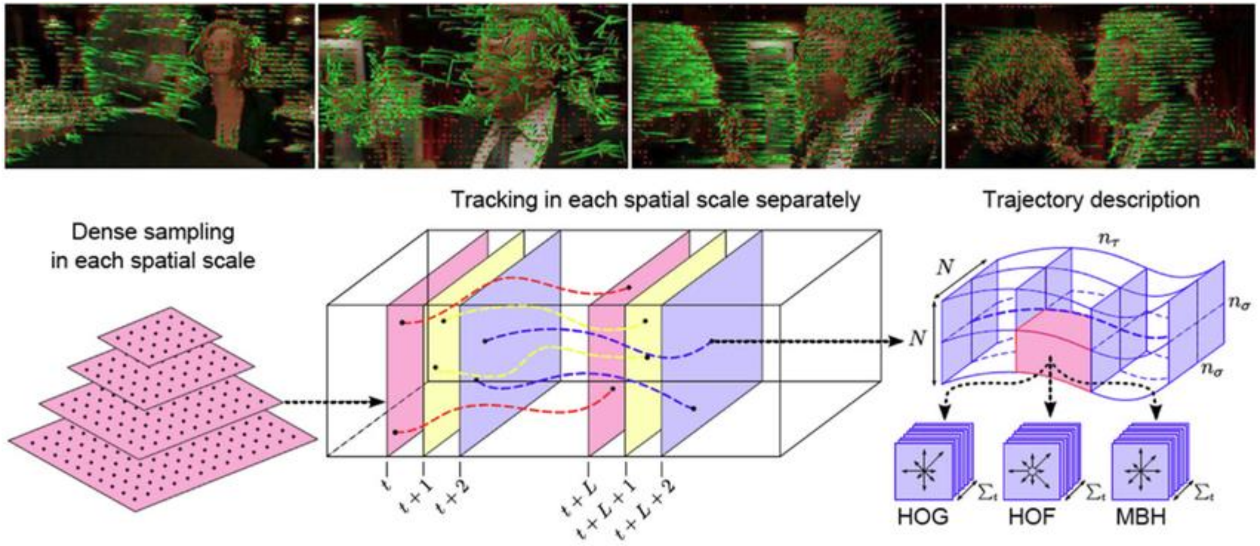
COMPRESSED-DOMAIN VIDEO CLASSIFICATION WITH DEEP NEURAL NETWORKS: “THERE’S WAY TOO MUCH INFORMATION TO DECODE THE MATRIX”

Aaron Chadha, Alhabib Abbas, Yiannis Andreopoulos

Background: Action recognition

- Action recognition: Image sequences -> Actions

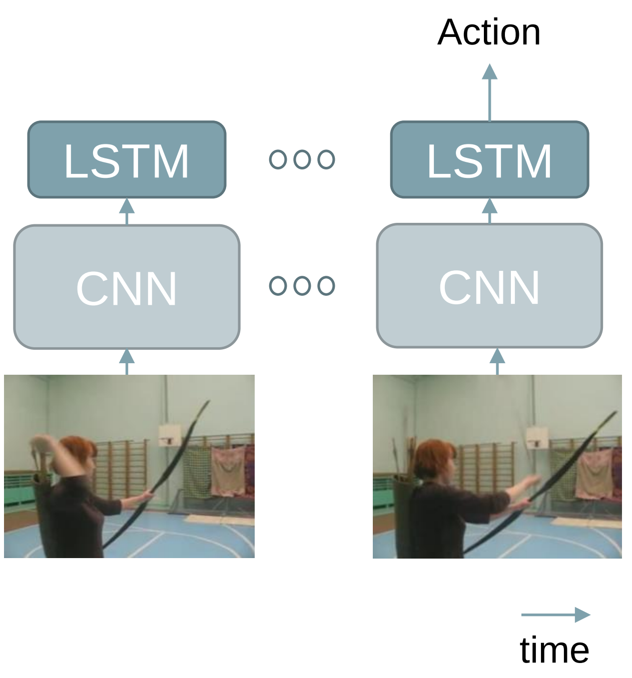
- Before deep learning – dense trajectories using optical flow:



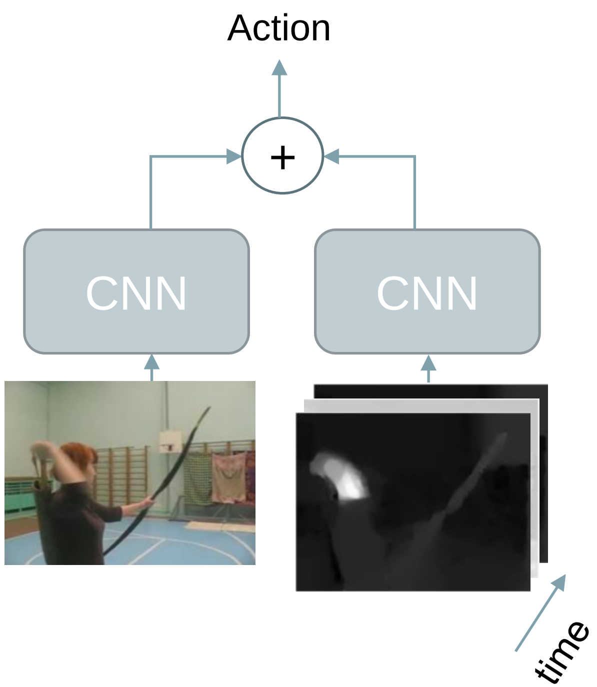
Background: Action Recognition

State-of-the-art deep learning methods:

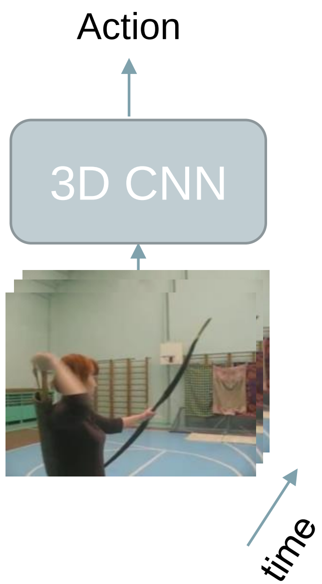
a) LSTM + CNN



b) Flow + RGB



c) 3D RGB CNN



Common Issues

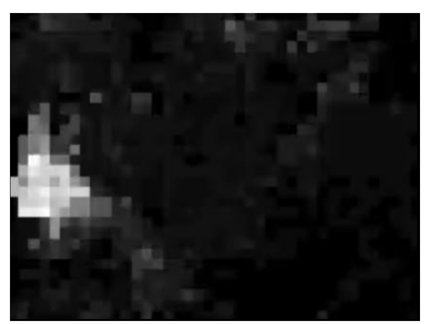
- RGB frame inputs require full video decoding
- Optical flow is expensive to compute (per-pixel)
- LSTMs are notoriously slow to train
- 3D CNN on (large) RGB frames requires heavy processing
- Short temporal extent of inputs -> only looking at local motion cues and not long term dependencies
- Redundancy between consecutive frames

Our Proposal

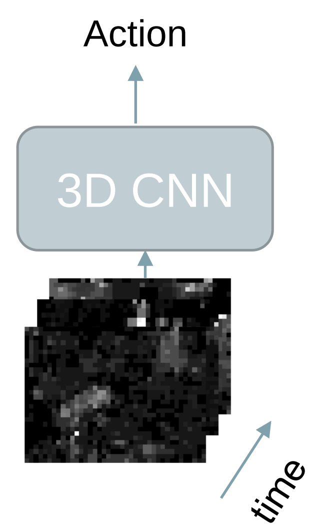
- We would like to use the video codec directly as a *spatio-temporal sensor*
- We propose a 3D convolutional neural network that directly ingests motion vector flow extracted from the encoder bitstream

