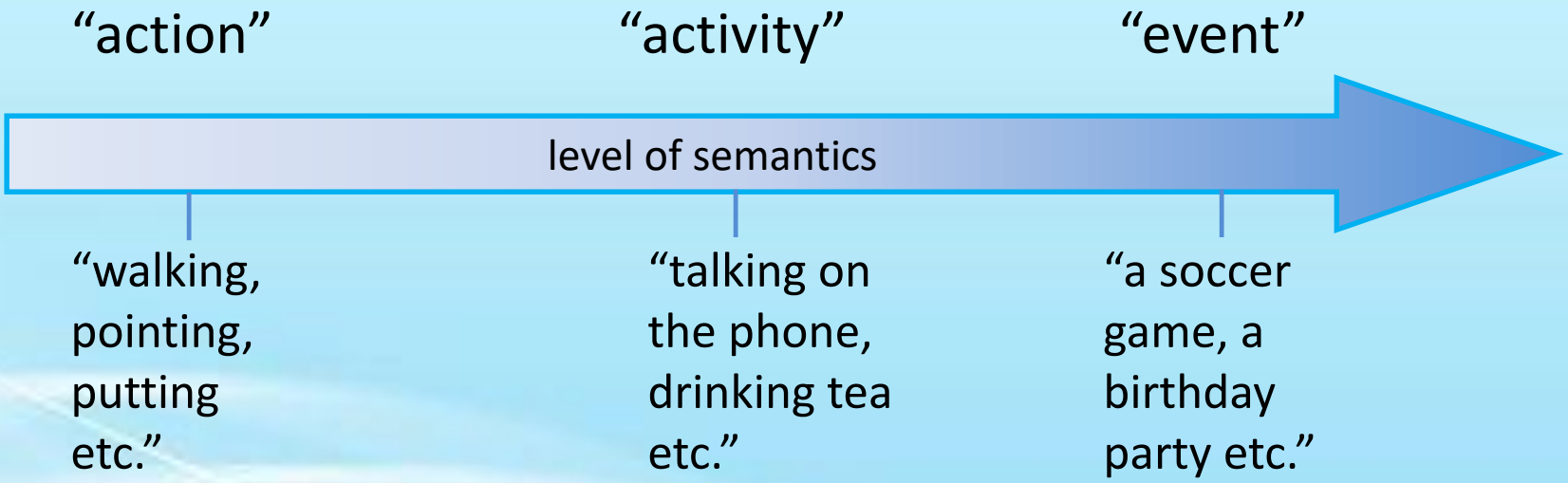
Action Recognition

The Bradley Department of Electrical and Computer Engineering

-Yingzhou Lu

What is action recognition?

- Input: video/image
- Output: the “action label”



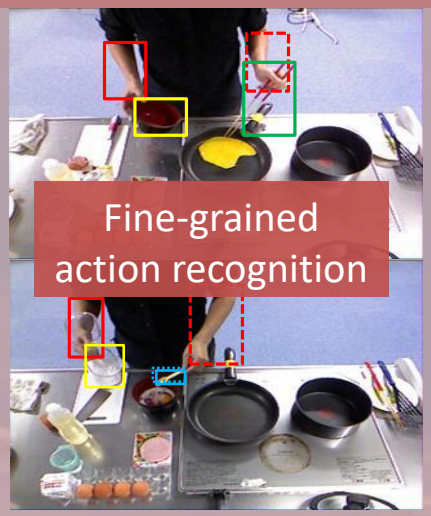
Why perform action recognition?

- Surveillance footage
- User-interfaces
- Automatic video organization / tagging
- Search-by-video?

