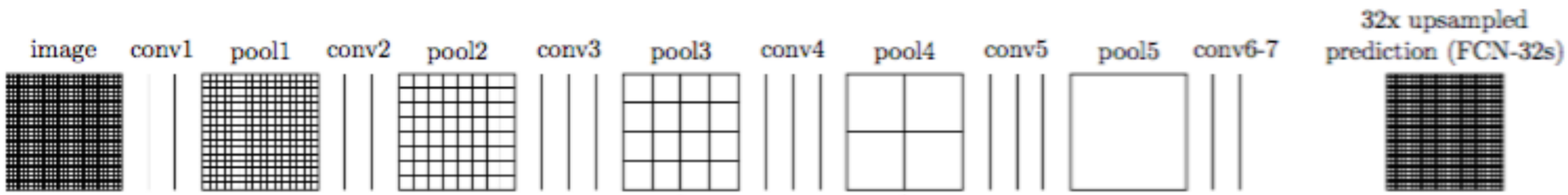
- Evaluate expensive detector sparsely in time (e.g., every 1/2 second), then use more efficient tracking algorithm to update annotations over sequence of frames

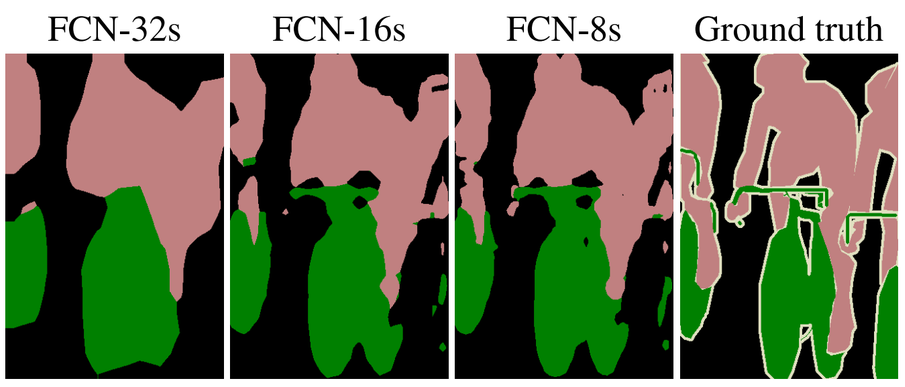
Tracking

- Evaluate expensive detector sparsely in time (e.g., every 1/2 second), then use more efficient tracking algorithm to update annotations over sequence of frames

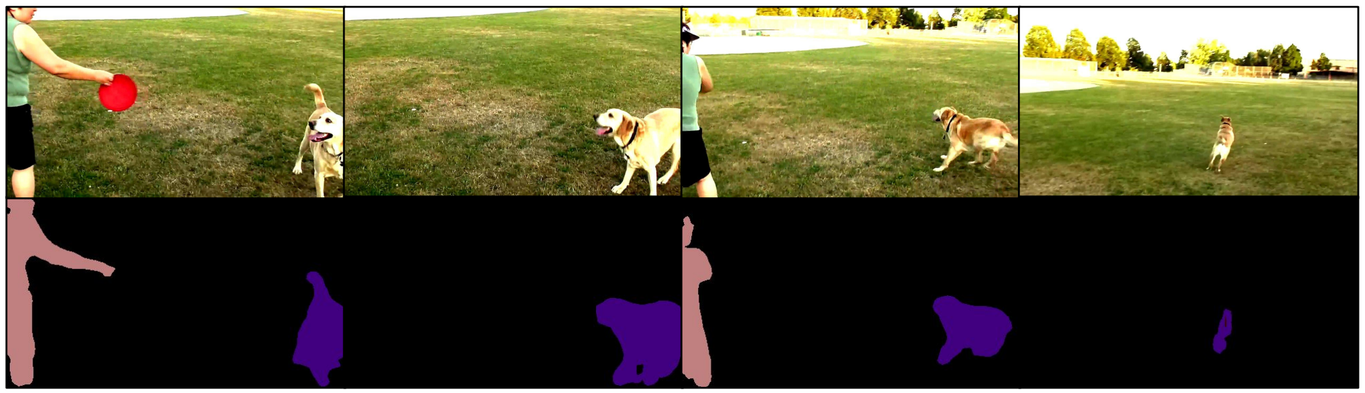
A fully convolutional network for image segmentation

