Advanced Computer Vision

Tanveer Hussain
htanveer3797@gmail.com

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

Course designed by **Naeem Ullah Khan** (naeemullah.khan@kaust.edu.sa), updated by Tanveer Hussain

Videos

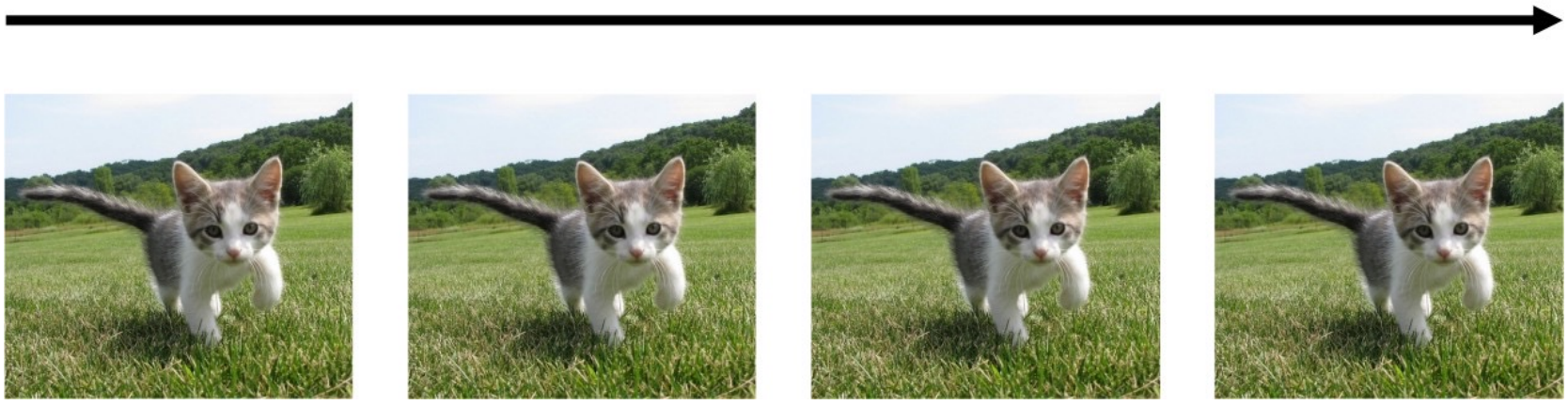
Videos are all around us
Span an enormous space of spatial and temporal signals

Video = 2D + Time

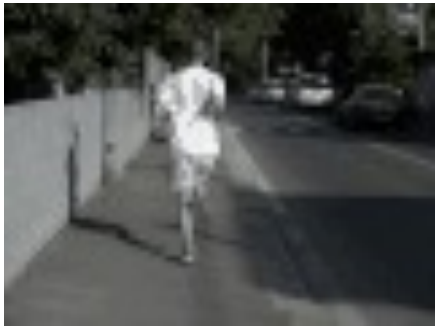
A video is a **sequence** of images

4D tensor: T x 3 x H x W

(or 3 x T x H x W)



Video Classification



Input video:
T x 3 x H x W

Swimming
Running
Jumping
Eating
Standing

Video Classification

Images: Recognize **objects**

Dog
Cat
Fish
Truck

Videos: Recognize **actions**

Swimming
Running
Jumping
Eating
Standing

Problem: Videos are big!

Input video:
T x 3 x H x W

Videos are ~30 frames per second (fps)

Size of uncompressed video
(3 bytes per pixel):

SD (640 x 480): **~1.5 GB per minute**
HD (1920 x 1080): ~**10 GB per minute**

Problem: Videos are big!

Input video:
T x 3 x H x W

Videos are ~30 frames per second (fps)

Size of uncompressed video (3 bytes per pixel):

Video Summarization

SD (640 x 480): **~1.5 GB per minute**
HD (1920 x 1080): ~**10 GB per minute**

Solution: Train on short **clips:** low fps and low spatial resolution
e.g. T = 16, H=W=112
(3.2 seconds at 5 fps, 588 KB)

Training on Clips

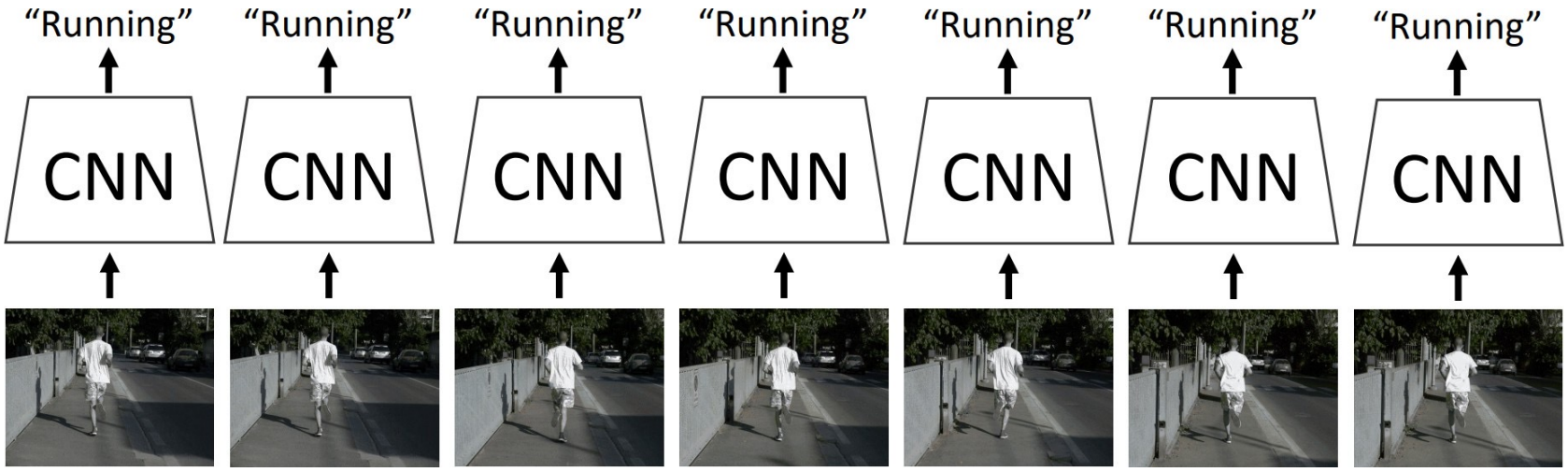
Raw video: Long, high FPS

Training: Train model to classify short **clips** with low FPS

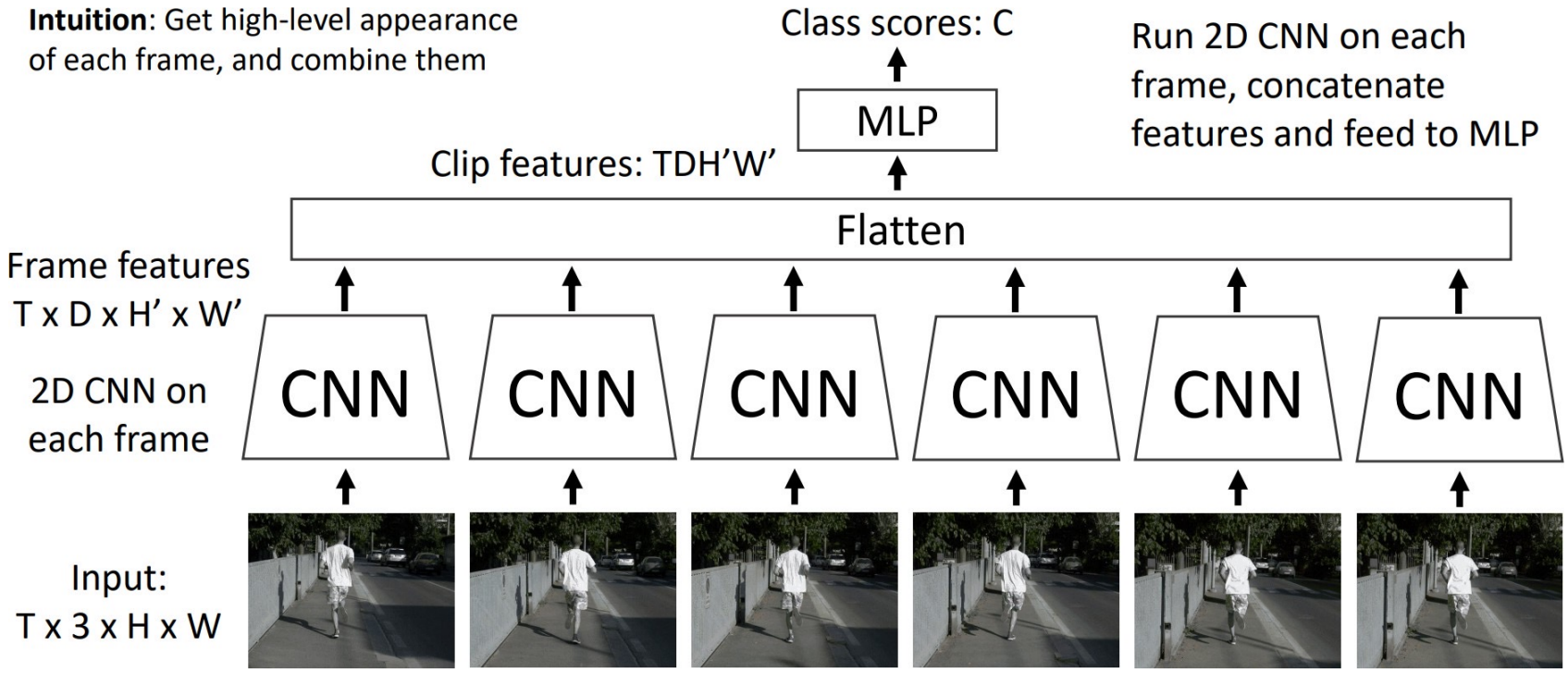
Testing: Run model on different clips, average predictions

Video Classification: Single-Frame CNN

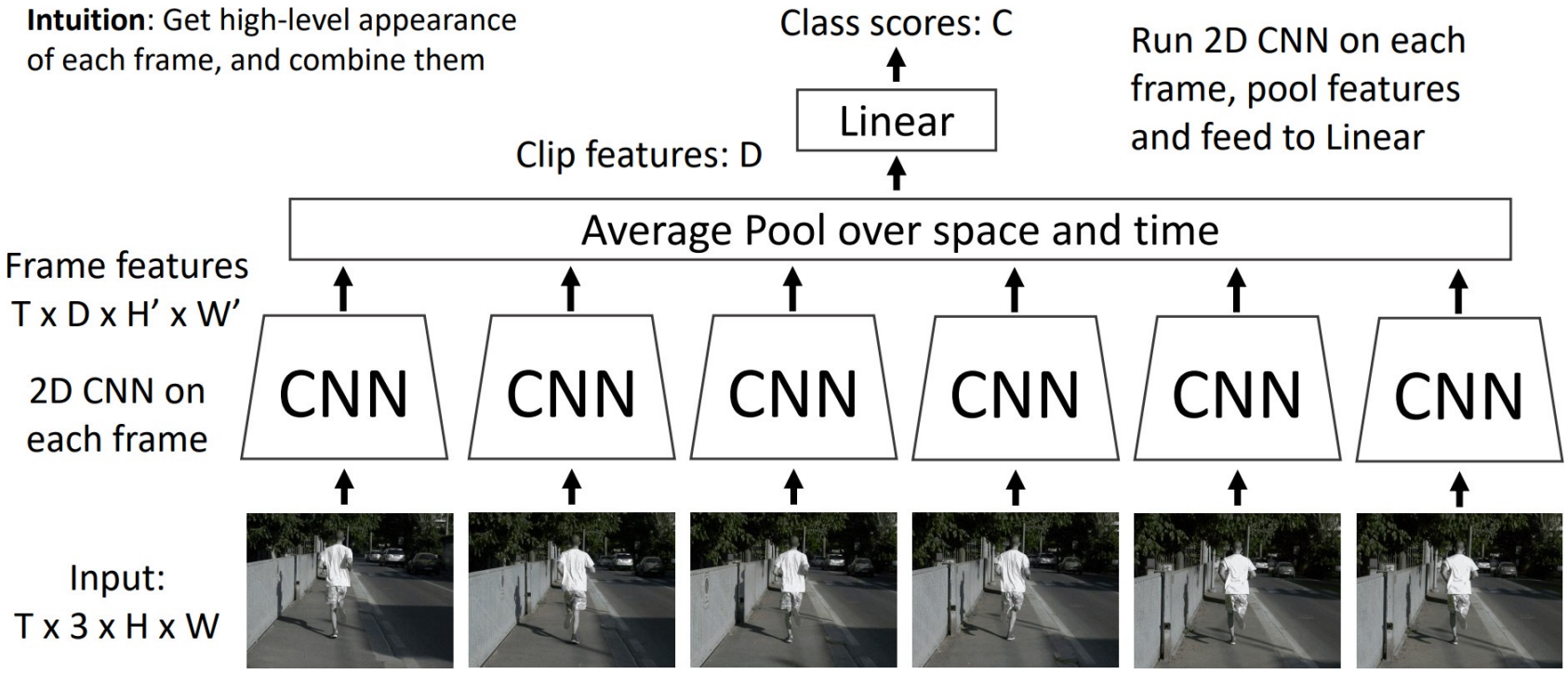
- **Simple idea:** Train normal 2D CNN to classify video frames independently!
- Average predicted probs at test-time
- Often a very strong baseline for video classification