Example Applications

Automatic Tele-Rehabilitation

Daily activity recognition



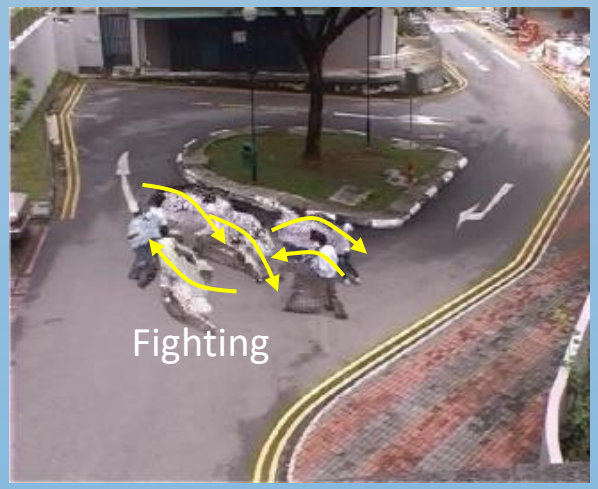
Fine-grained action recognition

Interactive action recognition

Intelligent Assisted Living and Home Monitoring

credit: Bingbing Ni

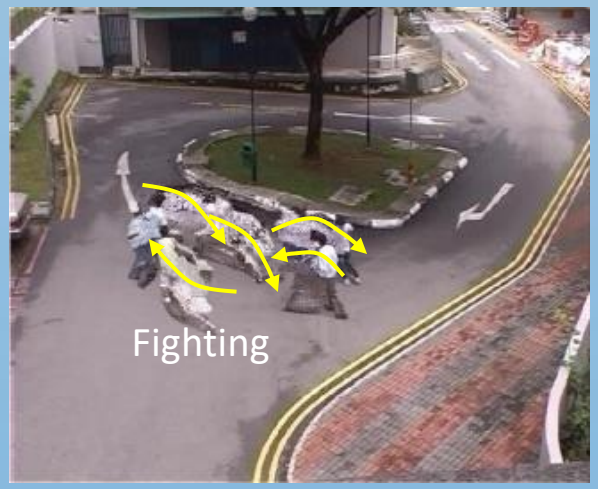
Example Applications

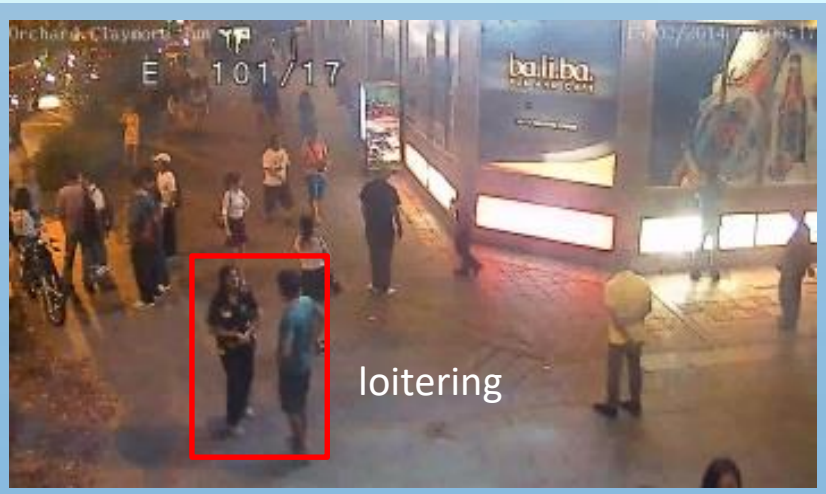


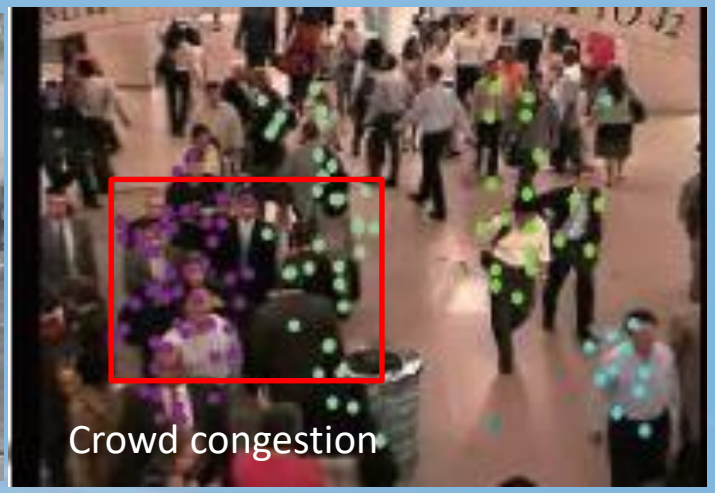
Crowd Behavior/Event Analysis

credit: Bingbing Ni

Example Applications







Crowd Behavior/Event Analysis

credit: Bingbing Ni

Demo

Why Action Recognition Is Challenging?