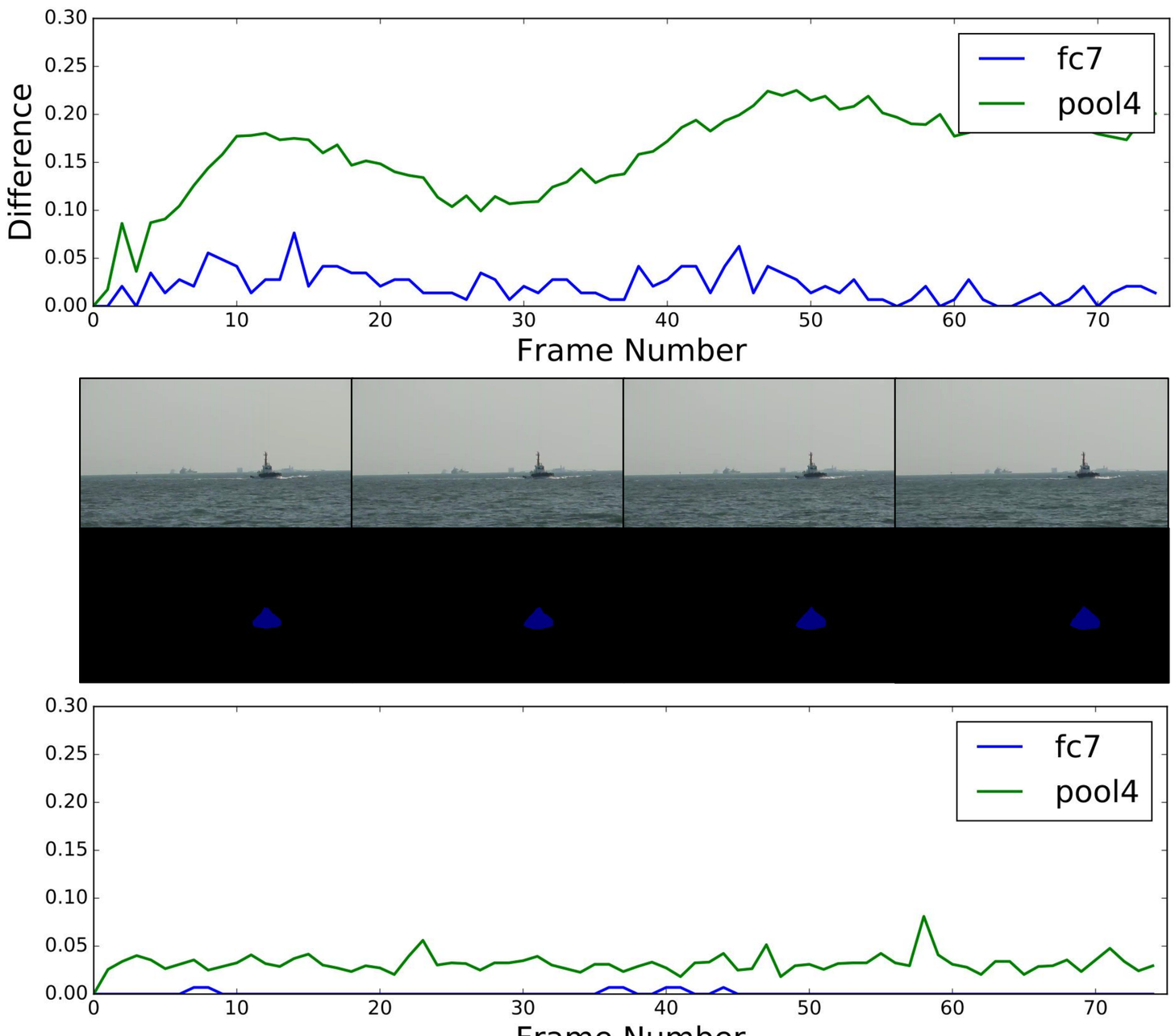






Temporal stability of deep(er) features

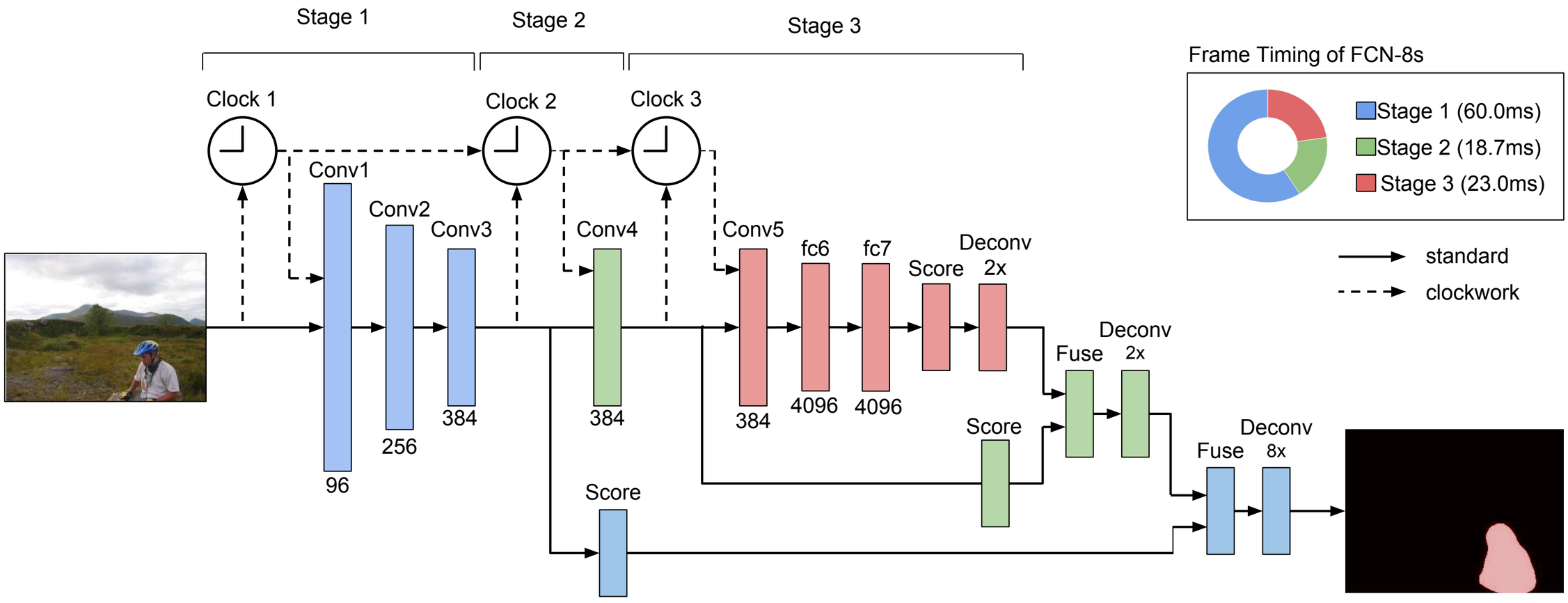




Observation:
Deeper features feature more temporal stability

(more semantic information changes less rapidly in a scene)

Clockwork network: reuse deeper layer outputs in subsequent frames

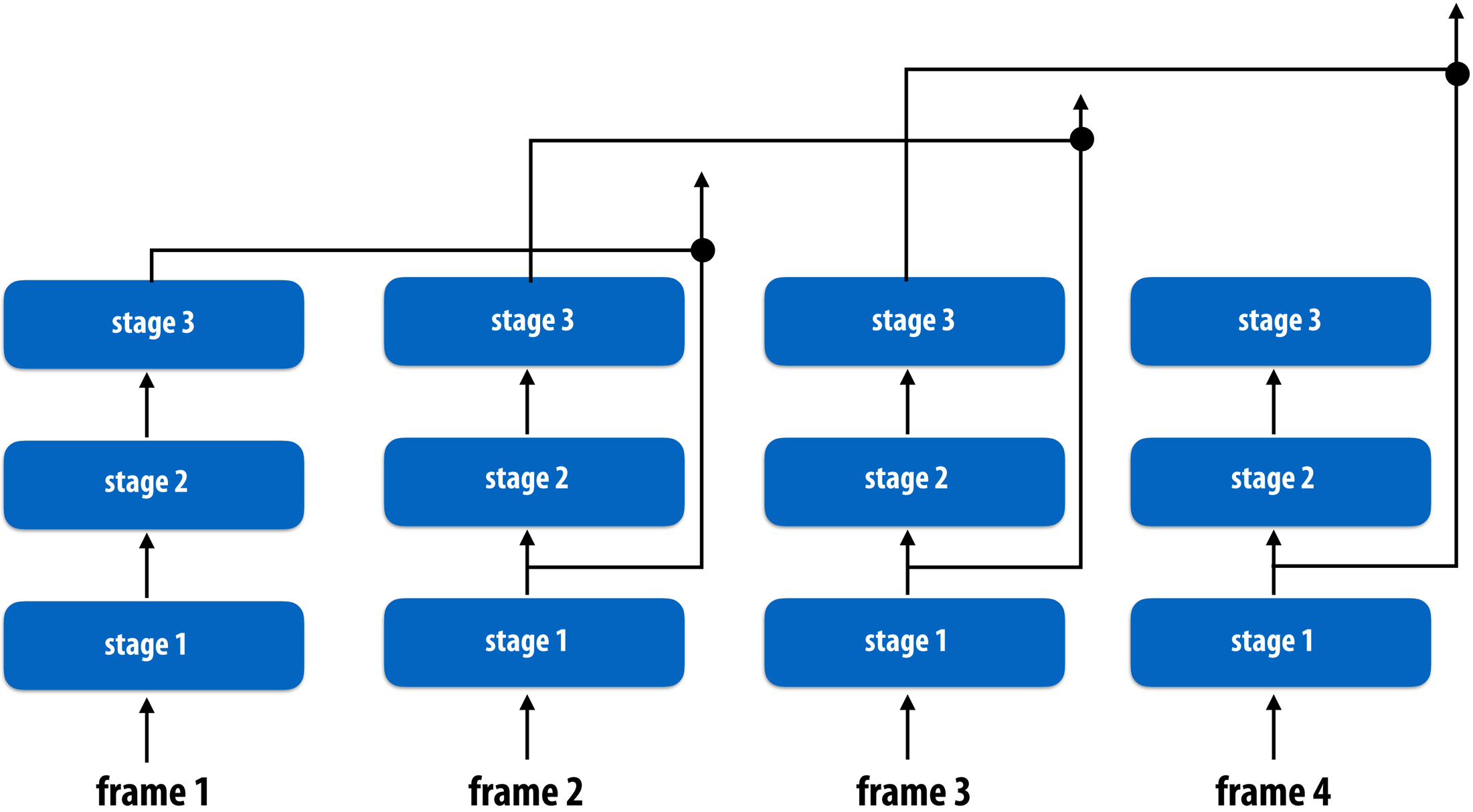


Evaluate lower (early) layers each frame

Optionally combine (fresh) output of lower layers with output of higher layers from previous frames.