Video

MV flow



Proposal

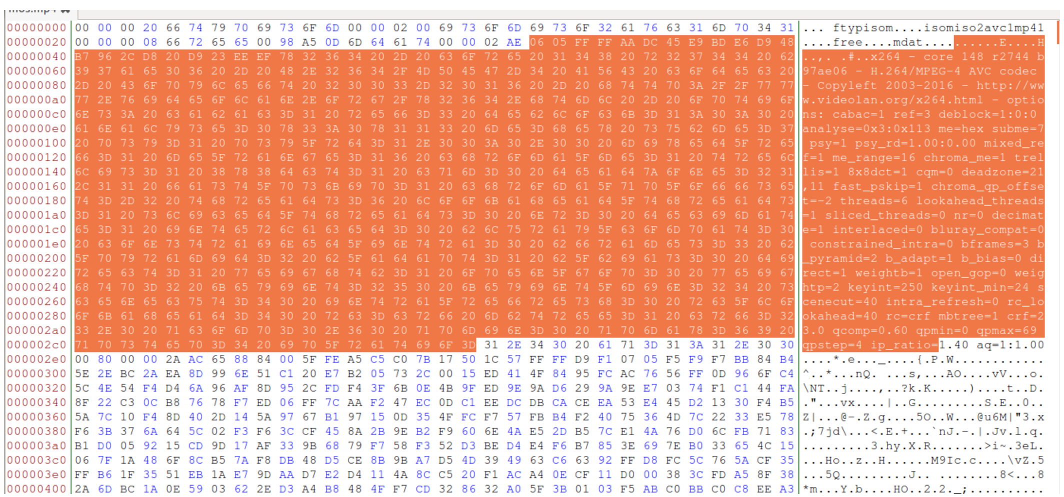


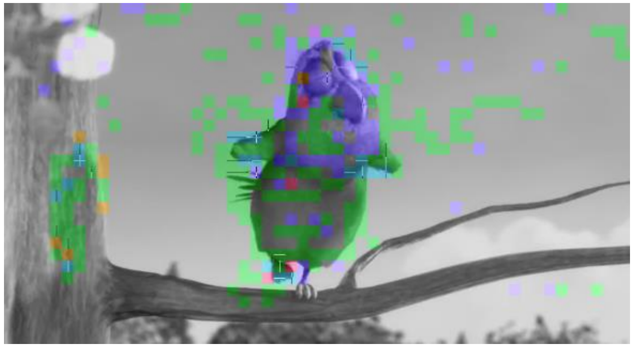
3D-CNN Input

- Macroblock (MB) bitstream representation from codec:

| ADDR | TYPE | QUANT | VECTOR | CBP | b0 | b1 | ... b5 |

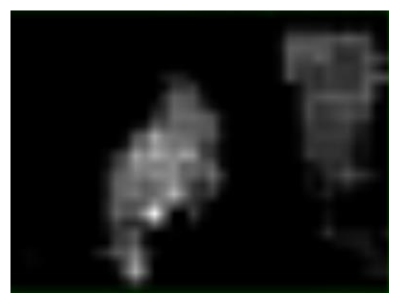
  - Addr: Block address in image
  - Type: Intra(I), inter (P), or bi-directional inter (B) frames
  - Vector: Motion vector





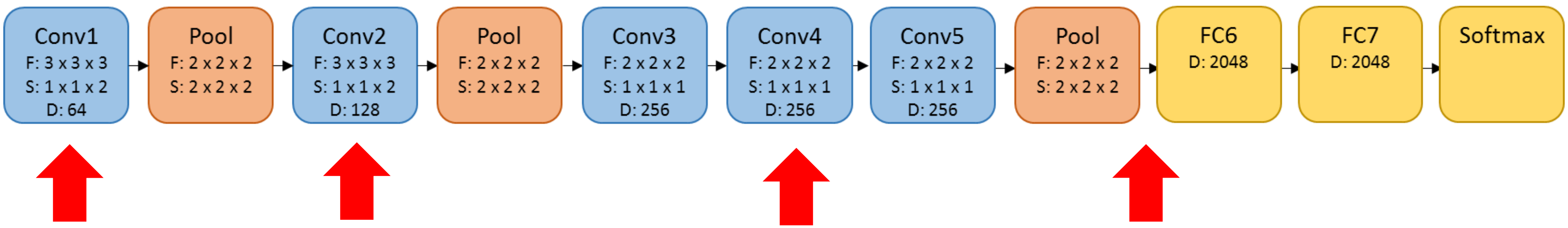
3D-CNN Input

- Only ingest P-frames
- 8 x 8 macroblock size (equates to 40 x 30 MV frame representations on UCF-101)
- 2 channels, δx and δy
- Low spatial resolution -> longer temporal extent
- CNN input is 4D: 24 x 24 x 2 x 160 resolution