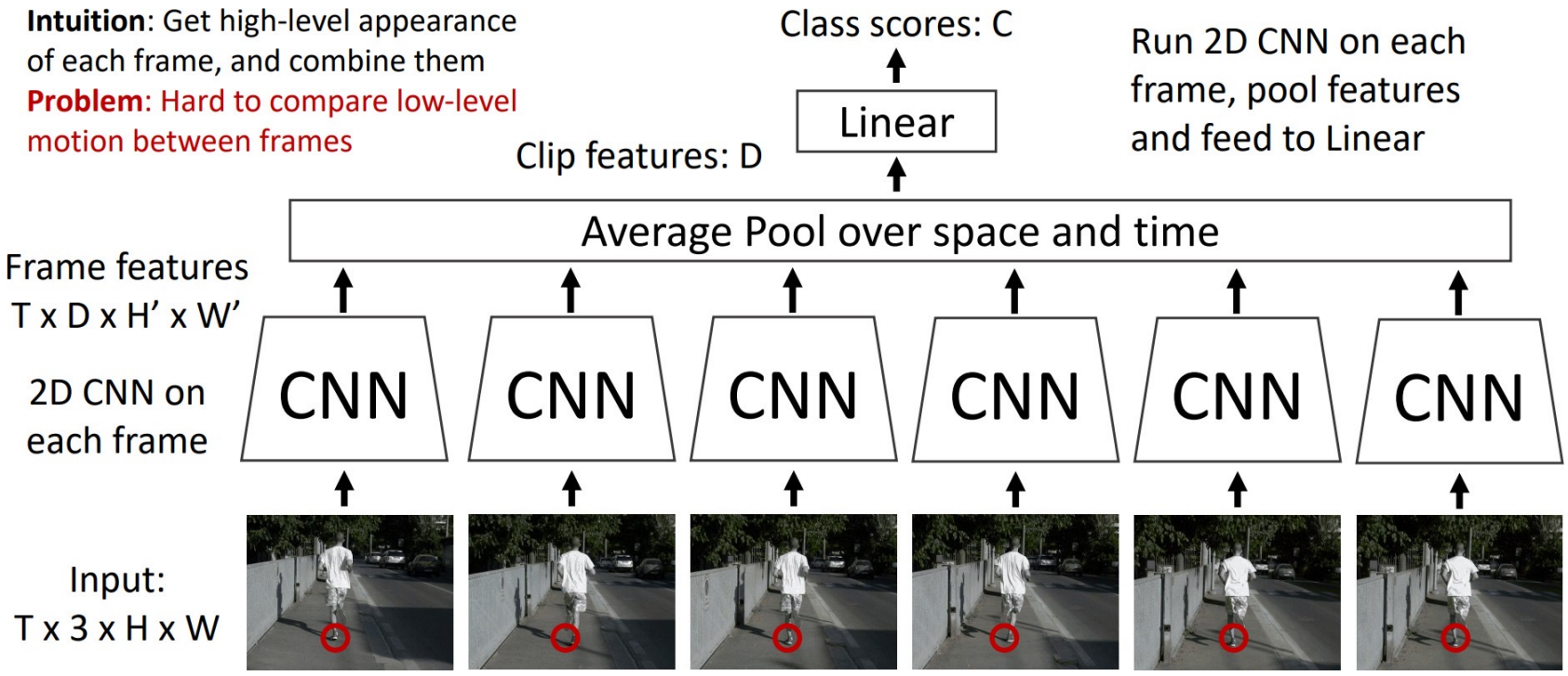
Video Classification: Late Fusion (with FC layers)



Video Classification: Late Fusion (with pooling)

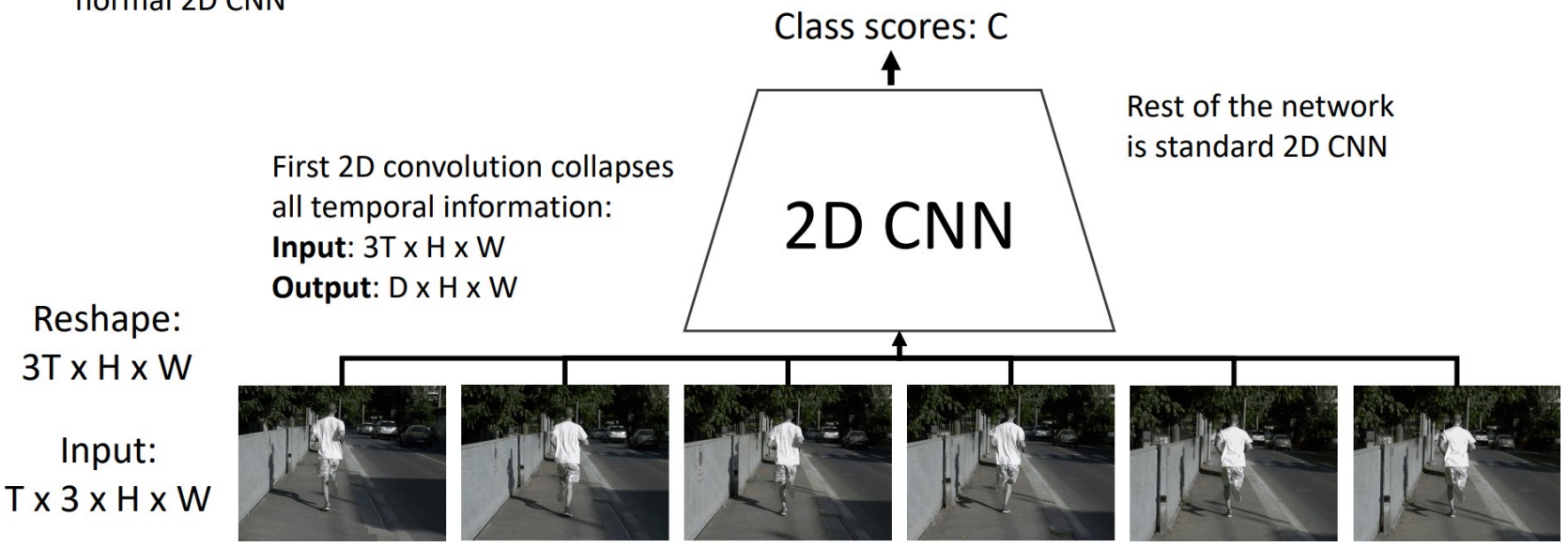


Video Classification: Late Fusion (with pooling)



Video Classification: Early Fusion

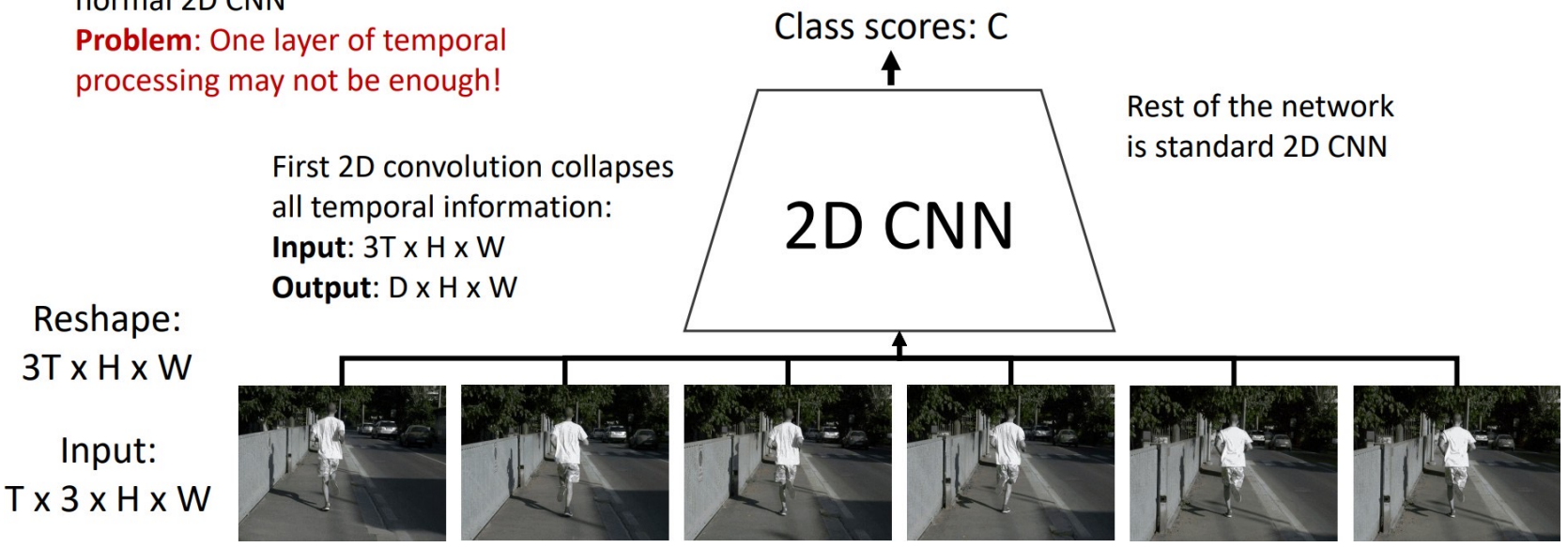
Intuition: Compare frames with very first conv layer, after that normal 2D CNN



Video Classification: Early Fusion

Intuition: Compare frames with very first conv layer, after that normal 2D CNN

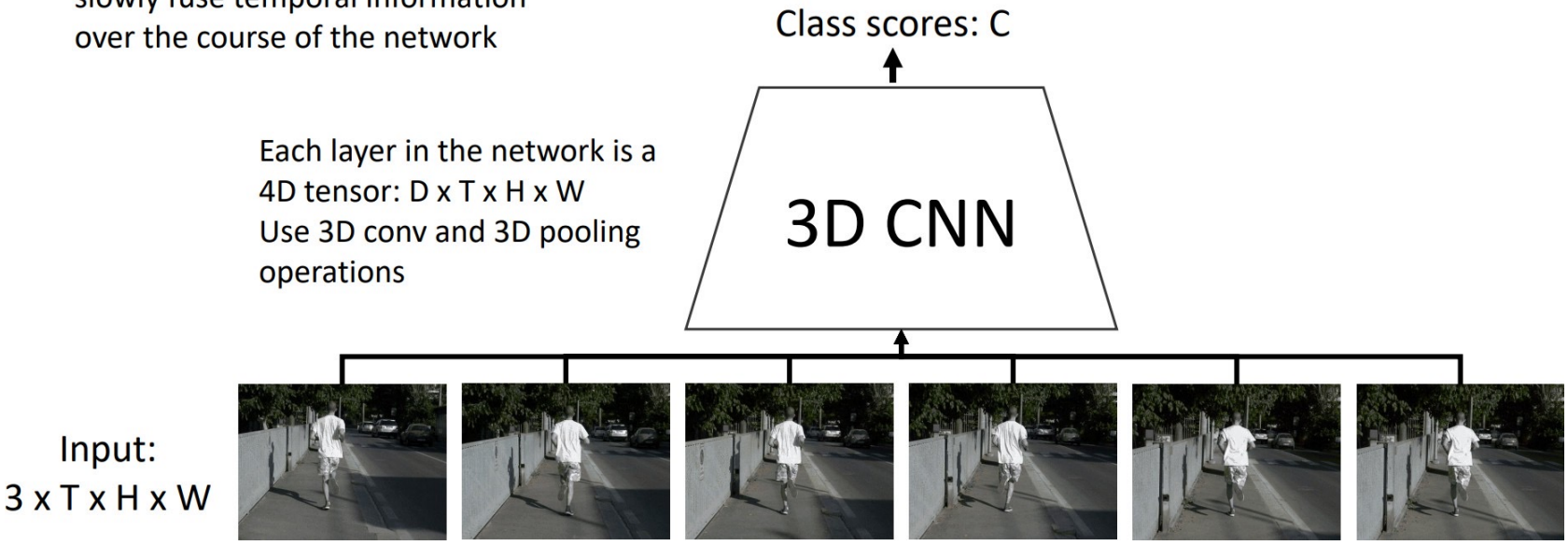
Problem: One layer of temporal processing may not be enough!



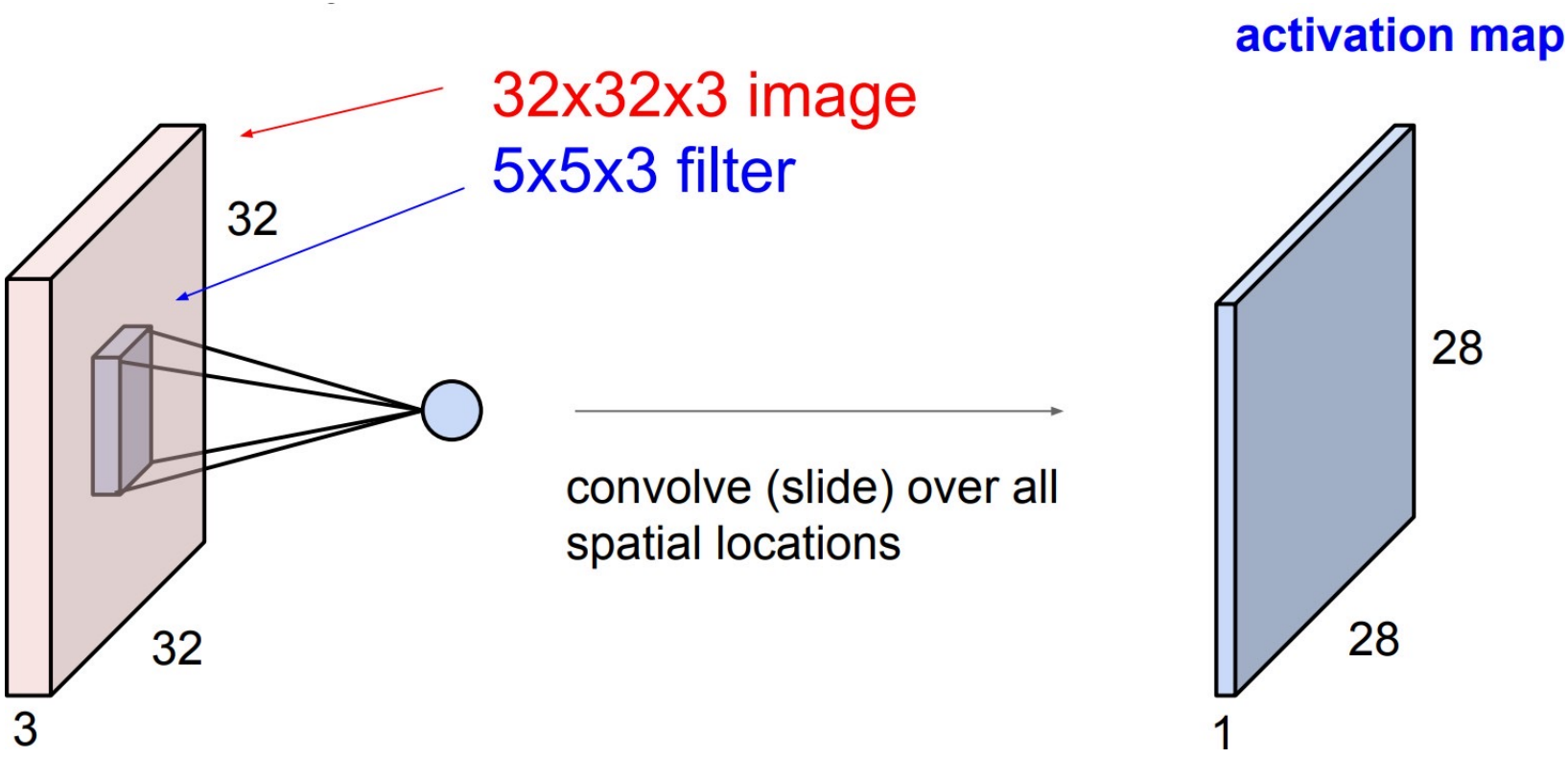
Video Classification: 3D CNN

Intuition: Use 3D versions of convolution and pooling to slowly fuse temporal information over the course of the network