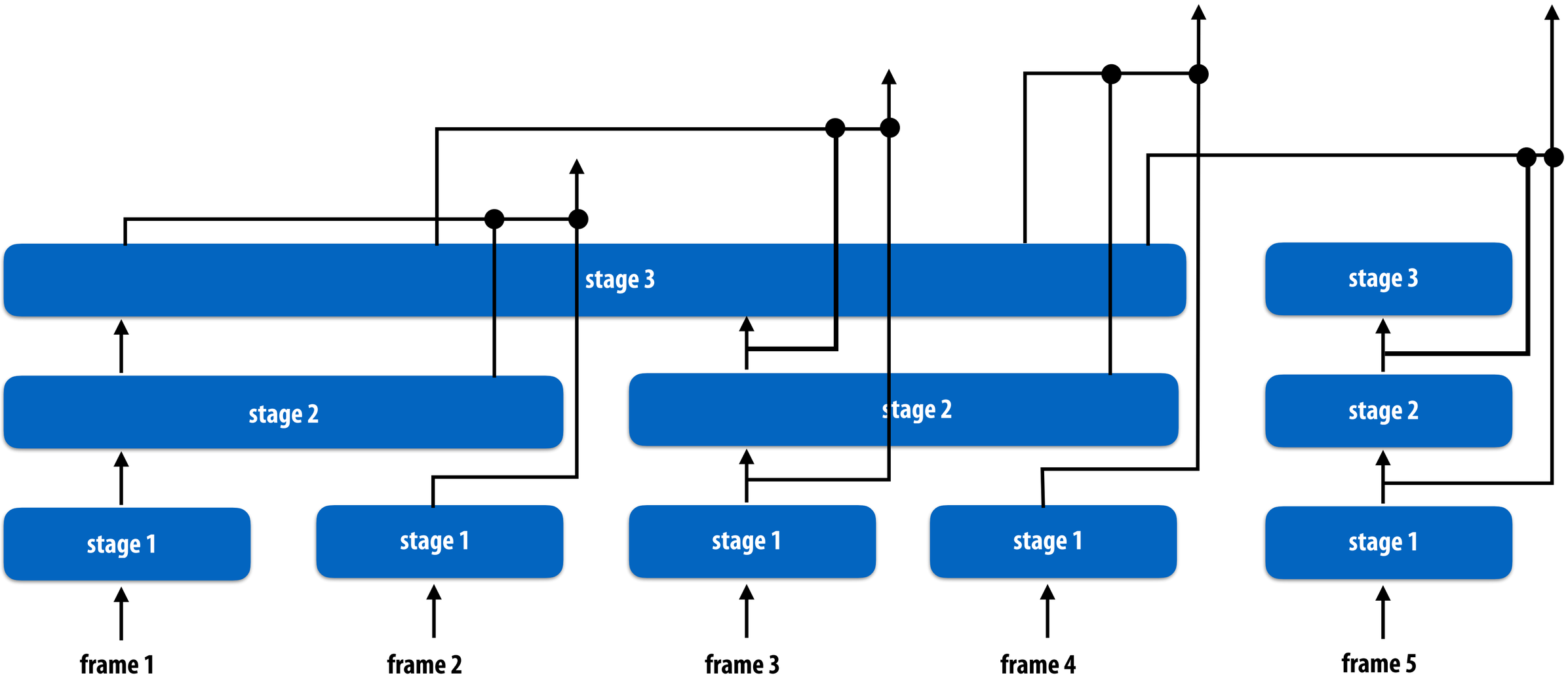
Clockwork convnet

- Reduced latency: generate output only after evaluating first layer

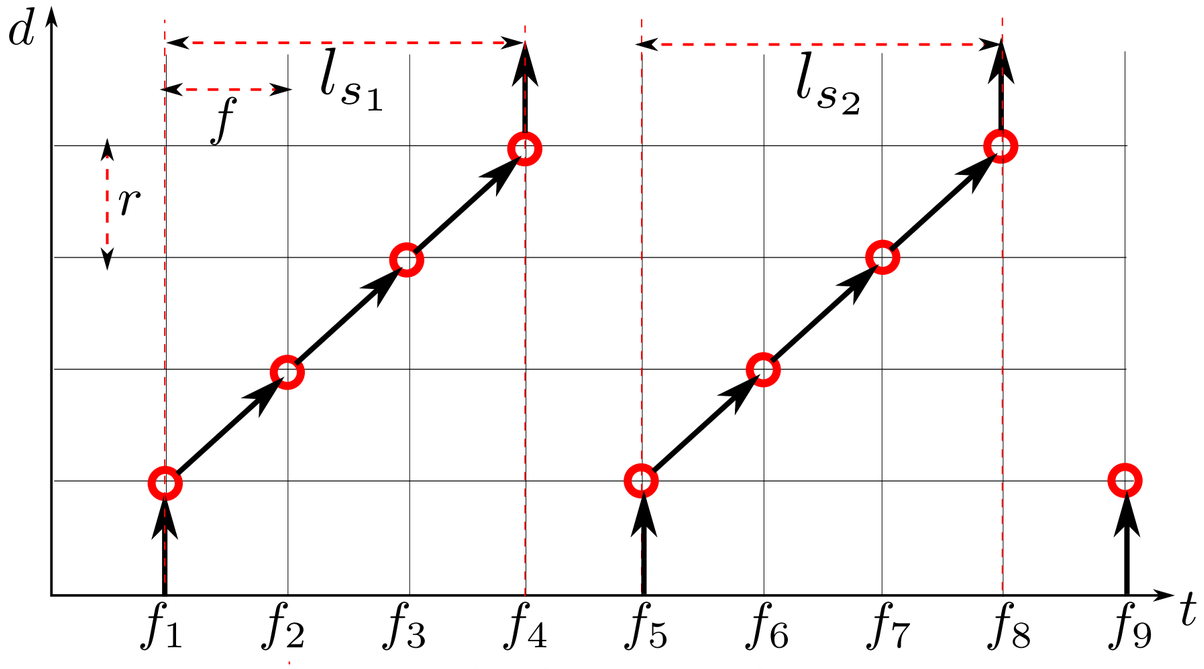


Clockwork convnet

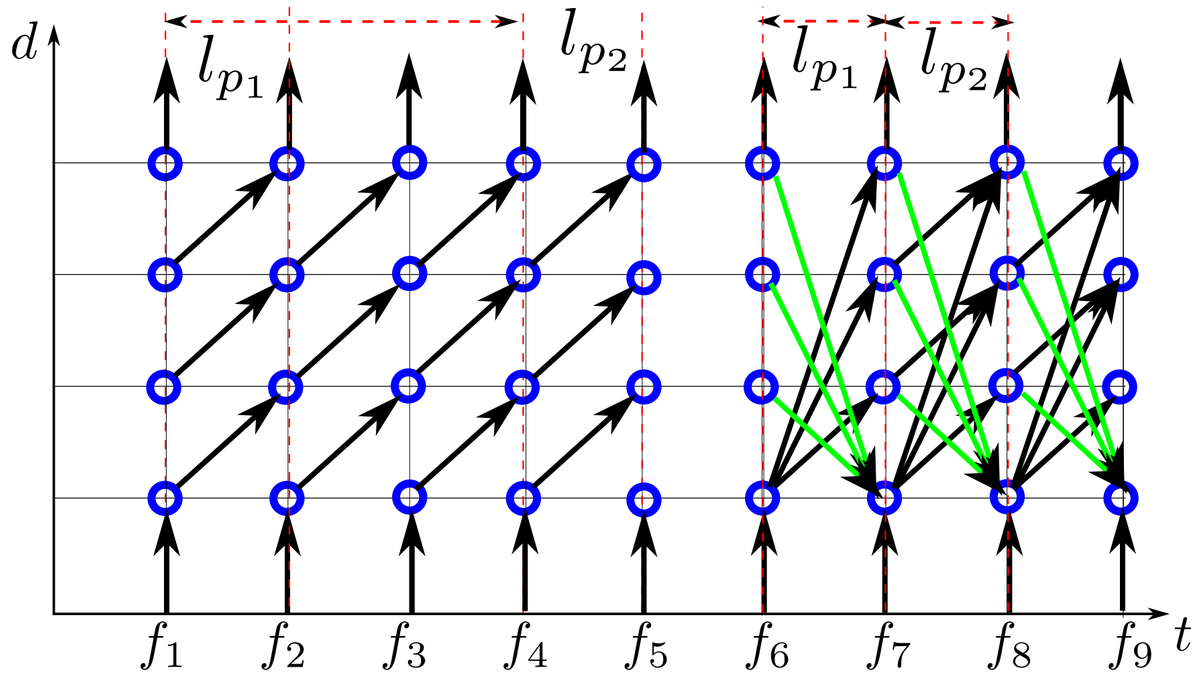
- Increase throughput: update higher layers at a lower rate



Another example: parallel video networks



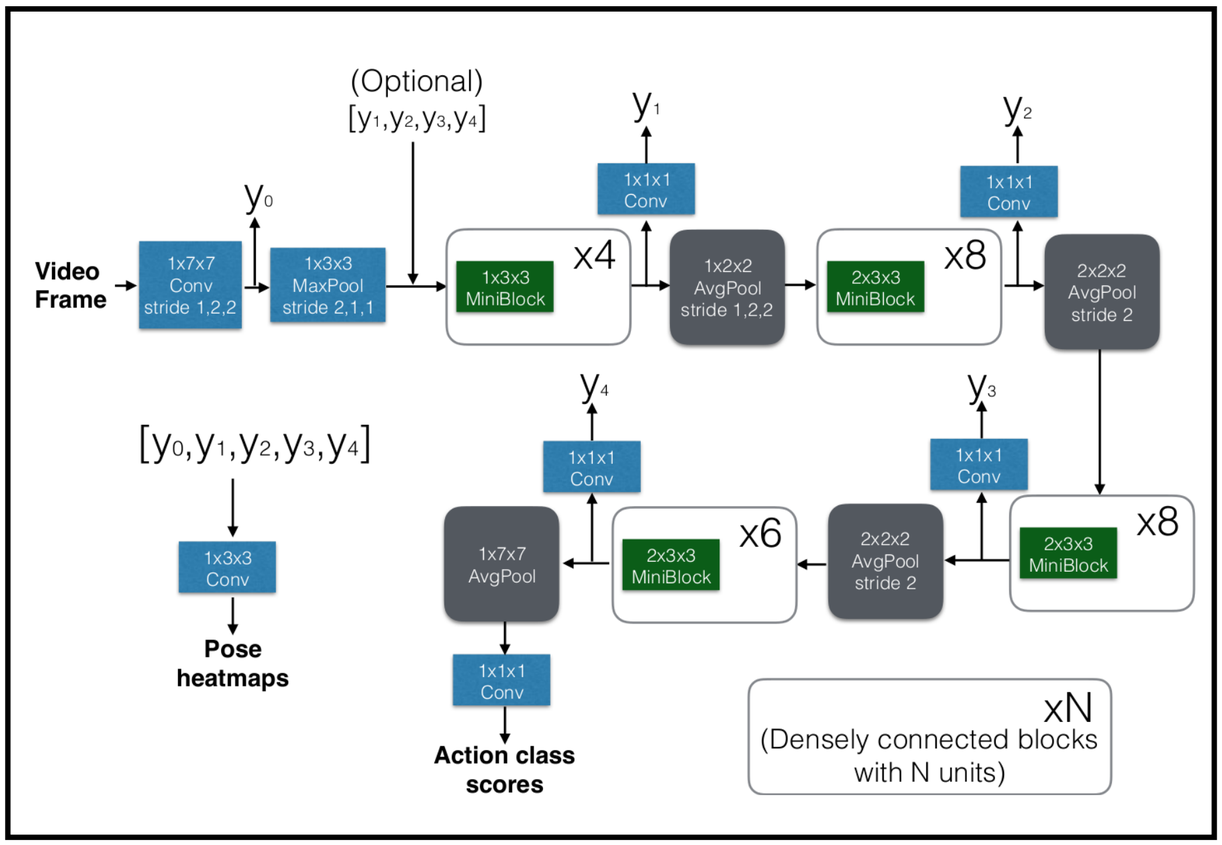
Non-pipelined execution



No skips (increases parallelism)

With skips (reduces latency)

Pipelined execution



[Carriera et al. 2018]

Model specialization

- Common principle in DNN design/training is to learn most general model (via large datasets, regularization, etc.) to perform all instances of a task

- But many cameras see a very specific distribution of images
  - Only certain types of object classes
  - Always from the same/similar viewpoint
  - Objects appear in same regions of screen

- Specialization has been a major theme in this class w.r.t hardware design. Now we wish to specialize models for the video stream

Example: model simplification

Specialized Model (specialized to scene, camera viewpoint, and window of time)

15 ms/frame

Expensive Model (Mask R-CNN)

250 ms/frame *

*** Mask R-CNN is performing instance segmentation, specialized model is only performing segmentation (mask r-CNN is performing a slightly more complex task)**