3D-CNN Architecture

- 3D CNN (F = filter, S = stride, D = depth):



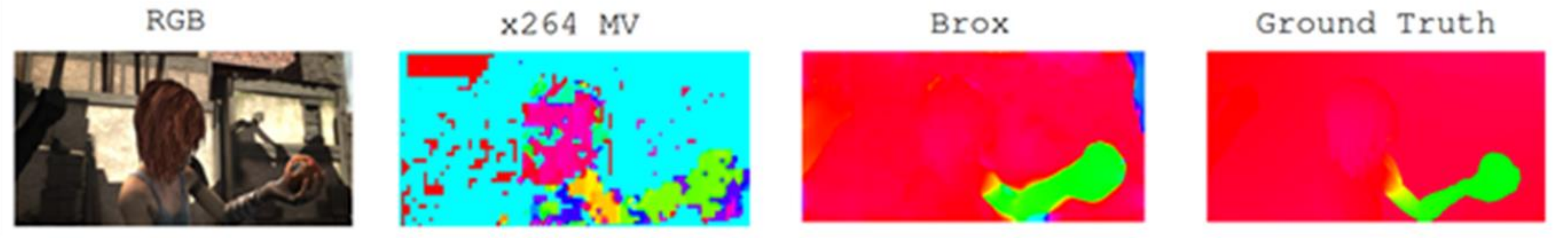
Temporal stride of 2 to quickly downsample input

2 x 2 x 2 conv filters to minimize weights, sufficiently large to cover spatial extent

3 x 3 x 5 input to fully connected layers

Experimental Evaluation

- Visual comparison of inputs (MPI-Sintel):



- Visual quality measured in terms of EPE (lower is better)

Input	Runtime per frame (ms) Decoding	Runtime per frame (ms) Flow Estimation	% P	% NZ	EPE
Proposed	0	0.16 (CPU)	62	21	15.26
Brox	3.08 (CPU)	6270 (GPU)	–	–	6.32
FlowNet2	3.08 (CPU)	123 (GPU)	–	–	3.14

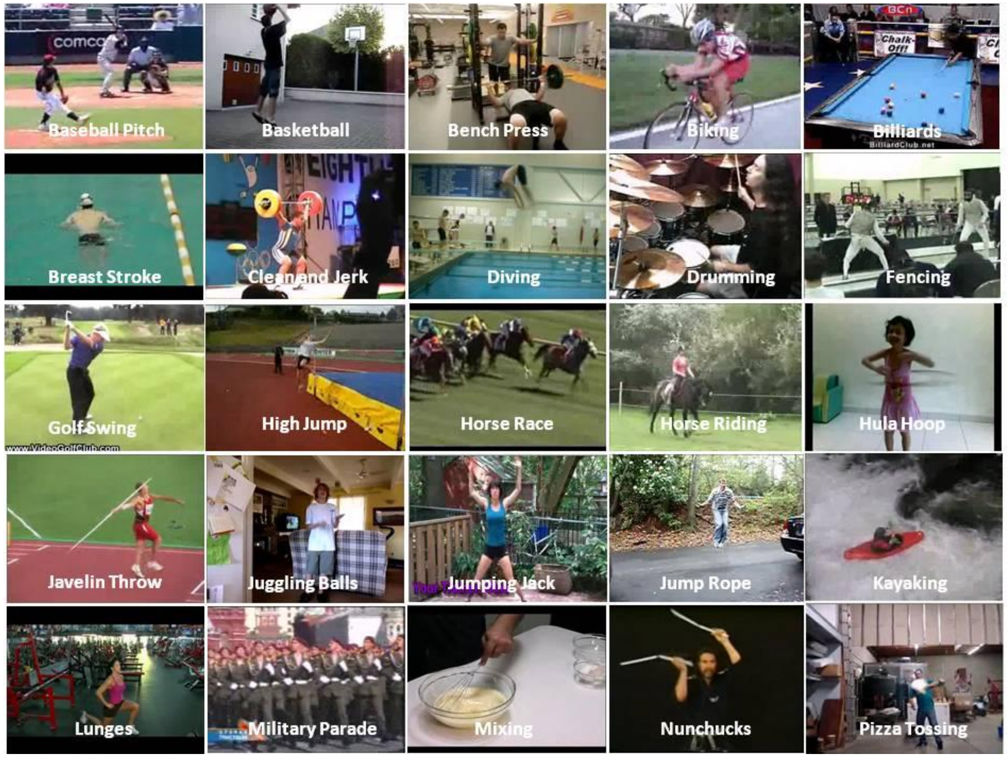
- Proposed MV flow is over 4 times faster than Brox to compute