- Different scales
  - People may appear at different scales in different videos, yet perform the same action.
- Movement of the camera
  - The camera may be a handheld camera, and the person holding it can cause it to shake.
  - Camera may be mounted on something that moves.

Why Action Recognition Is Challenging?

- Occlusions
  - Action may not be fully visible

Figure from Ke et al.

Why Action Recognition Is Challenging?

- Background “clutter”
  - Other objects/humans present in the video frame.
- Human variation
  - Humans are of different sizes/shapes
- Action variation
  - Different people perform different actions in different ways.
- Etc…

Design good features for action representation

Example features

“space-time interest points”

“dense trajectories”

“motion history images”

“body joints”

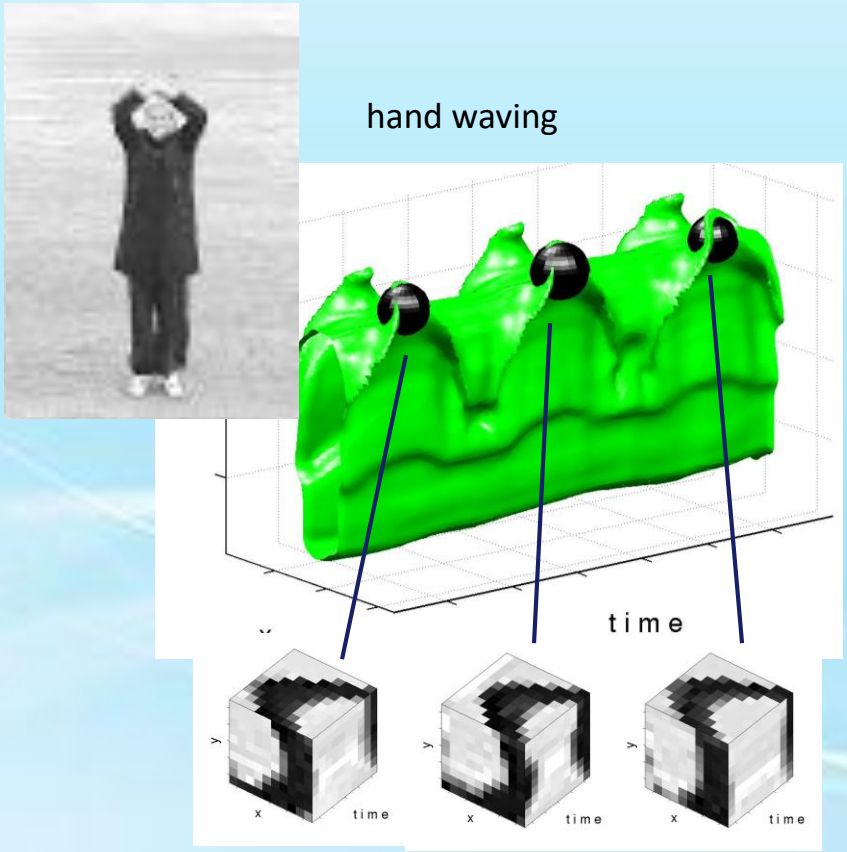
Paper Overview

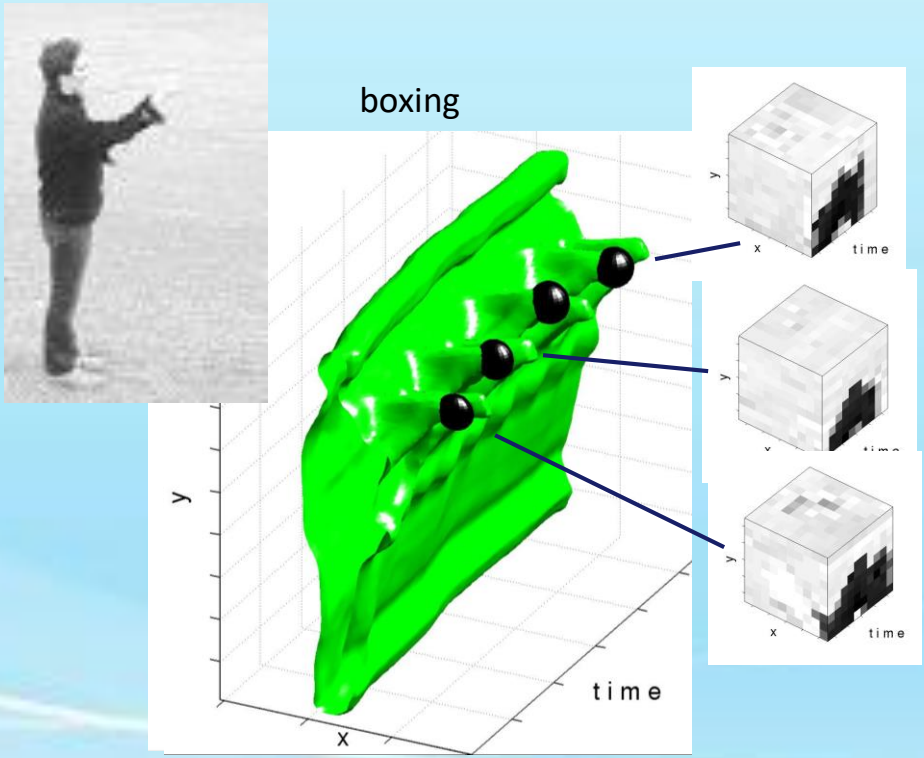
- Recognizing Human Actions: A Local SVM Approach - Christian Schuldt, Ivan Laptev and Barbara Caputo (ICPR 2004)
  - Use local space-time features to represent video sequences that contain actions.
  - Classification is done via an SVM. Results are also computed for KNN for comparison.

Space-time local features

No global assumptions $\Rightarrow$

Consider local spatio-temporal neighborhoods





Local Space-time features

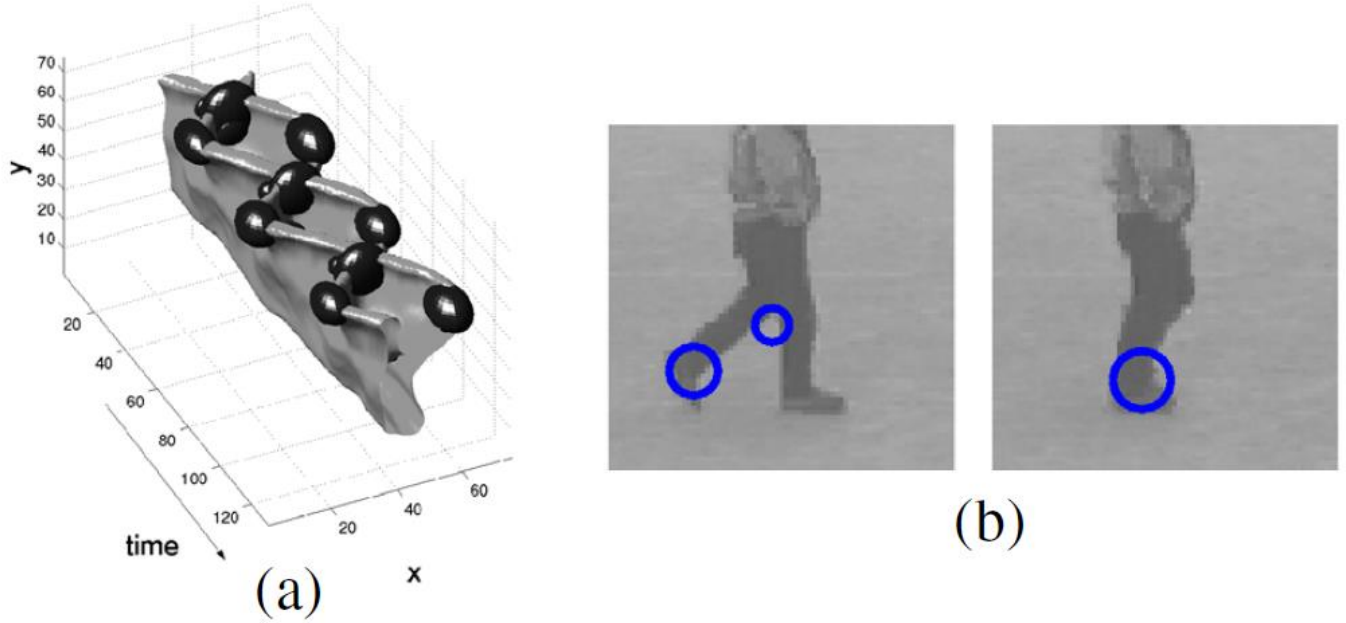


Figure 1. Local space-time features detected for a walking pattern: (a) 3-D plot of a spatio-temporal leg motion (up side down) and corresponding features (in black); (b) Features overlaid on selected frames of a sequence.