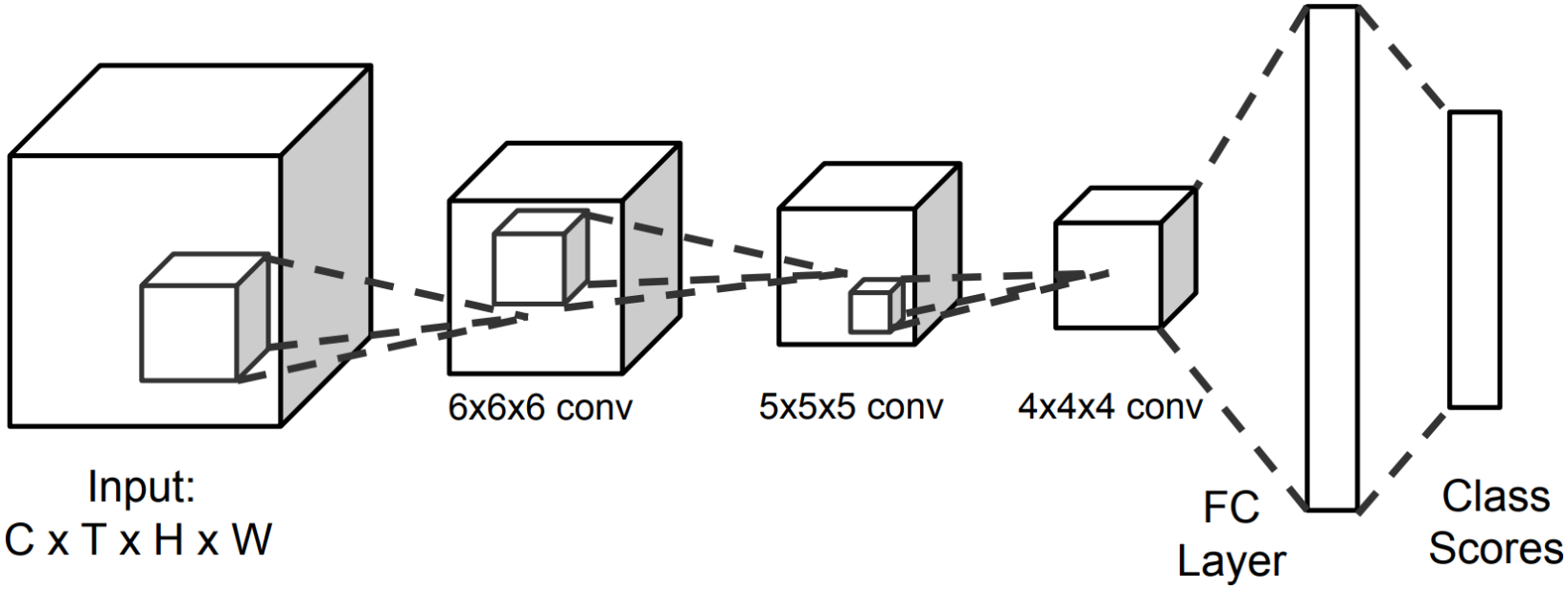
Each layer in the network is a 4D tensor: D x T x H x W
Use 3D conv and 3D pooling operations



Convolution Layer



3D Convolution



Video Datasets

space of video >> space of image → lots of training data

"ImageNet"-equivalent dataset for videos?

Massive human labelling efforts

UCF101

YouTube videos

13320 videos, 101 action categories

Kinetics

Kinetics

YouTube videos

650,000 video clips, 600 human action classes

YouTube | 8M

YouTube-8M

8M video clips, Machine-generated annotations from 3,862 classes

Sports-1M

YouTube videos

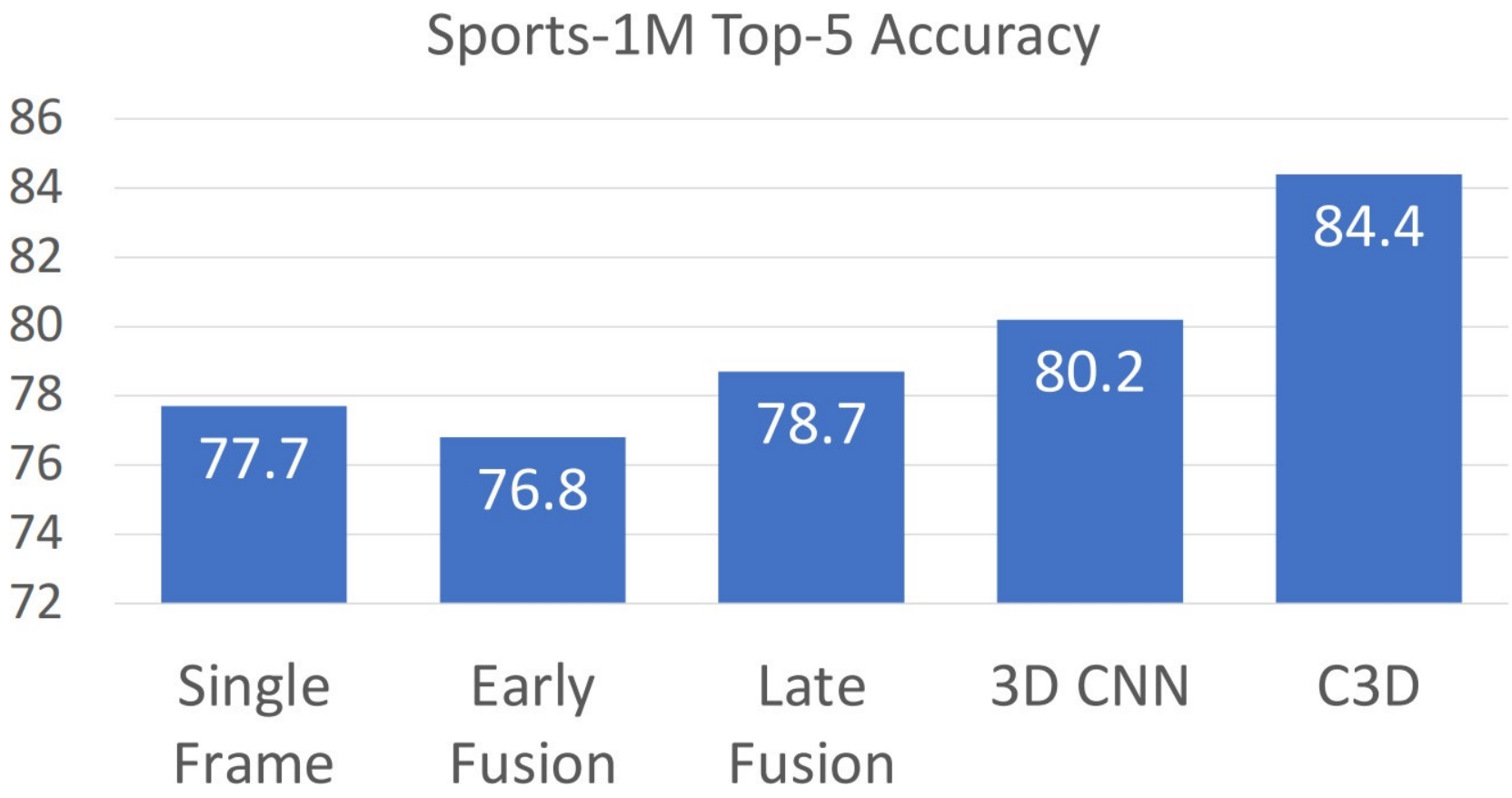
1,133,157 videos, 487 sports labels

C3D: The VGG of 3D CNNs

- 3D CNN that uses all 3x3x3 conv and 2x2x2 pooling (except Pool1 which is 1x2x2)
- Released model pretrained on Sports-1M: Many people used this as a video feature extractor
- Problem: 3x3x3 conv is very expensive!
  - AlexNet: 0.7 GFLOP
  - VGG-16: 13.6 GFLOP
  - C3D: 39.5 GFLOP (2.9x VGG!)

Layer	Size
Input	3 x 16 x 112 x 112
Conv1 (3x3x3)	64 x 16 x 112 x 112
Pool1 (1x2x2)	64 x 16 x 56 x 56
Conv2 (3x3x3)	128 x 16 x 56 x 56
Pool2 (2x2x2)	128 x 8 x 28 x 28
Conv3a (3x3x3)	256 x 8 x 28 x 28
Conv3b (3x3x3)	256 x 8 x 28 x 28
Pool3 (2x2x2)	256 x 4 x 14 x 14
Conv4a (3x3x3)	512 x 4 x 14 x 14
Conv4b (3x3x3)	512 x 4 x 14 x 14
Pool4 (2x2x2)	512 x 2 x 7 x 7
Conv5a (3x3x3)	512 x 2 x 7 x 7
Conv5b (3x3x3)	512 x 2 x 7 x 7
Pool5	512 x 1 x 3 x 3
FC6	4096
FC7	4096
FC8	C

Results

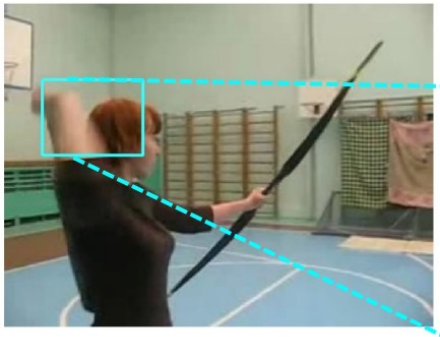

Sports-1M Top-5 Accuracy
86
84
82
80
78
76
74
72
77.7
76.8
78.7
80.2
84.4
Single Frame
Early Fusion
Late Fusion
3D CNN
C3D

Recognizing Actions from Motion

- We can easily recognize actions using only motion information

Measuring Motion: Optical Flow

Image at frame t



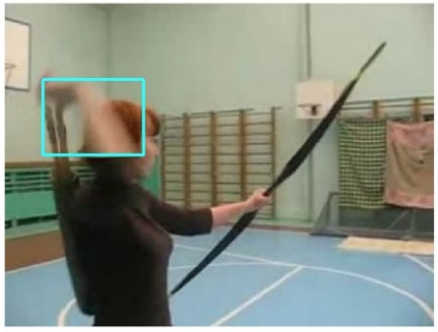
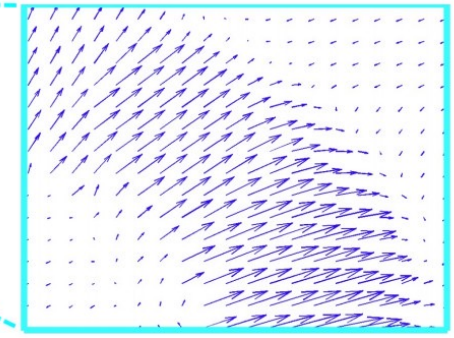


Image at frame t+1

Optical flow gives a displacement field F between images I_t and I_{t+1}

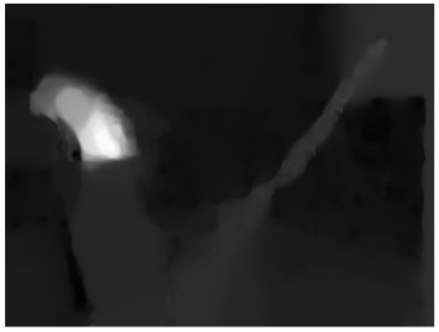


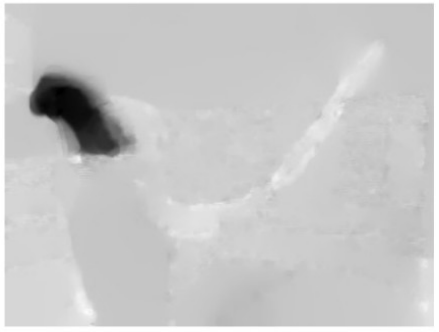
Tells where each pixel will move in the next frame:

$F(x, y) = (dx, dy)$

$I_{t+1}(x+dx, y+dy) = I_t(x, y)$

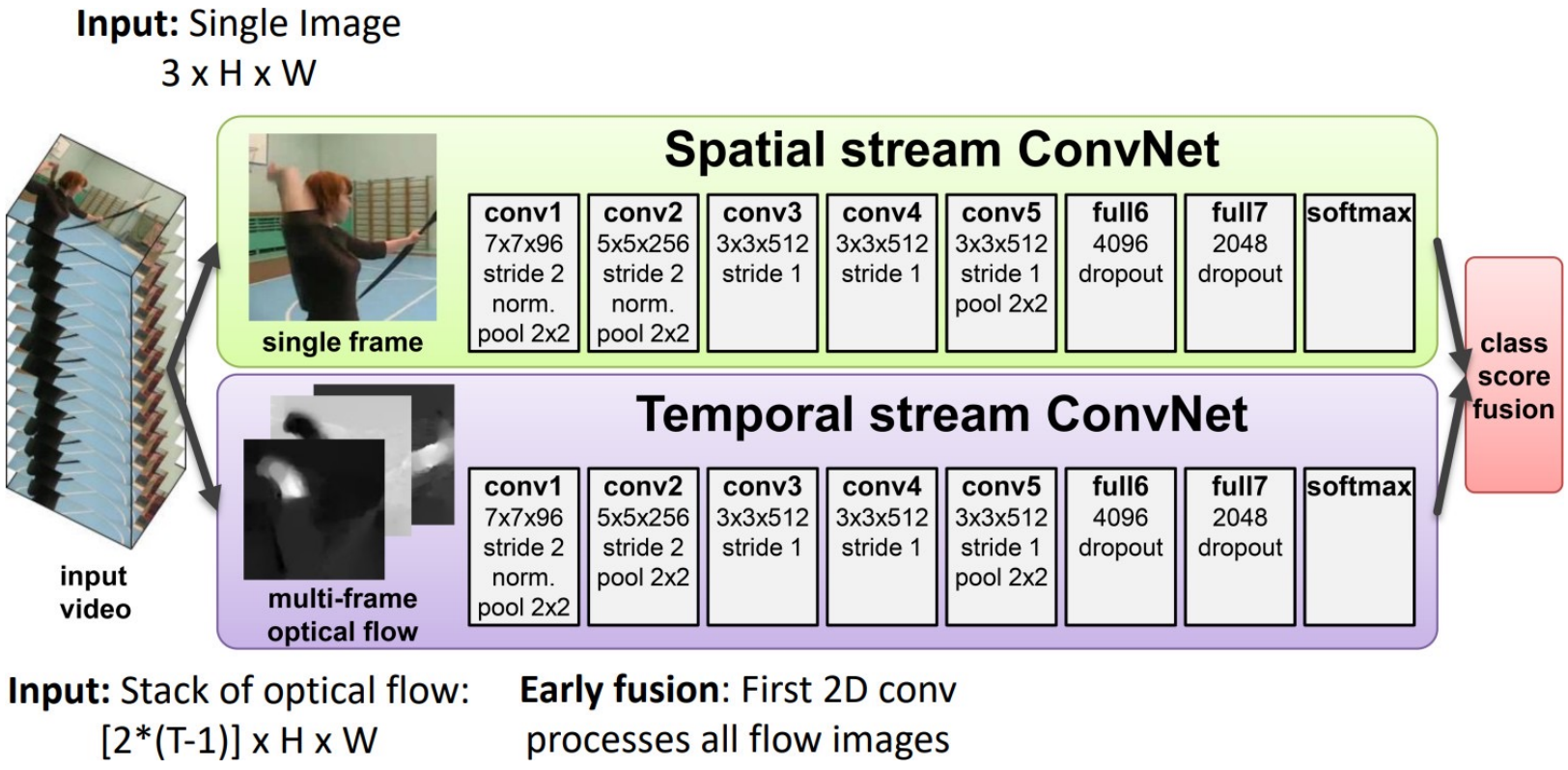
Horizontal flow dx



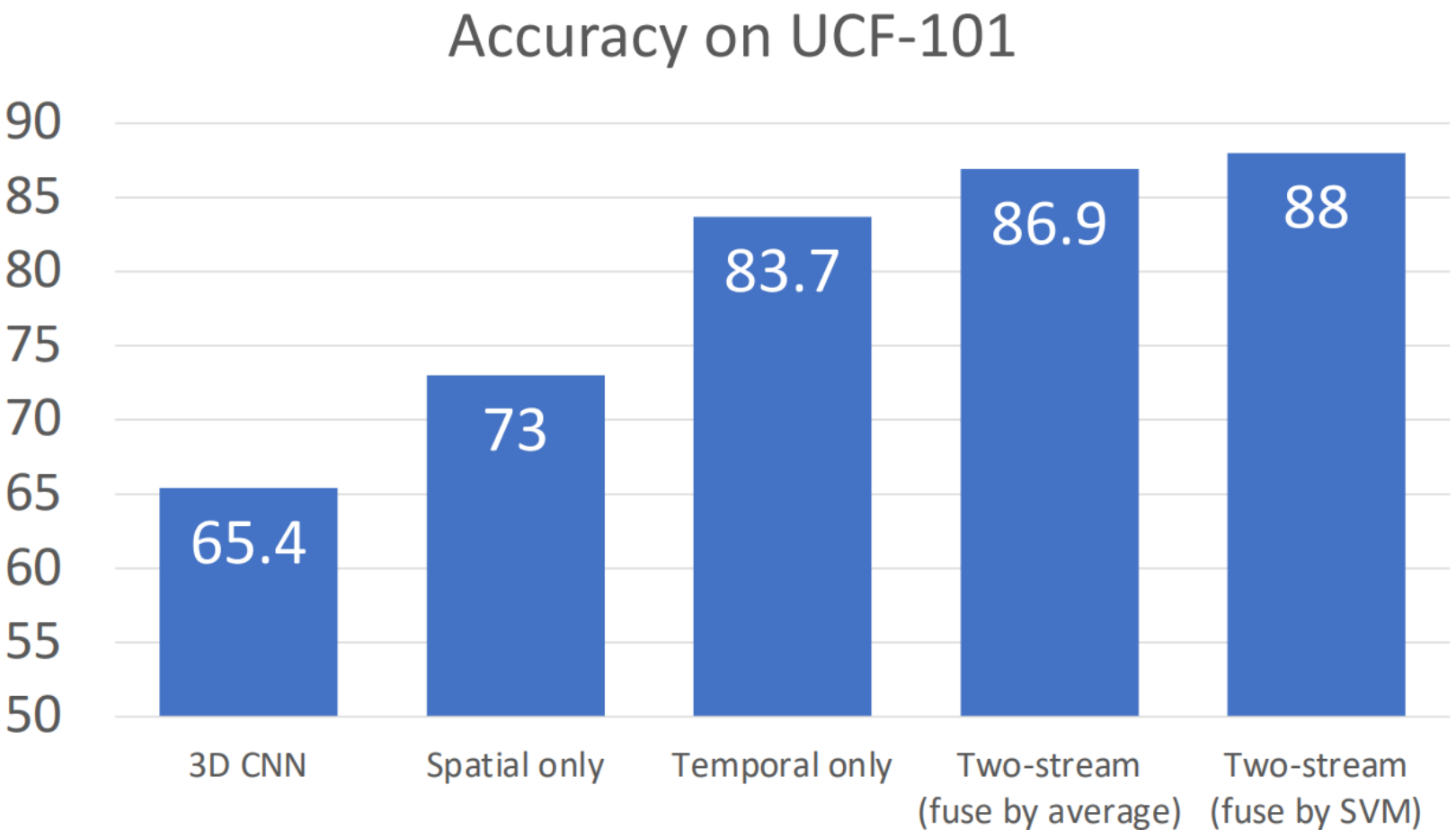


Vertical Flow dy

Separating Motion and Appearance: Two-Stream Networks

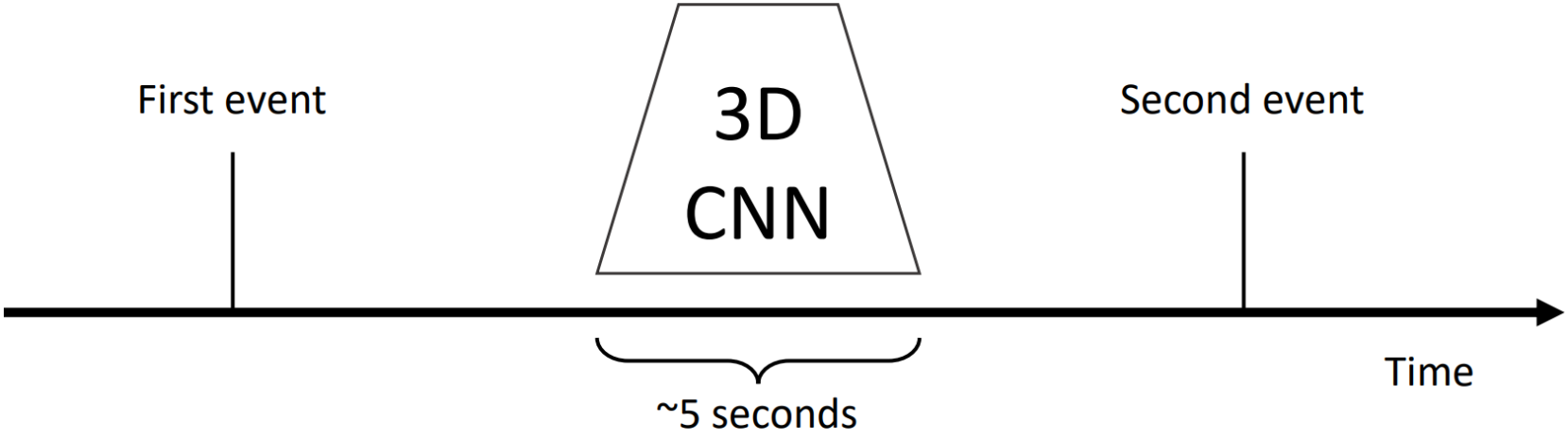


Separating Motion and Appearance: Two-Stream Networks



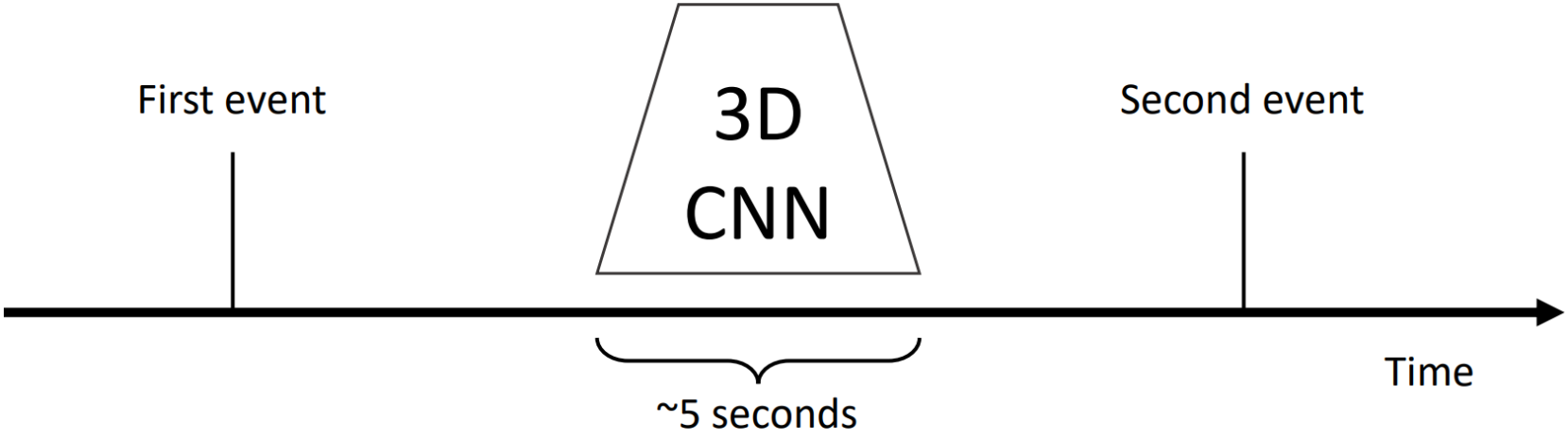
Modeling long-term temporal structure

- So far all our temporal CNNs only model local motion between frames in very short clips of $\sim 2-5$ seconds. What about long-term structure?



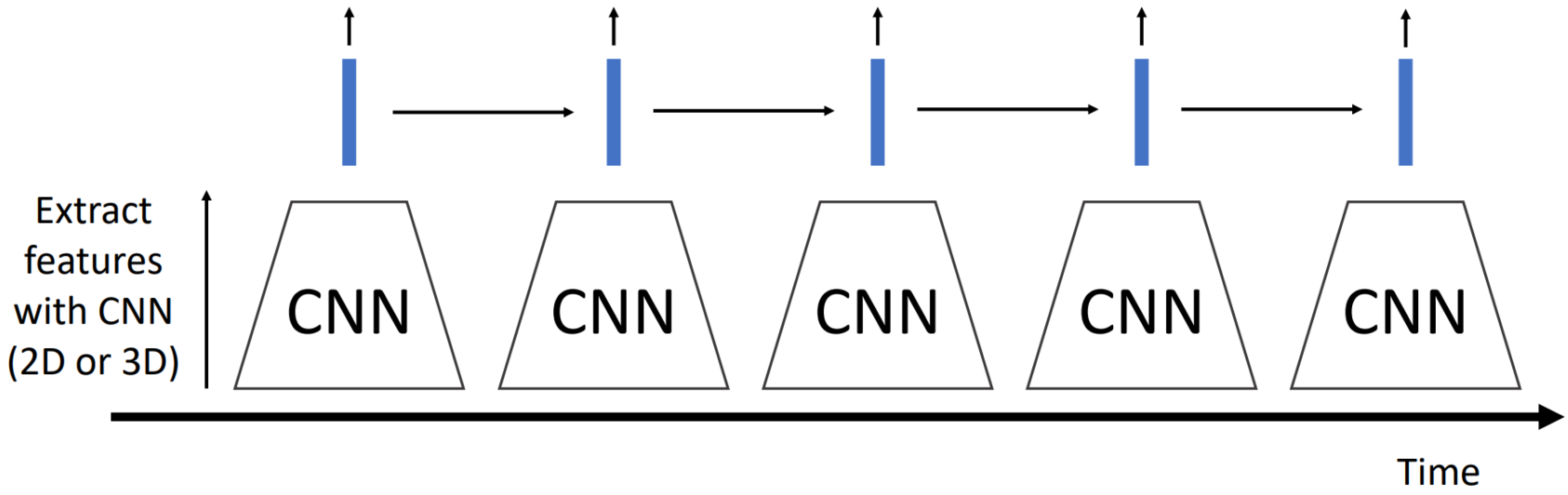
Modeling long-term temporal structure

- So far all our temporal CNNs only model local motion between frames in very short clips of $\sim 2-5$ seconds. What about long-term structure?
- We know how to handle sequences!
- How about recurrent networks?