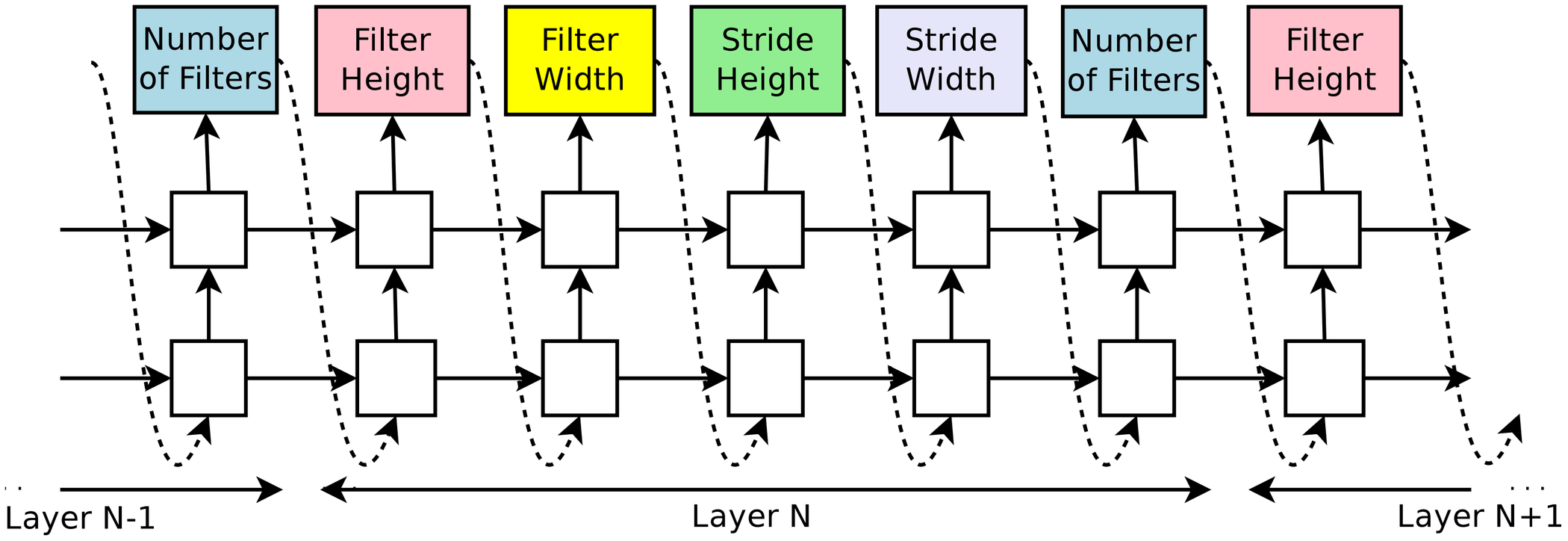
Modern trend: automating search

- Automating search for most efficient topologies
- Basic idea of simple methods: templatize network by hyperparameters, conduct (brute force) search over possibilities
    - Recall MobileNet parameters:
        - Network depth
        - Number of filters per layer
        - Input image resolution
- Many more intelligent forms of search:
    - e.g., guided by reinforcement learning

Neural architecture search via reinforcement learning

[Le 2017]

- DNN is expressed as a string of hyperparameters:



- RNN generates hyperparameters
- Specified network is trained
- Accuracy on validation set is reward
- RNN weights updated

Noscope

[Kang 17]

- Idea: given an expensive network that performance a specified detection* task, the system should automatically create a highly optimized implementation of a video stream

- Binary classification (present/not present) task on a single class, in an traffic camera video stream

(a) empty frame **(b)** frame with a car **(c)** subtracted frames

*** Noscope actually performs a simpler classification task on a precropped region of the viewport (not detection, which involves object location)**

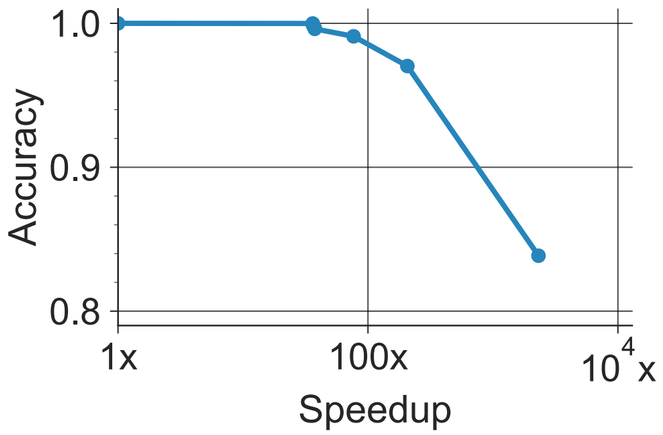
Noscope optimizations

- Model specialization for model search
  - Teacher network: Yolo object detection network
  - Student network: compact specialized network (2-4 conv layers)
  - Student "learns" to mimic the teacher

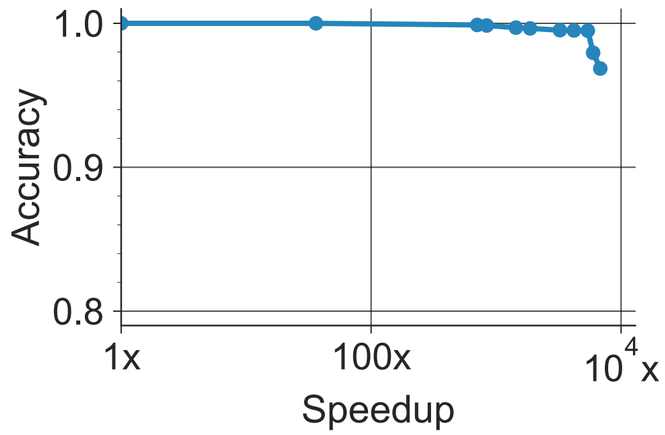
- Difference detectors with learned thresholds
  - "Same as background", "same as previous frame"
  - Learn thresholds for how often to check for differences (in frames), and what the magnitude of a meaningful difference is

- Cascades
  - Run cheap specialized model (student) first, then run teacher model if student does not make a confident prediction