Figure from Schuldt et al.

Space-Time Interest Points: Detection

What neighborhoods to consider?

Distinctive neighborhoods $\Rightarrow$ High image variation in space and time $\Rightarrow$ Look at the distribution of the gradient

Definitions:

$f : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$ Original image sequence

$g(x, y, t; \Sigma)$ Space-time Gaussian with covariance $\Sigma \in \mathrm{SPSD}(3)$

$L_\xi(\cdot; \Sigma) = f(\cdot) * g_\xi(\cdot; \Sigma)$ Gaussian derivative of f

$\nabla L = (L_x, L_y, L_t)^T$ Space-time gradient

$$\mu(\cdot; \Sigma) = \nabla L(\cdot; \Sigma)(\nabla L(\cdot; \Sigma))^T * g(\cdot; s\Sigma) = \begin{pmatrix} \mu_{xx} & \mu_{xy} & \mu_{xt} \\ \mu_{xy} & \mu_{yy} & \mu_{yt} \\ \mu_{xt} & \mu_{yt} & \mu_{tt} \end{pmatrix}$$

Second-moment matrix

Representation of Features

- construct its scale-space representation using Gaussian convolution kernel g

$$L(\cdot, \sigma^2, \tau^2) = f * g(\cdot, \sigma^2, \tau^2)$$

- compute the second-moment matrix using spatio-temporal image gradients

$$\mu(\cdot; \sigma^2, \tau^2) = g(\cdot; s\sigma^2, s\tau^2) * \left(\nabla L (\nabla L)^T\right)$$

- define positions of features by local maxima

$$H = \det(\mu) - k \operatorname{trace}^3(\mu)$$

Representation of Features

- Spatial-temporal "jets" (4th order) are computed at each feature center:

$$l = (L_x, L_y, L_t, L_{xx}, ..., L_{tttt})$$

- Using k-means clustering, a vocabulary consisting of words h_i is created from the jet descriptors.
- Finally, a given video is represented by a histogram of counts of occurrences of features corresponding to h_i in that video: $H = (h_1, ..., h_n)$

Recognition Methods

- 2 representations of data:
  - [1] Raw jet descriptors (LF) (“local feature” kernel)
    - Wallraven, Caputo, and Graf (2003)
  - [2] Histograms (HistLF) (X^2 kernel)
- 2 classification methods:
  - SVM
  - K Nearest Neighbor

Design good classifiers

Example: Support Vector Machine (SVM) classifier

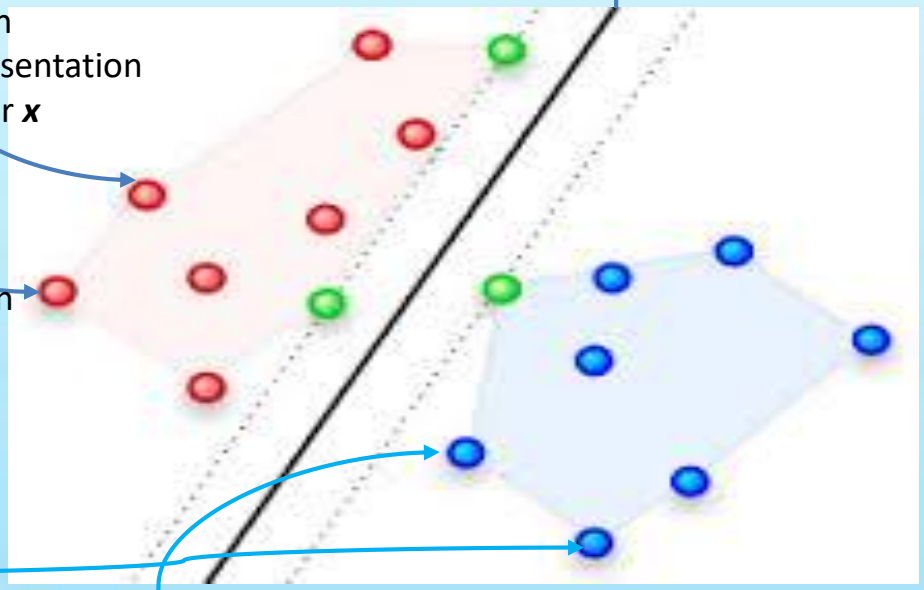
positive samples “walking”

action representation vector ***x***

negative samples “not walking”

action representation vector ***x***

w: separating hyper-plane



Linear SVM: $y = \boldsymbol{w}^T\boldsymbol{x} + b$

y>0 : “walking”

y<0 : “not walking”