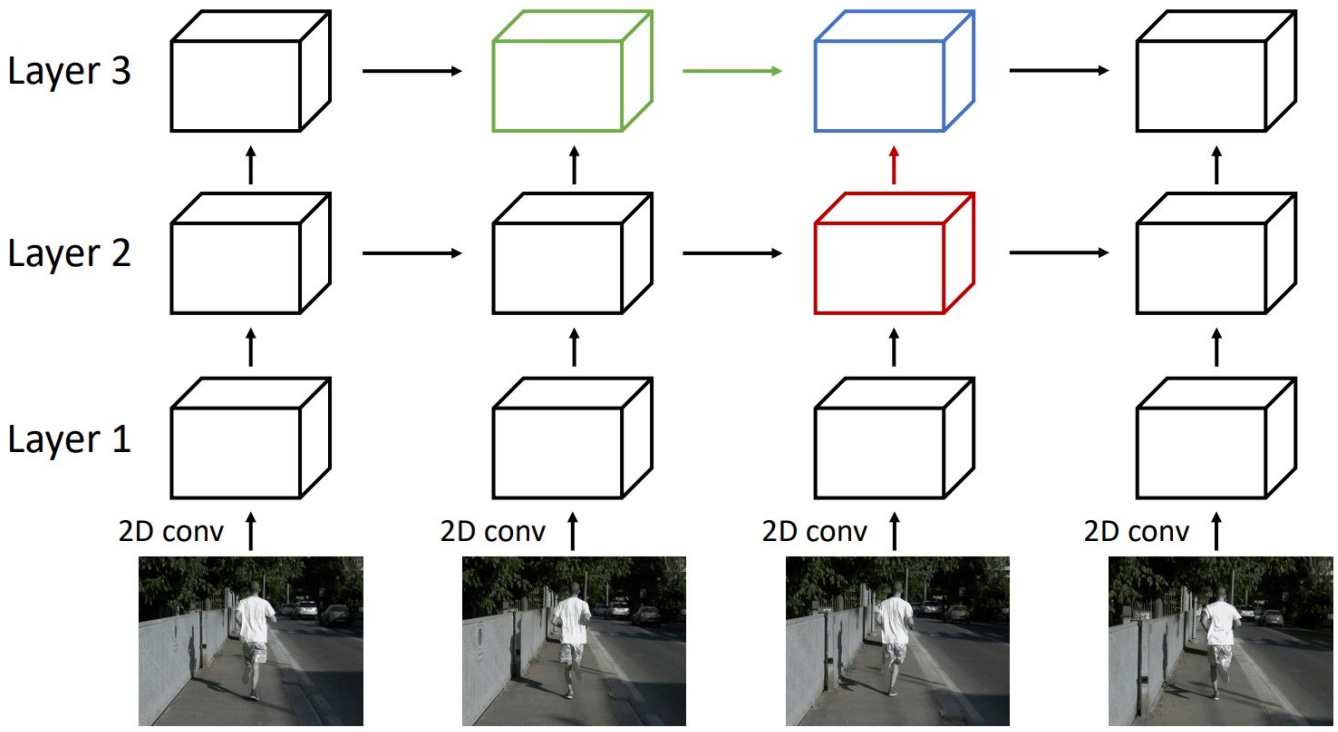


Modeling long-term temporal structure

- Process local features using recurrent network (e.g. LSTM)
- Many to many: one output per video frame



Recurrent Convolutional Network



Entire network uses 2D feature maps: C x H x W

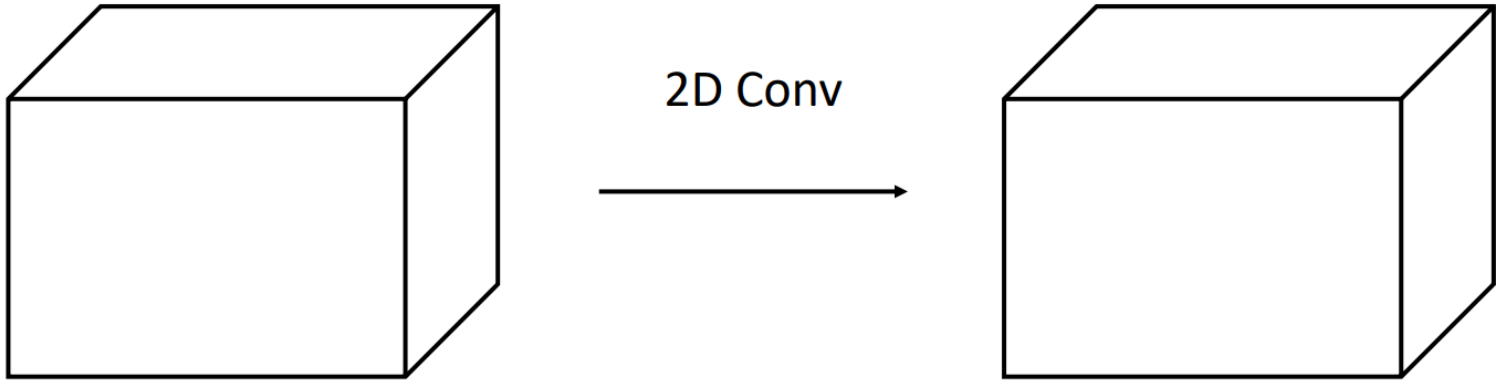
Each depends on two inputs:
1. Same layer, previous timestep
2. Prev layer, same timestep

Use different weights at each layer, share weights across time

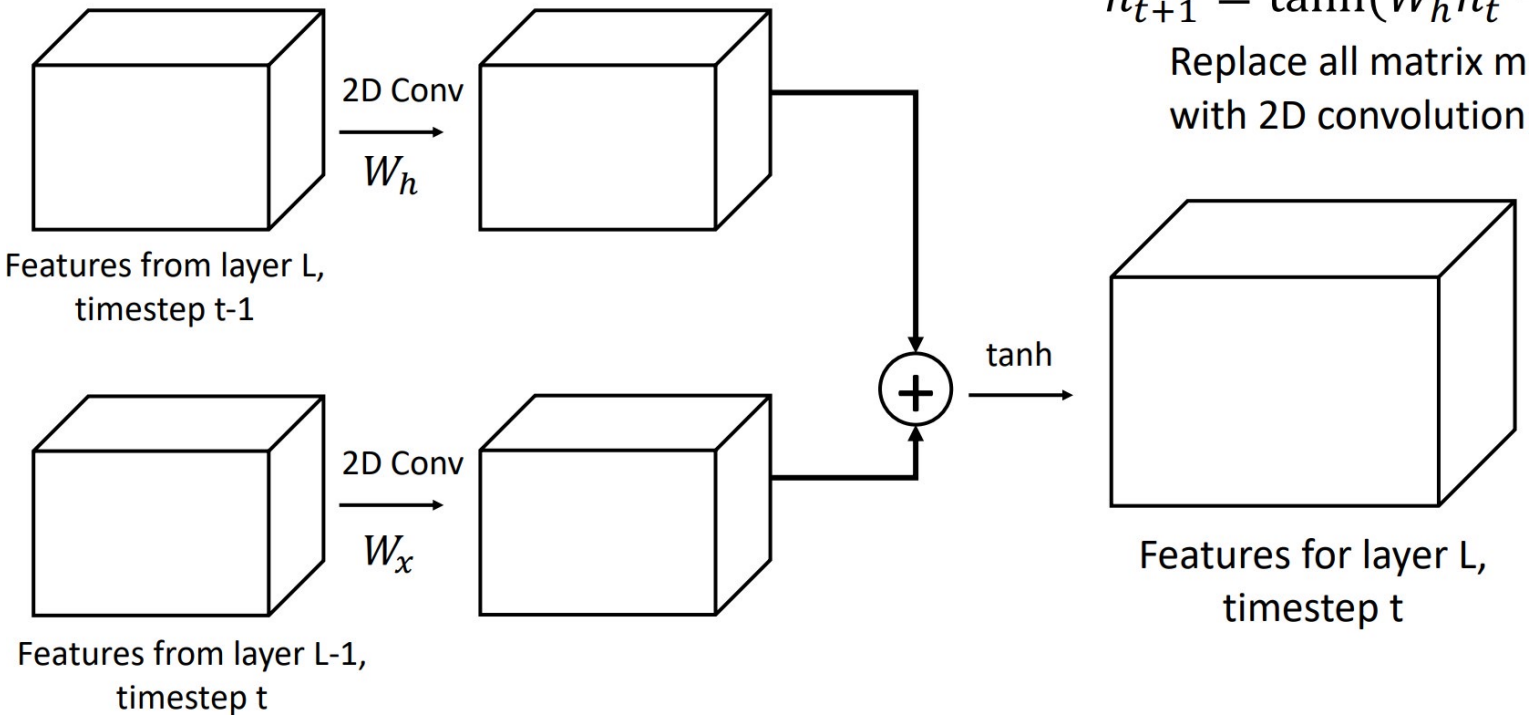
Ballas et al, "Delving Deeper into Convolutional Networks for Learning Video Representations", ICLR 2016

Recurrent Convolutional Network

Normal 2D CNN:



Recurrent Convolutional Network

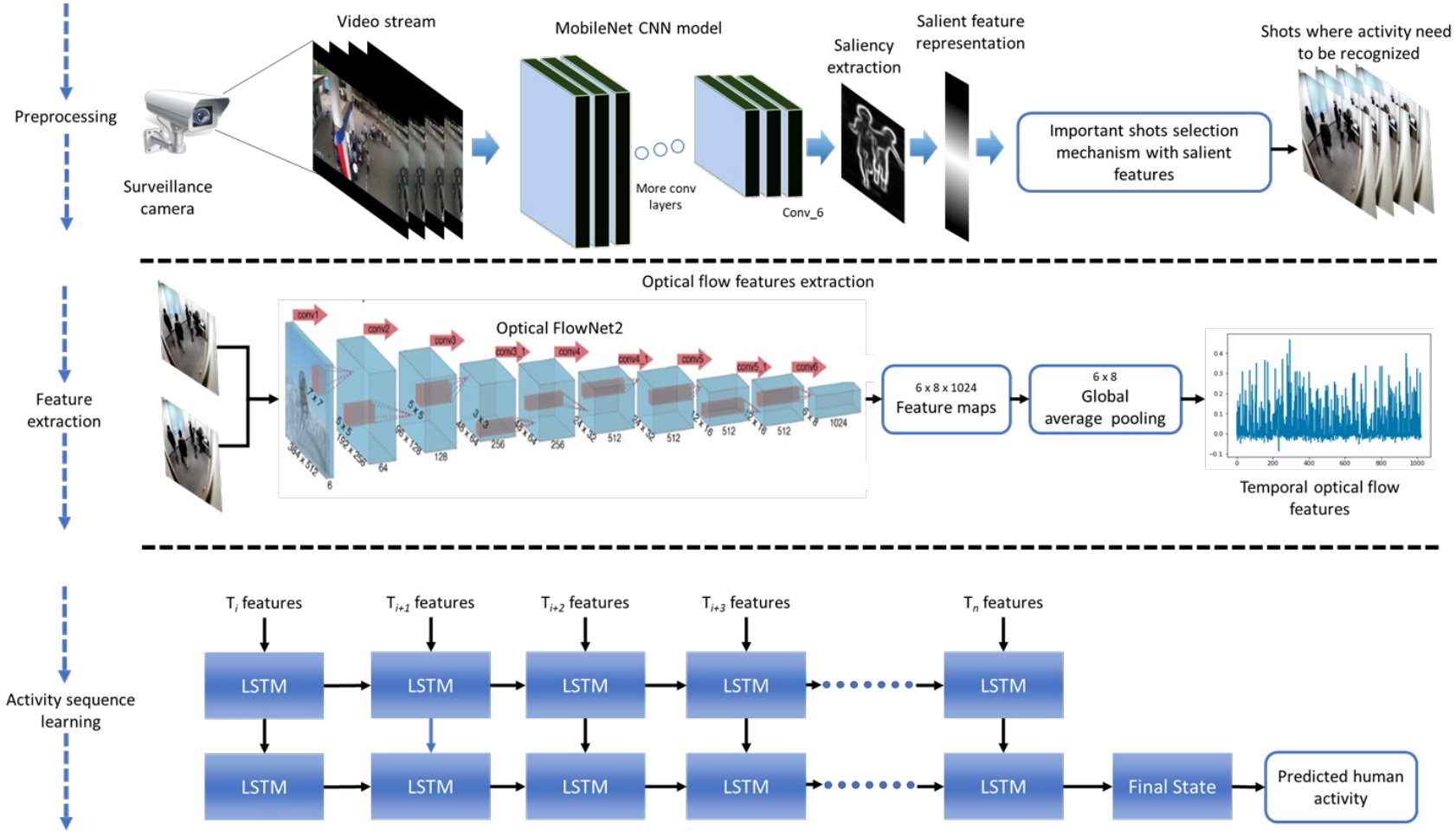


Recall: Vanilla RNN

$$h_{t+1} = \tanh(W_h h_t + W_x x)$$

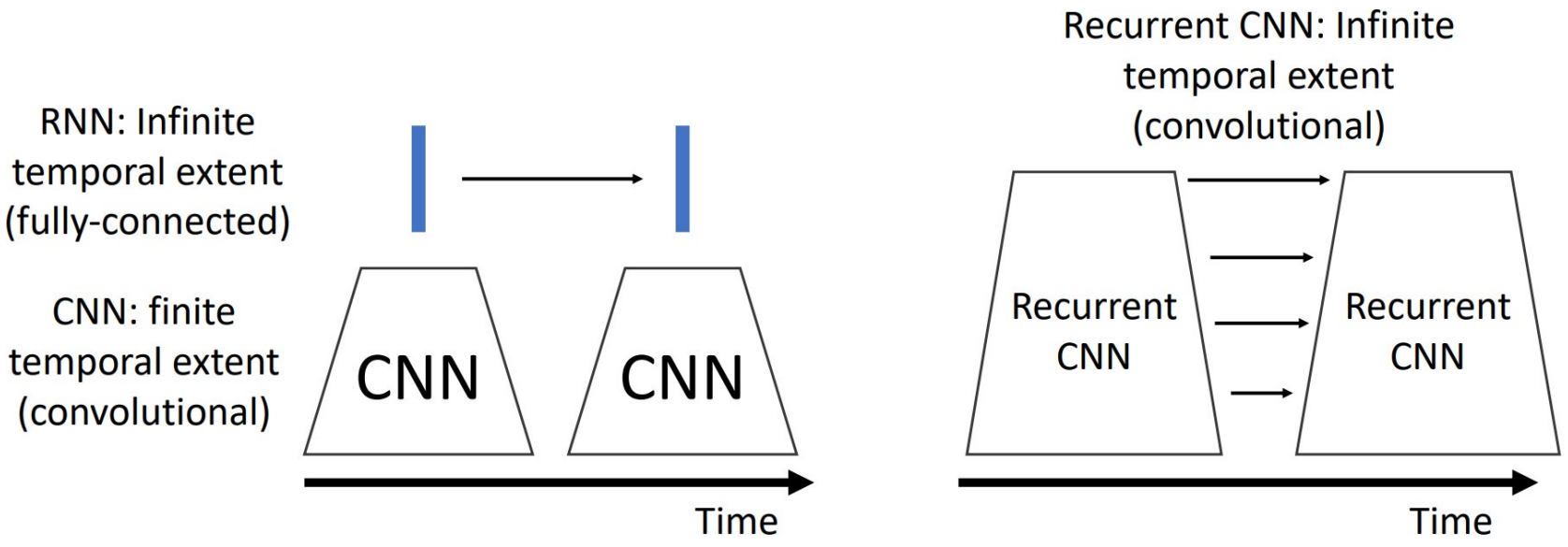
Replace all matrix multiply with 2D convolution!