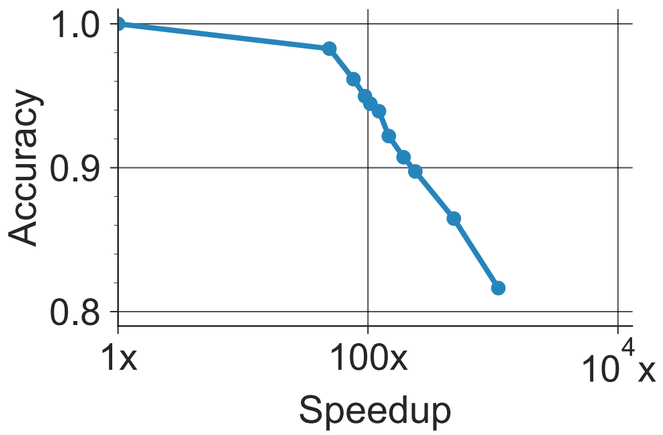
Noscope results *



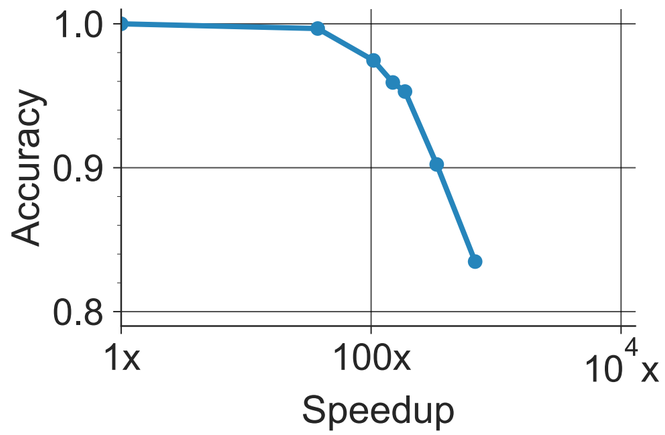
(a) taipei



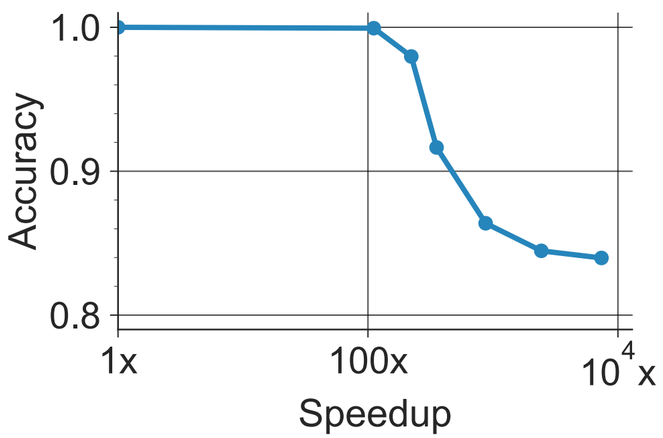
(b) coral



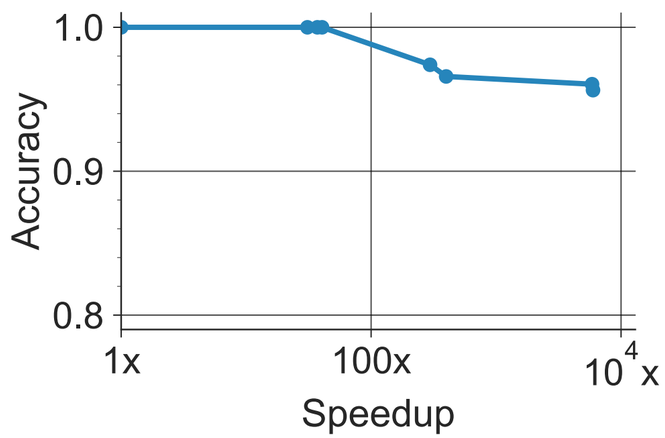
(c) amsterdam



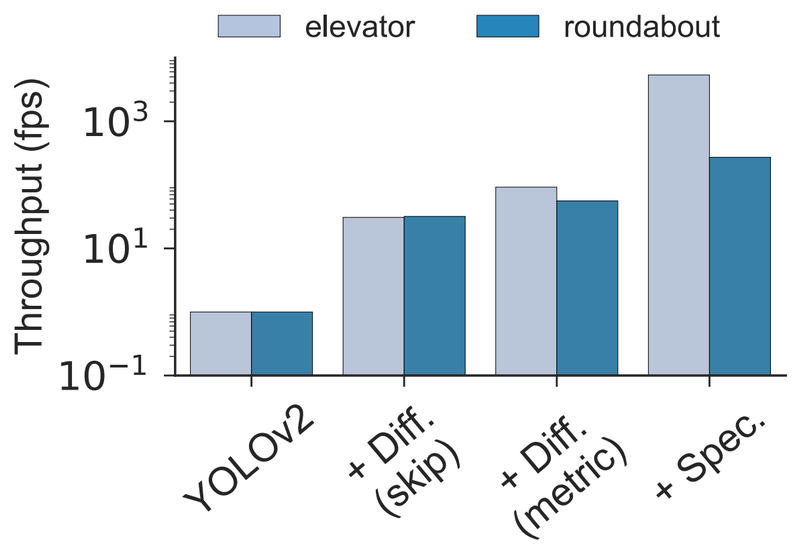
(d) night-street



(e) store



(f) elevator



Factor Analysis

Class thought experiment

Click to open expanded view

AWS DeepLens - Deep learning enabled video camera for developers

Amazon Web Services

Price: $249.00 FREE Shipping for Prime members

This item will be released on June 14, 2018.
Pre-order now.
Ships from and sold by Amazon.com.

- Deep learning in your hands - A fully programmable video camera designed to expand deep learning skills.
- Start learning right away - Learn the basics of deep learning through example projects, computer vision models, tutorials, and real world, hands-on exploration on a physical device.
- A new way to learn machine learning - Allows developers of all skill levels get started with deep learning in less than 10 minutes.
- Sample projects - AWS DeepLens can run custom models from Amazon SageMaker, and comes with a collection of pre-trained models ready to run on the device with a single click.
- Fully programmable - Easy to customize and is fully programmable using AWS Lambda providing a familiar programming environment for developers to experiment with.
- Integrated with AWS - Stream video back to AWS using Amazon Kinesis Video Streams, and apply more advanced video analytics using Amazon Rekognition Video. The device also connects securely to AWS IoT, Amazon SQS, Amazon SNS, Amazon S3, Amazon DynamoDB, and more.

Share

Pre-order: Add to Cart

or 1-Click Checkout

Pre-order with 1-Click

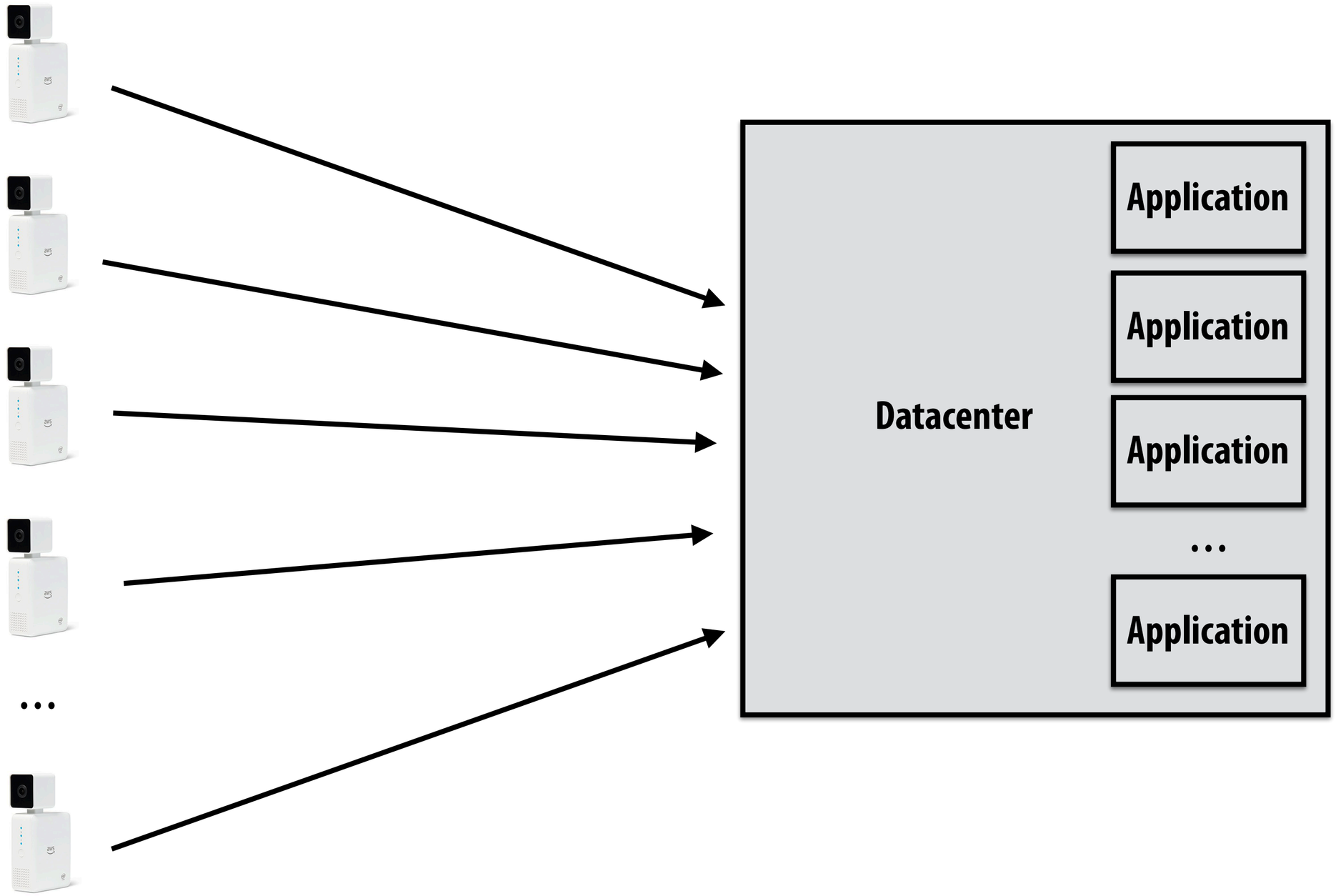
Not yet released
Free shipping once released