Experimental Evaluation

- We measure classification performance on UCF-101 and HMDB-51 datasets:

UCF-101: 13k videos,
101 classes,
320 x 240 resolution,
25 FPS

HMDB-51: 7k videos,
51 classes,
320 x 240 resolution,
30 FPS



Experimental Evaluation

- Accuracy compared to state-of-the-art:

Framework	Input Size	Complexity #A, #W ($\times 10^6$)	Accuracy (%) UCF	HMDB
Proposed	$24^2 \times 2 \times 160$	4.0, 29.4	77.5	49.5
SSCNN-Brox	$224^2 \times 20$	2.0, 90.6	83.7	54.6
SSCNN+	$224^2 \times 3$	2.0, 90.6	73.0	40.5
LTC-Brox	$58^2 \times 2 \times 100$	42.1, 12.2	82.6	56.7
LTC-Mpegflow	$58^2 \times 2 \times 60$	25.3, 10.6	63.8*	–
SFCNN+	$170^2 \times 3 \times 10$	1.80 , 26.7	65.4	–
C3D+	$112^2 \times 3 \times 16$	30.2, 63.7	82.3	–

Ours:
- 3105 FPS
- 29.4M weights

Best acc.:
- 185 FPS
- 90.6M weights

Conclusions

- MV flow extraction (P-frames only) is up to 4 orders of magnitude faster than optical flow variants
- Low spatial resolution is counter-balanced by very long temporal extent (160 frames)
- We achieve competitive accuracy (77.5% on UCF-101) to methods using optical flow
- Lightweight 3D CNN = up to an order of magnitude faster processing than recent work

- Code available at:

https://github.com/mvcnn

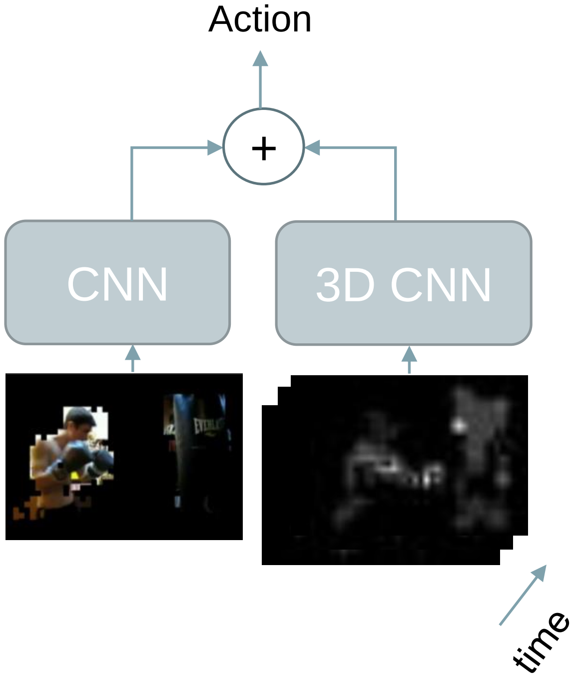
Further work

- We have since extended to a two-stream architecture using selectively decoded RGB frames:

Video

Selective decoding

Proposal



Further work

Framework	Accuracy (%)	
	UCF	HMDB
Proposed, $X = 10$	89.8	56.0
Proposed, $X = 50$	88.9	54.6
TSCNN (avg. fusion)	86.9	58.0
TSCNN (SVM fusion)	88.0	59.4
CNN-pool	88.2	–
C3D (3 nets)+IDT	90.4	–
LTC	91.7	64.8
EMV + RGB-CNN	86.4	–
IP+SVM	–	59.5
Line Pooling	88.9	62.2

End-to-end cost (ours):
$0.228 – $0.250

End-to-end cost (Zisserman et al.):
$11.103

Note: Cost is reported on evaluation for UCF-101, using AWS p2.xlarge and r3.xlarge instance prices as appropriate