Recurrent Convolutional Network

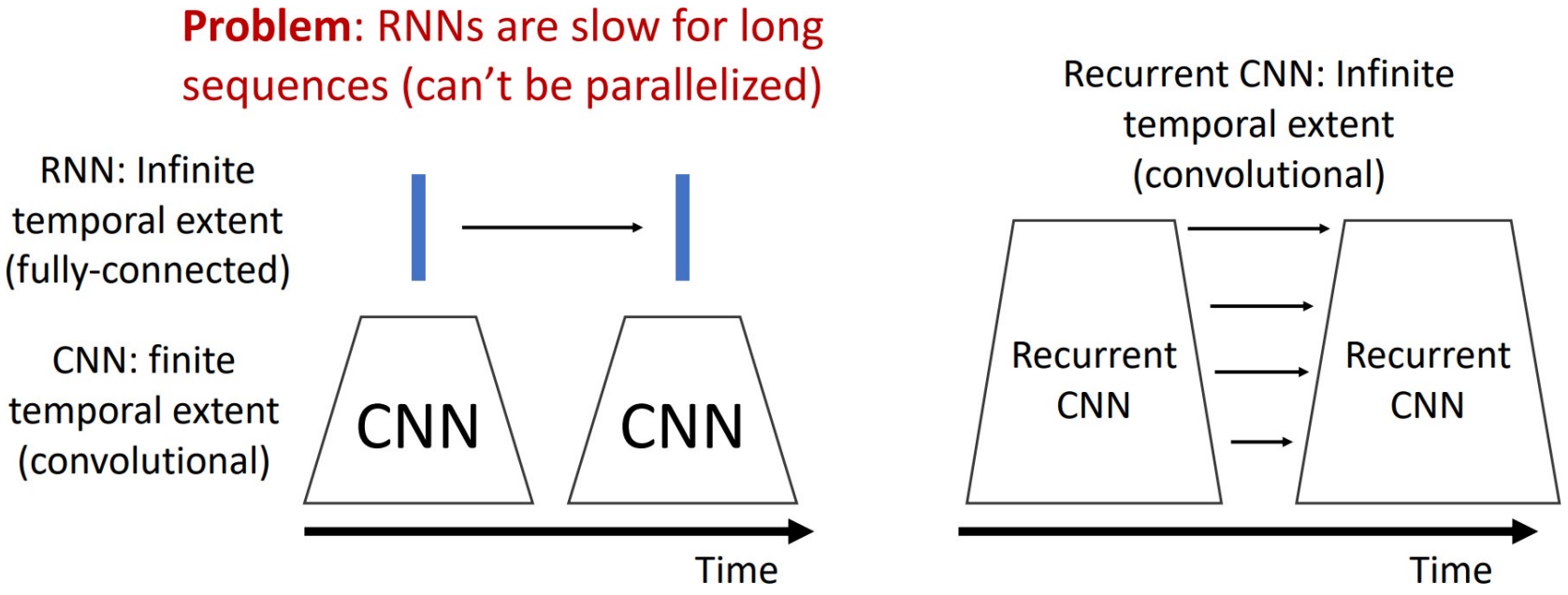


Source code: https://github.com/Aminullah6264/Activity_Rec_ML-LSTM

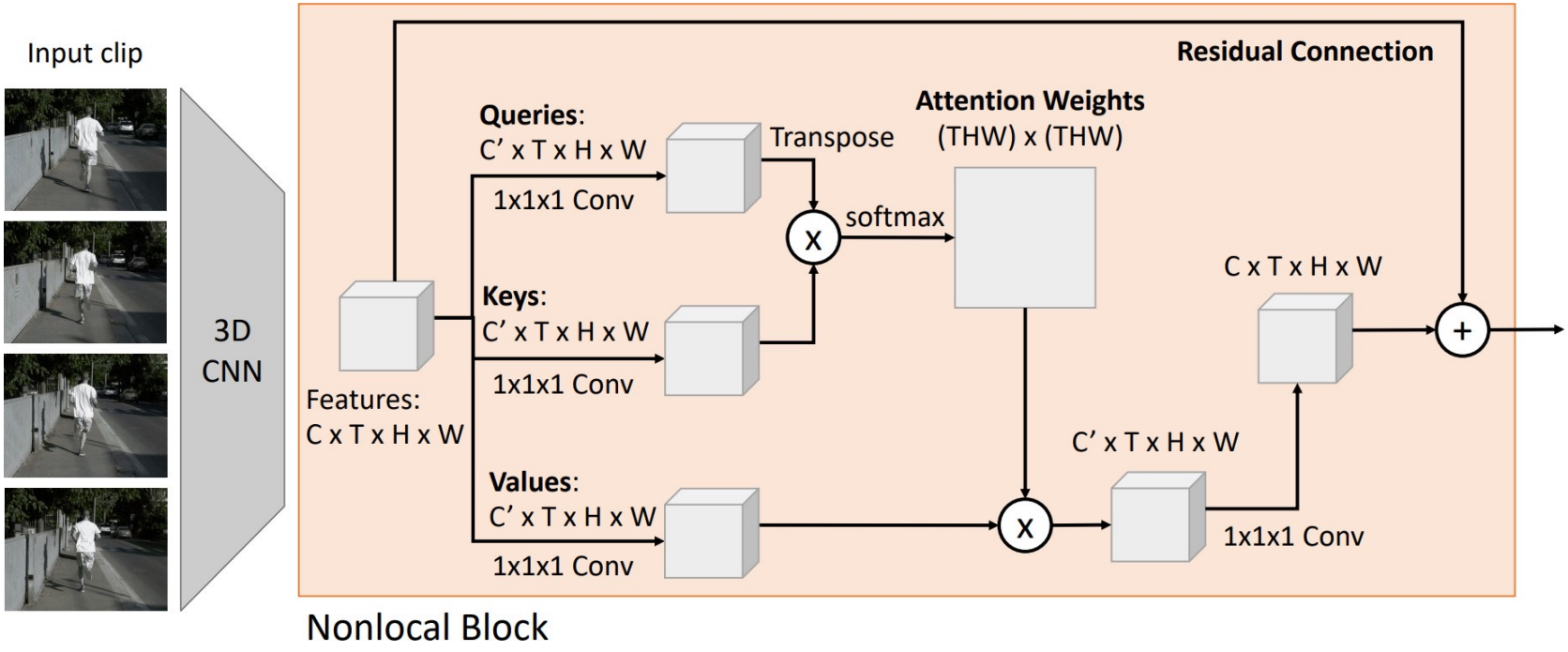
Modeling long-term temporal structure



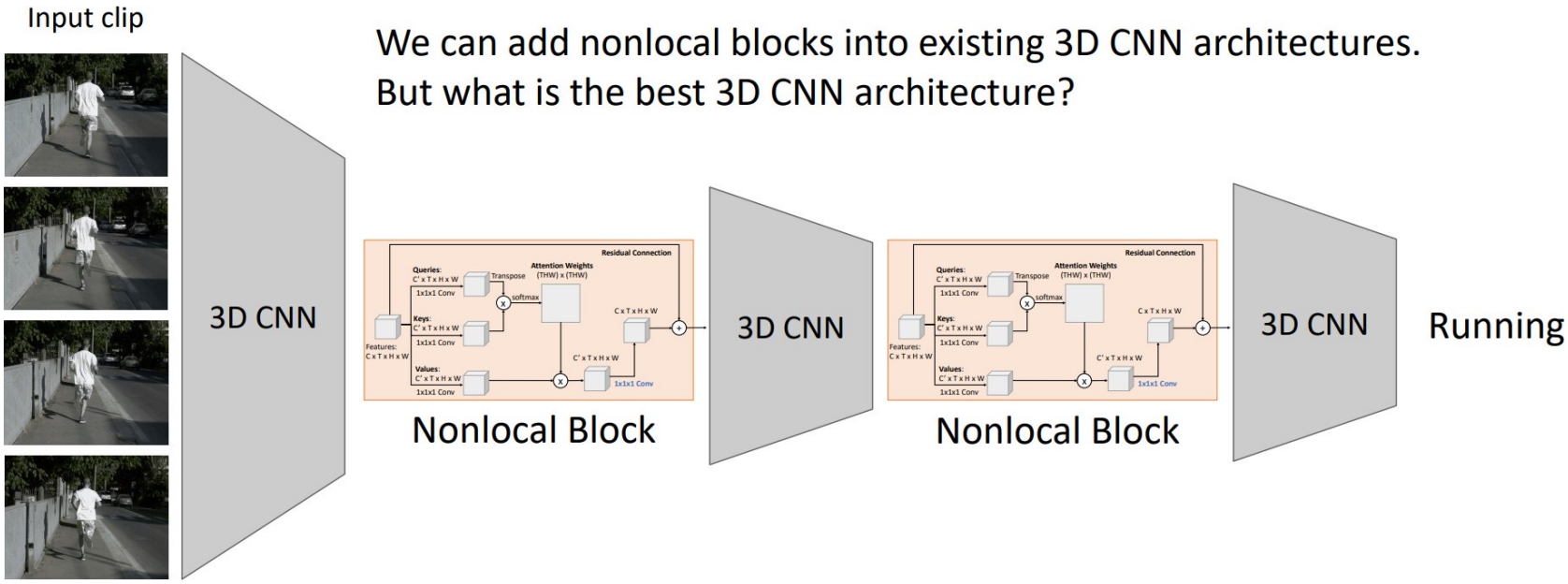
Modeling long-term temporal structure



Spatio-Temporal Self-Attention (Nonlocal Block)



Spatio-Temporal Self-Attention (Nonlocal Block)