Ship to:
Kayvon Fatahalian- PALO ALTO

Add to List

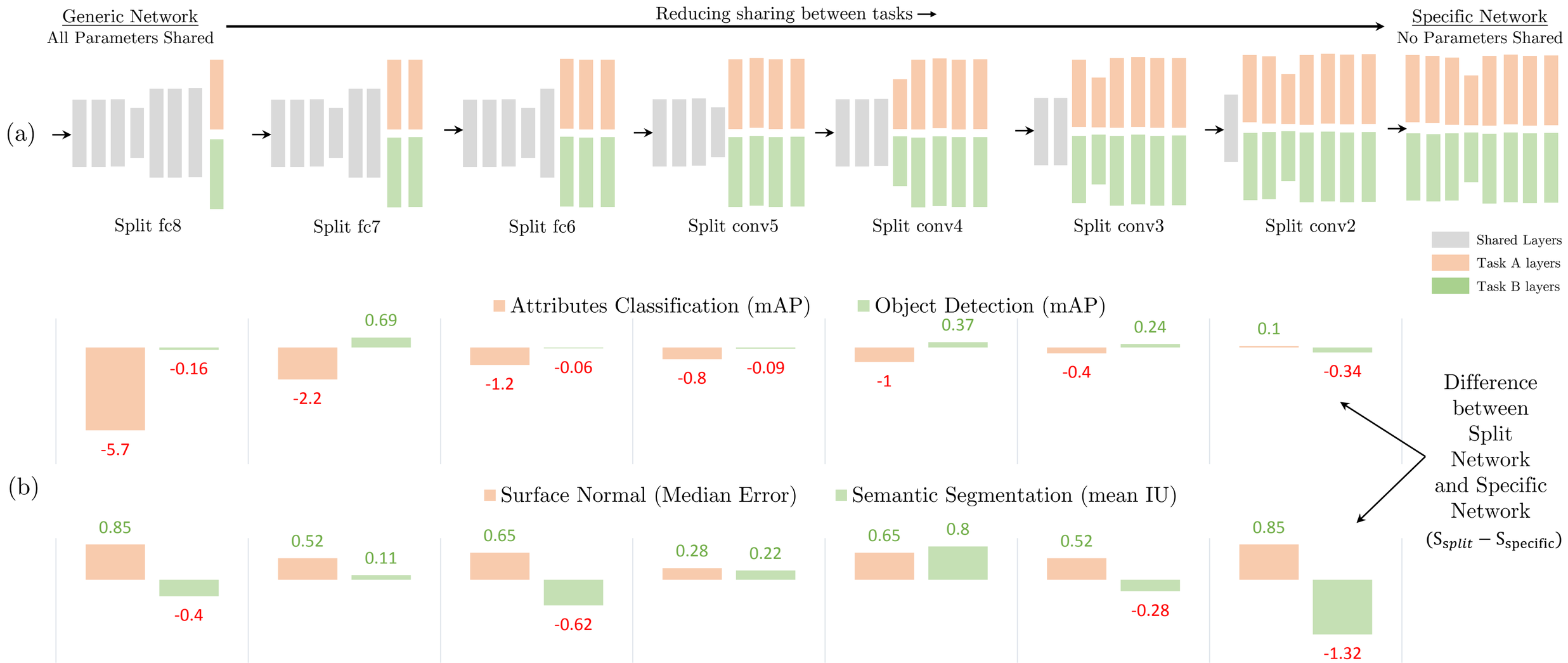
Setup

- Many cameras (e.g., traffic cameras in a major US city, cameras in a future Amazon Go store)
- Many applications needing access to streams for different tasks

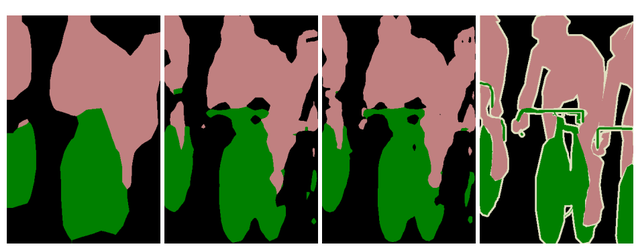


One more idea

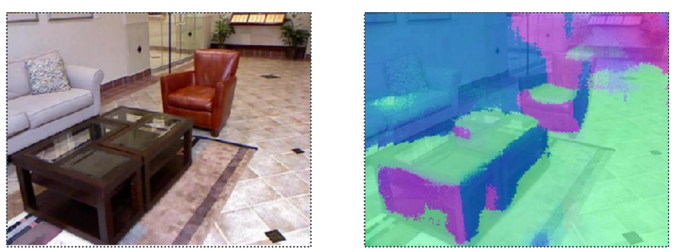
- System design forces different networks to share the same stem
- Amortize cost of early (and expensive layers) across different tasks



Semantic Segmentation

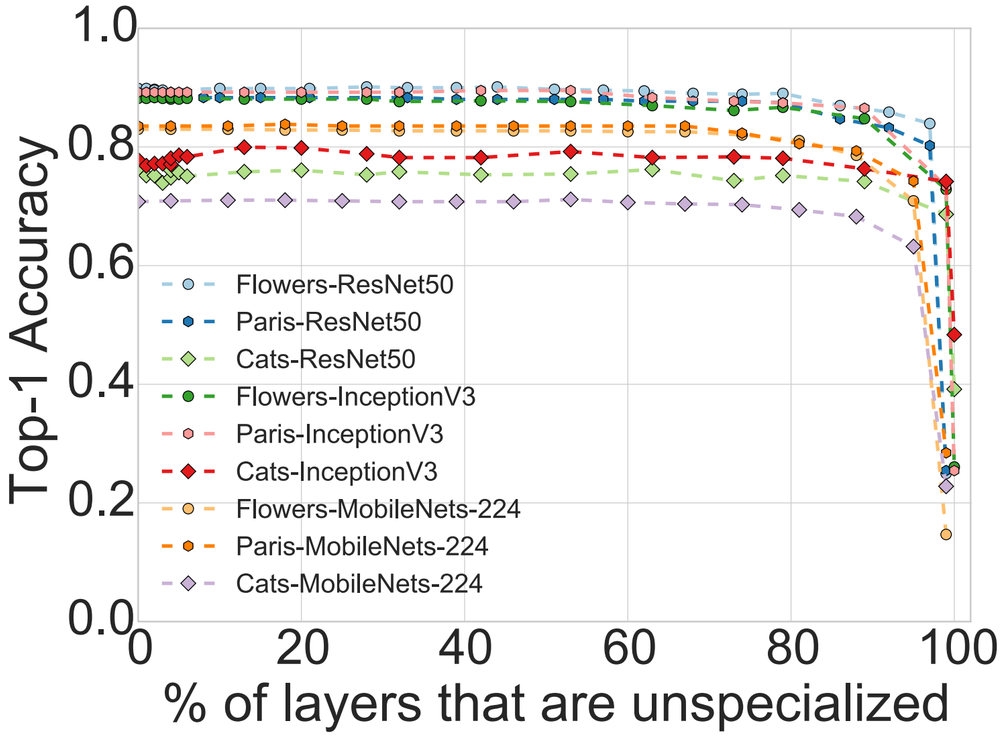


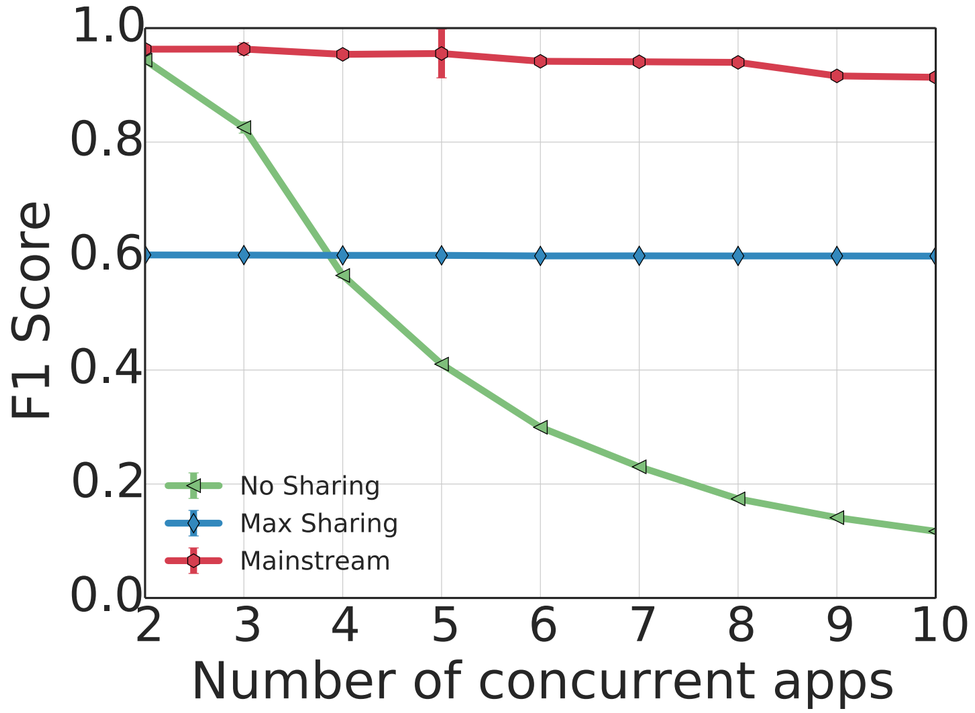
Normal estimation



[Image from: Misra et al. CVPR 2016]

Emerging interest in systems that learn how to share computation across users





[Jiang 18]

Summary

- An increasing number of cameras across world will be capturing near continuous video

- Many applications will seek to extract value from these data streams
  - Implications for efficiency of cities (transportation, infrastructure monitoring), brick-and-mortar commerce, security, health-care, robotics, human-robot interactions, autonomous vehicles

- Need significant efficient gains to process this worldwide visual signal
  - Hardware specialization (last time)
  - Algorithmic techniques to reduce the cost of inference
    - Specialization to video stream or scene context
    - Exploit temporal coherence of video

Discussion: privacy and ethics

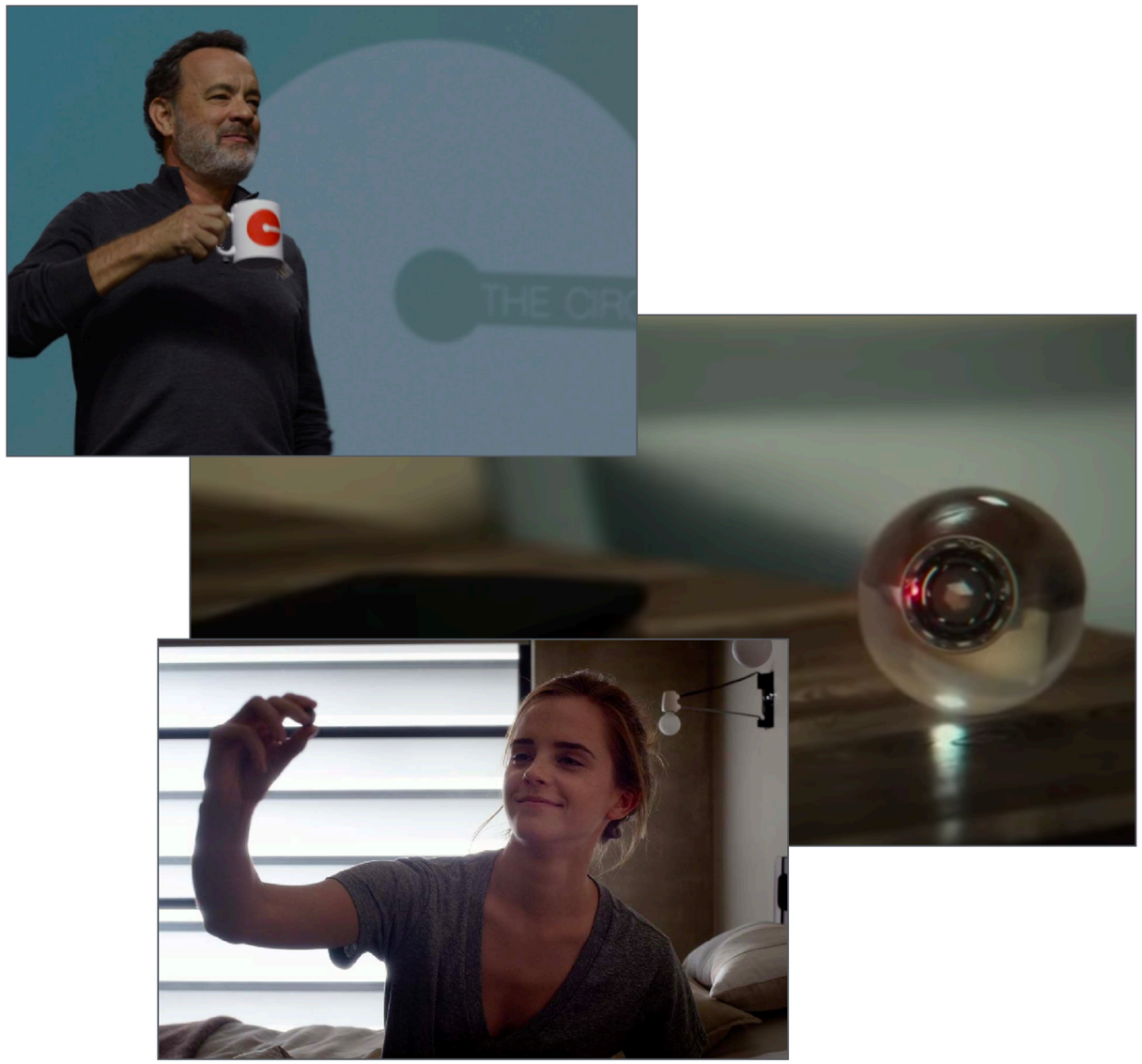


Image credit: The Circle (movie)