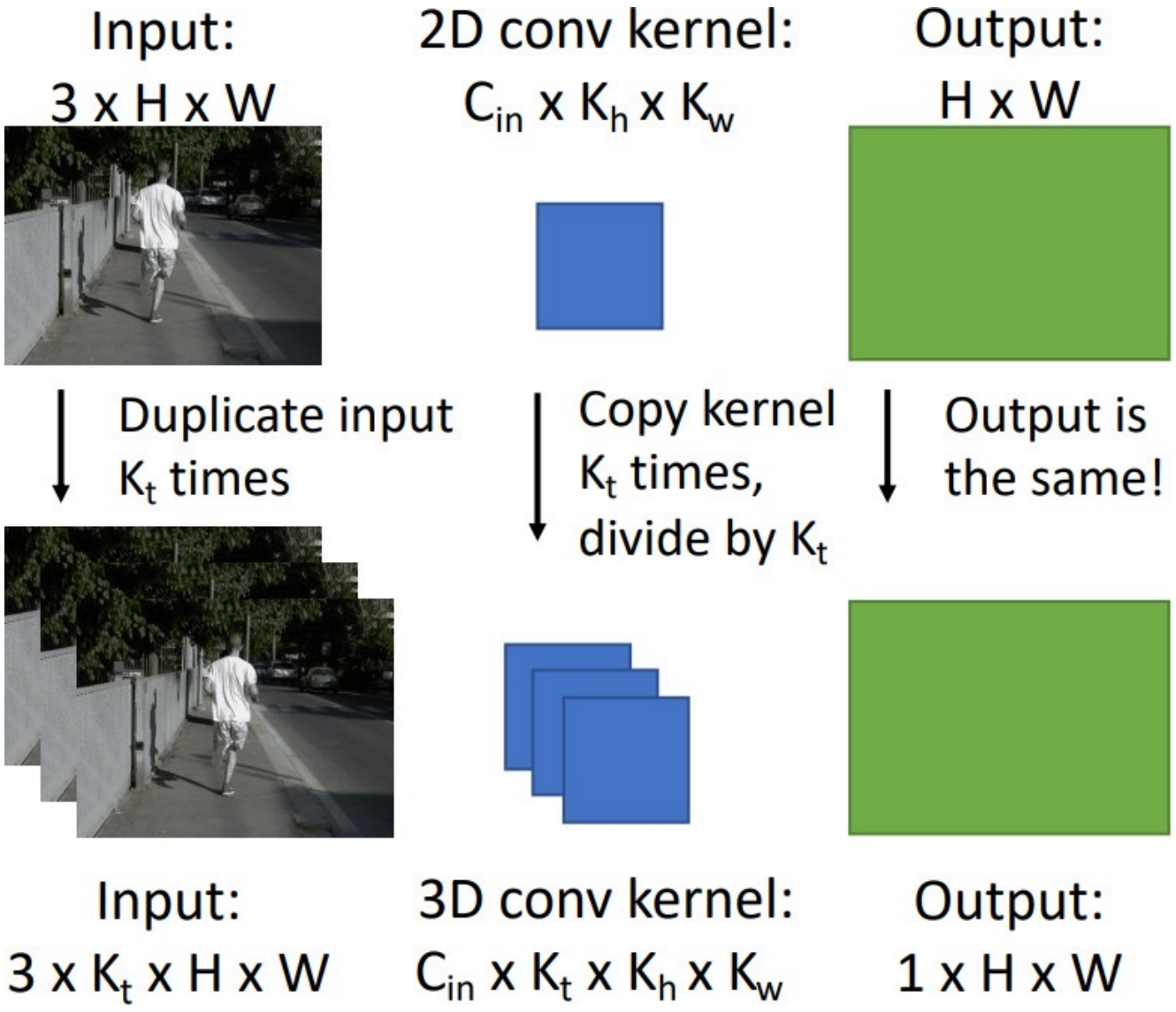


[0]Wang et al, “Non-local neural networks”, CVPR 2018

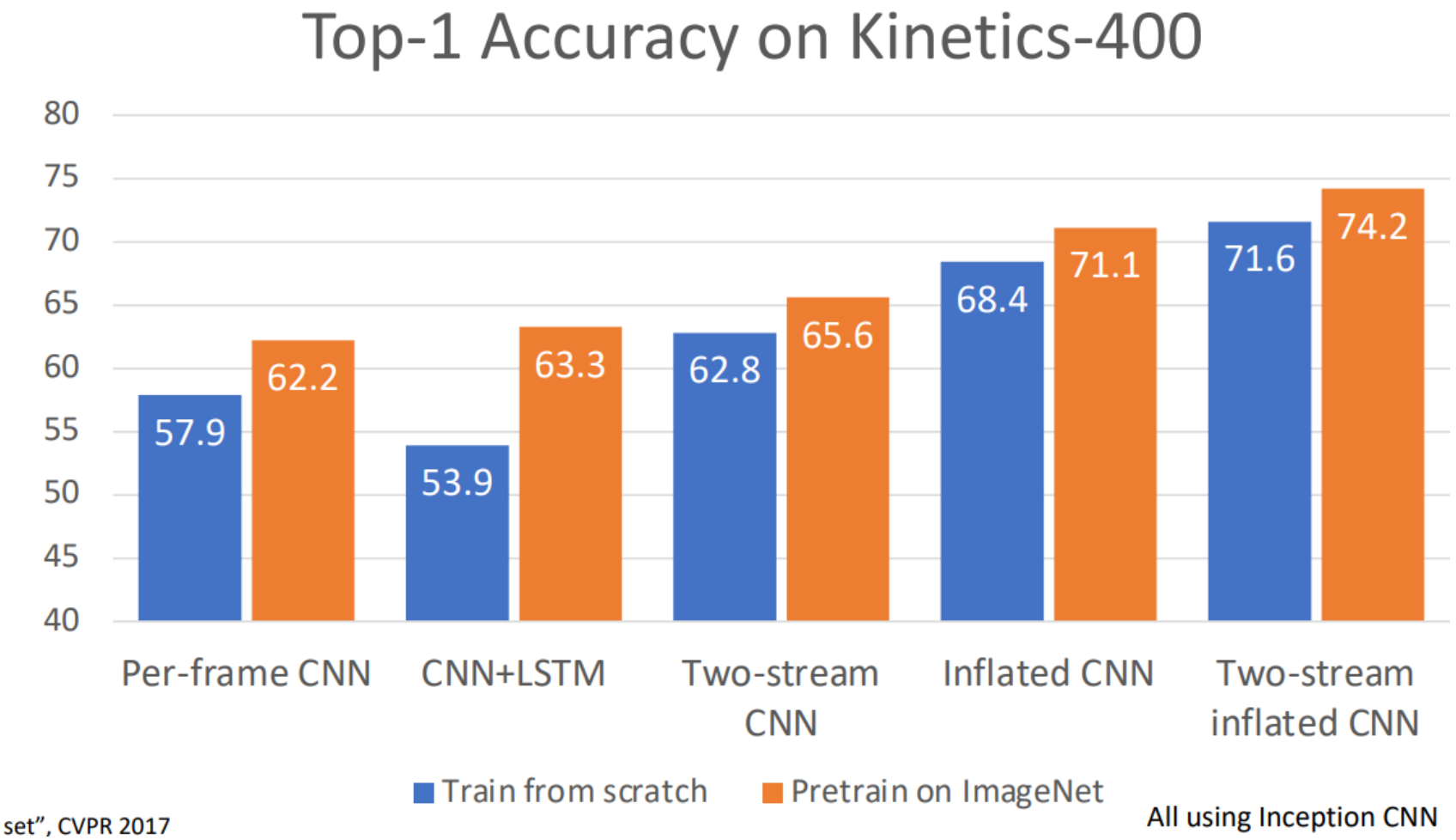
Inflating 2D Networks to 3D (I3D)

- There has been a lot of work on architectures for images.
- Can we reuse image architectures for video?
- **Idea:** take a 2D CNN architecture.
- Replace each 2D Kh x Kw conv/pool layer with a 3D Kt x Kh x Kw version
- Can use weights of 2D conv to initialize 3D conv: copy K_t times in space and divide by K_t
- This gives the same result as 2D conv given ”constant” video input

Inflating 2D Networks to 3D (I3D)



Inflating 2D Networks to 3D (I3D)

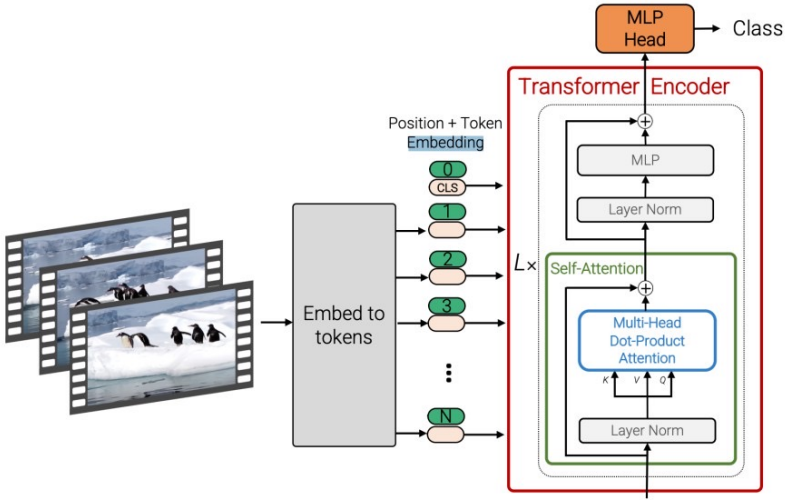


set", CVPR 2017

[0]Carreira and Zisserman, "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset", CVPR 2017

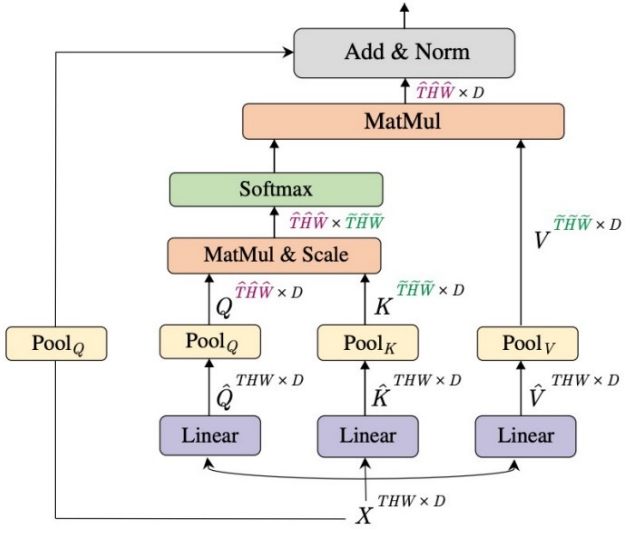
Vision Transformers for Video

Factorized attention: Attend over space / time



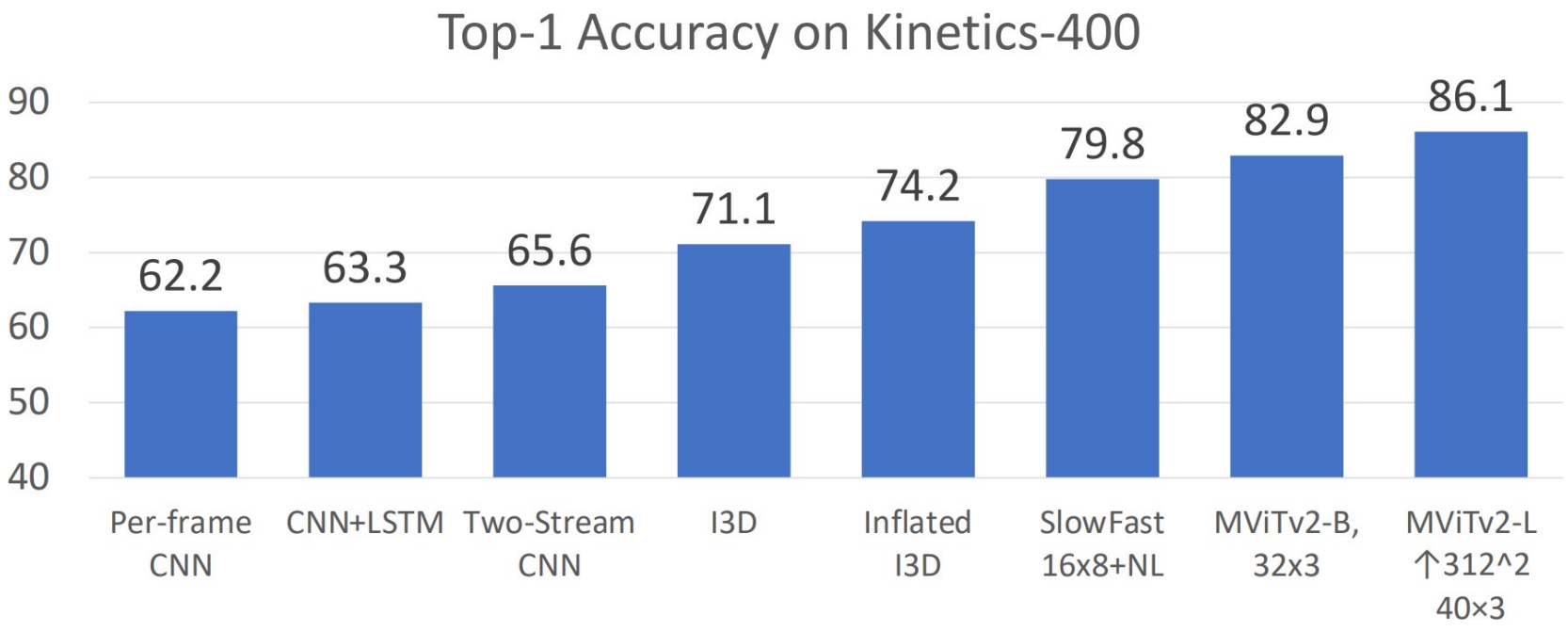
Bertasius et al, "Is Space-Time Attention All You Need for Video Understanding?", ICML 2021
Arnab et al, "ViViT: A Video Vision Transformer", ICCV 2021
Neimark et al, "Video Transformer Network", ICCV 2021

Pooling module: Reduce number of tokens



Fan et al, "Multiscale Vision Transformers", ICCV 2021
Li et al, "MViTv2: Improved Multiscale Vision Transformers for Classification and Detection", CVPR 2022

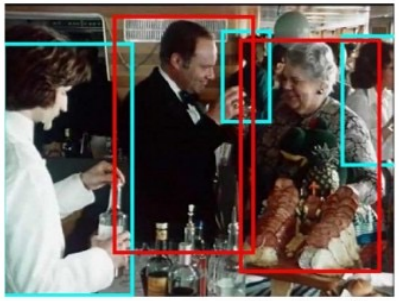
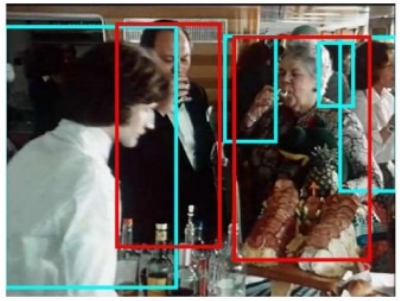
Vision Transformers for Video



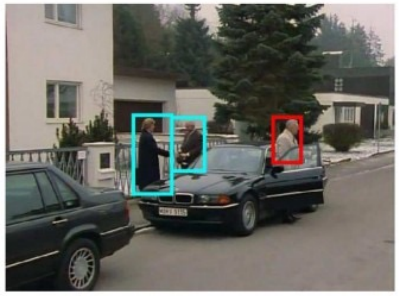
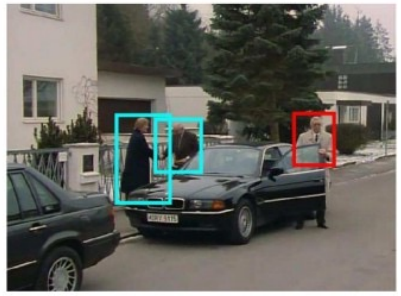
[0]Li et al, "MViTv2: Improved Multiscale Vision Transformers for Classification and Detection", CVPR 2022

Spatio-Temporal Detection

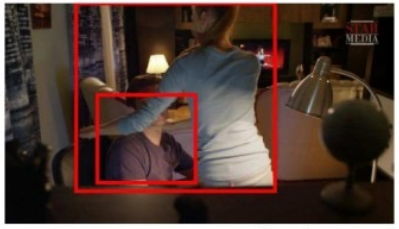
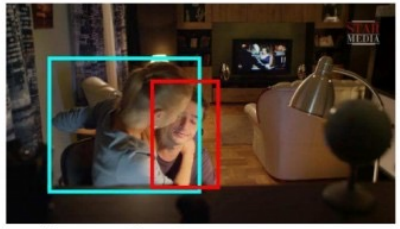
- Given a long untrimmed video, detect all the people in space and time and classify the activities they are performing
- Some examples from AVA Dataset:

clink glass → drink

open → close

grab (a person) → hug


look at phone → answer phone

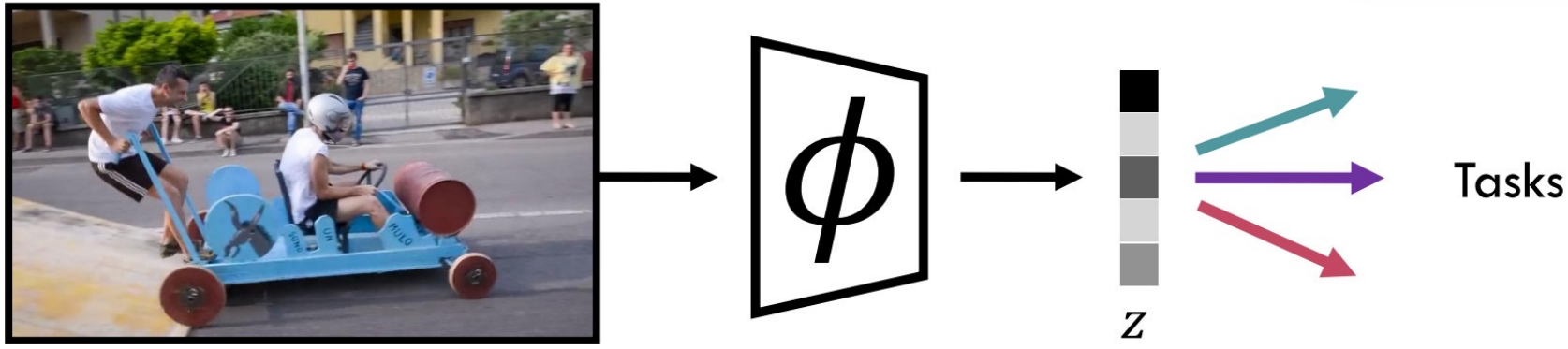
Ego4D: New Large-Scale Video Dataset

- 3670 hours of egocentric video (head-mounted cameras)
- Long videos: 1-10 hours each
- Diverse: data collected by 14 teams spread across 9 countries; 931 camera wearers (not just grad students!)
- Natural-language narrations (3.85M sentences)
- Support for 5 different tasks:
  - Episodic Memory
  - Hands and Objects
  - Audio-Video Diarization
  - Social Interactions
  - Forecasting

Ego4D: New Large-Scale Video Dataset

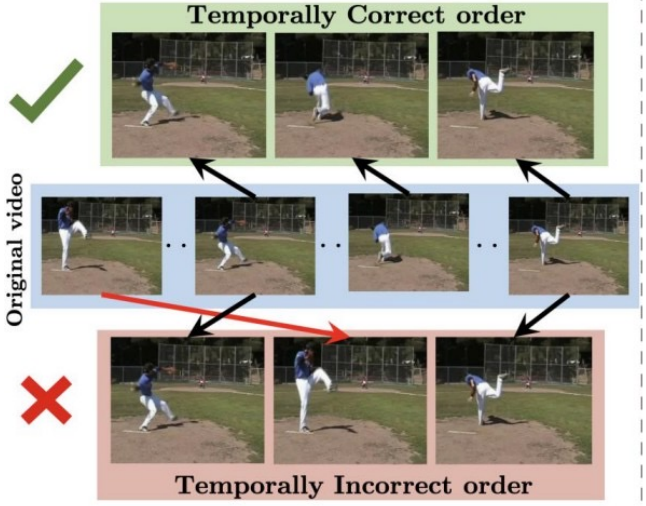
Self-Supervision in Videos

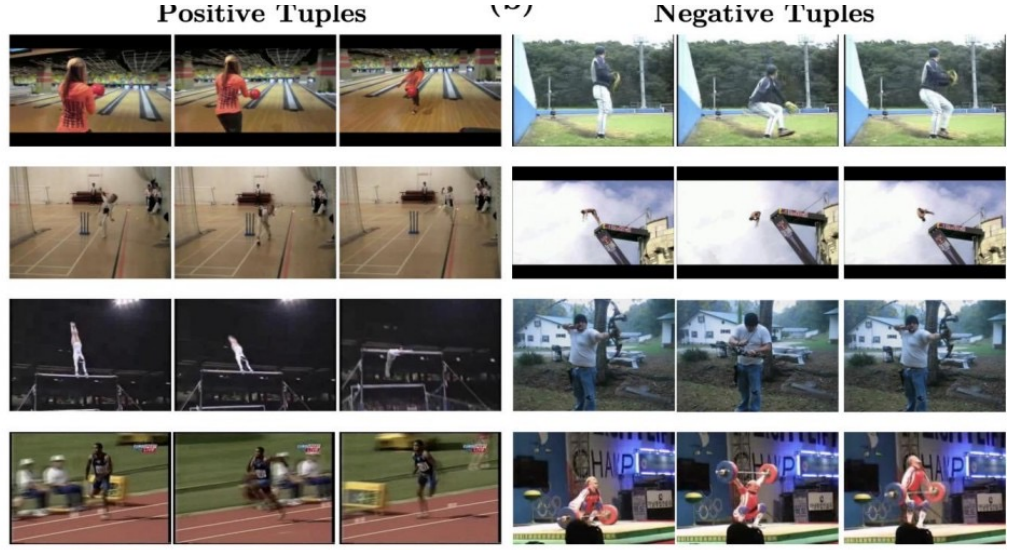
- Temporal order
- Cycle consistency
- Video Speedup
- Video colorization



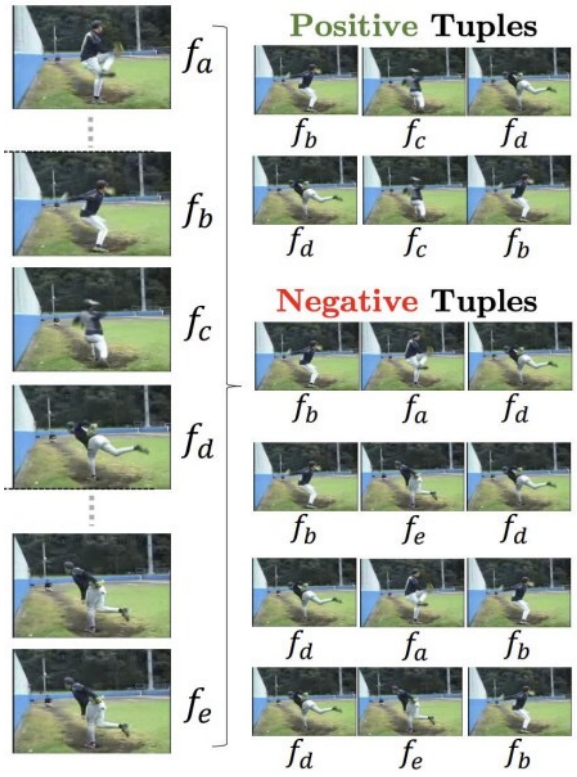
Frame Reordering

- **Training data:** Shuffled video frames, original video frames
- **Pretext task:** predict if the frames are in the correct temporal order (binary classification task)

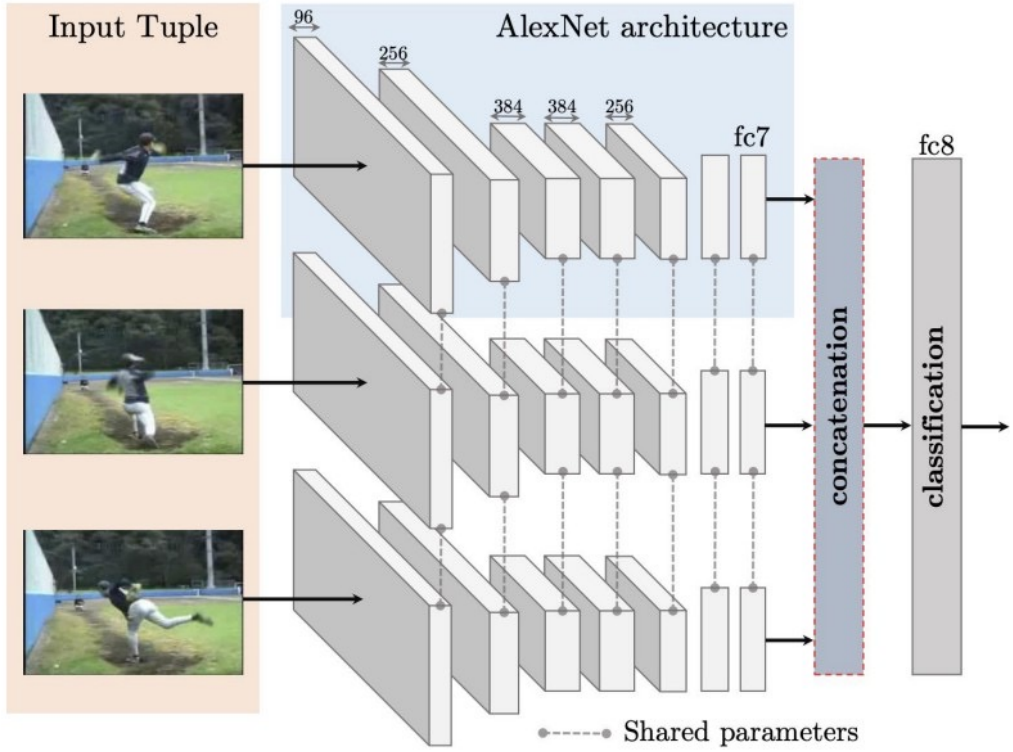




Frame Reordering



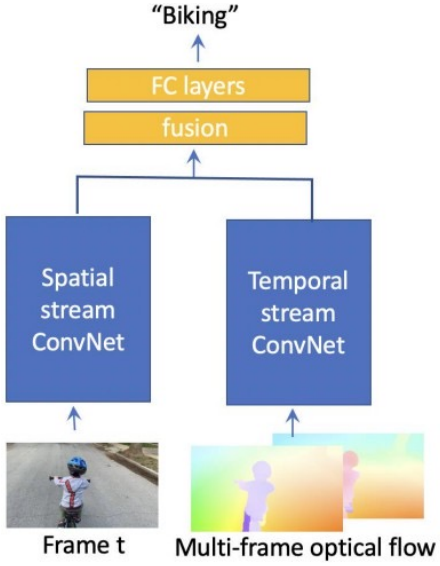
Generating positive and negative examples



Triplet Siamese network for sequence verification

Frame Reordering

- **Transfer learning:** Fine-tune spatial stream for video classification

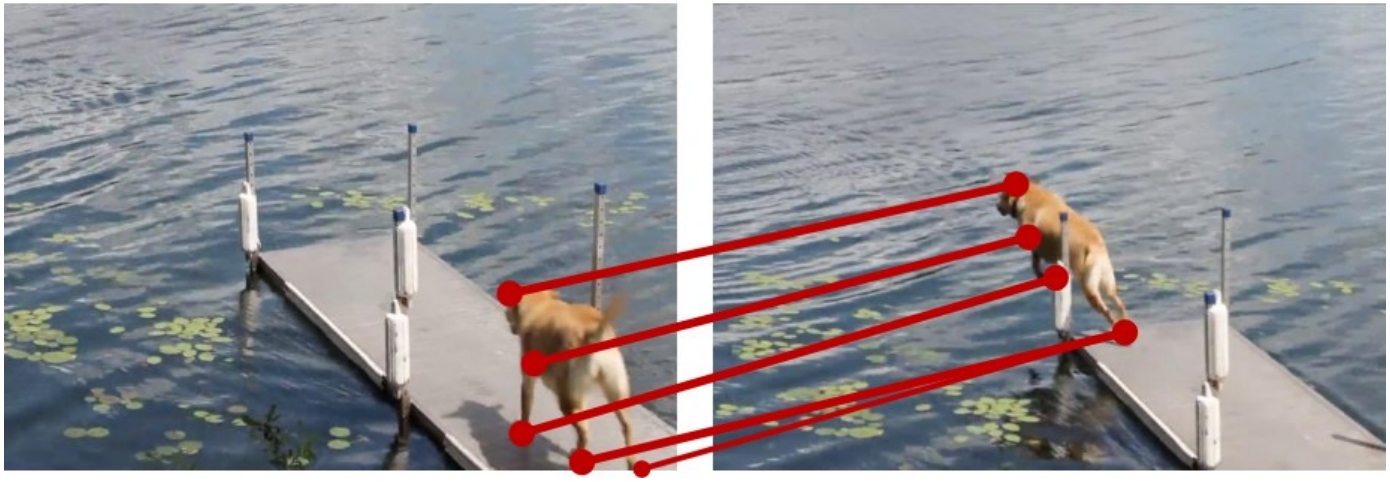


Dataset	Initialization	Mean Accuracy
UCF101	Random	38.6
	(Ours) Tuple verification	**50.2**
HMDB51	Random	13.3
	UCF Supervised	15.2
	(Ours) Tuple verification	**18.1**

[0]Misra et. al., Shuffle and Learn: Unsupervised Learning using Temporal Order Verification, ECCV 2016

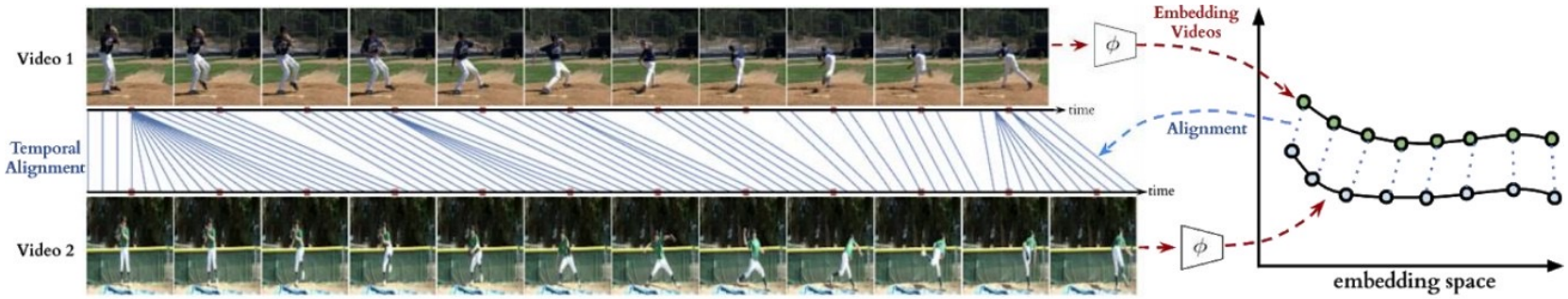
Learning correspondence

- **Ultimate Goal:** Correspondence

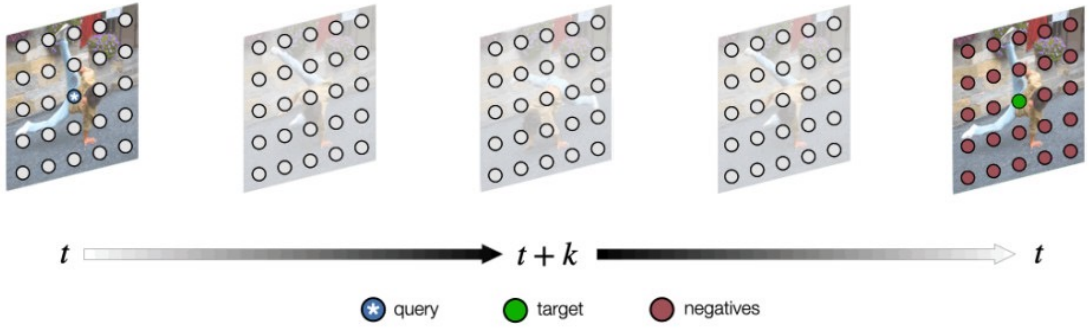


[0]Wang and Efros, Learning Correspondence from the Cycle-consistency of Time, CVPR 2019

Temporal cycle consistency



Dwibed et. al. Temporal Cycle-Consistency Learning, CVPR'19



Jabri et. al, Space time correspondence as Contrastive Random Walk, NeurIPS 2020

Learning the Speediness in Videos

- **Ultimate Goal:** Watch video content faster by adaptively speeding up the video

[0]Joint work with: Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, Bill Freeman, Miki Rubinstein and Michal Irani, CVPR 2020

References

These slides have been adapted from

- Fei-Fei Li, Yunzhu Li & Ruohan Gao, Stanford CS231n: Deep Learning for Computer Vision
- Assaf Shocher, Shai Bagon, Meirav Galun & Tali Dekel, WAIC DL4CV Deep Learning for Computer Vision: Fundamentals and Applications
- Justin Johnson, UMich EECS 498.008/598.008: Deep Learning for Computer Vision