The Dataset

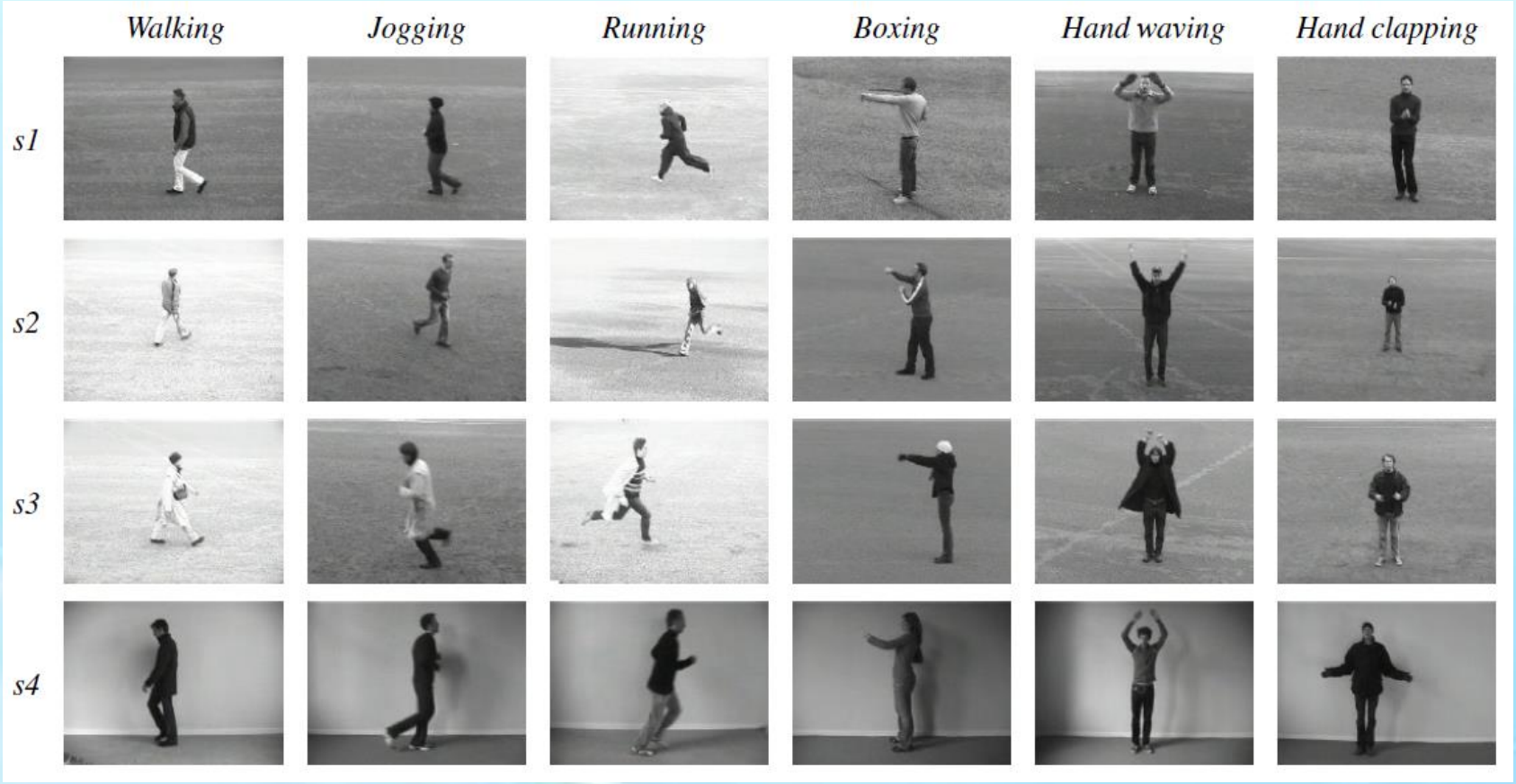


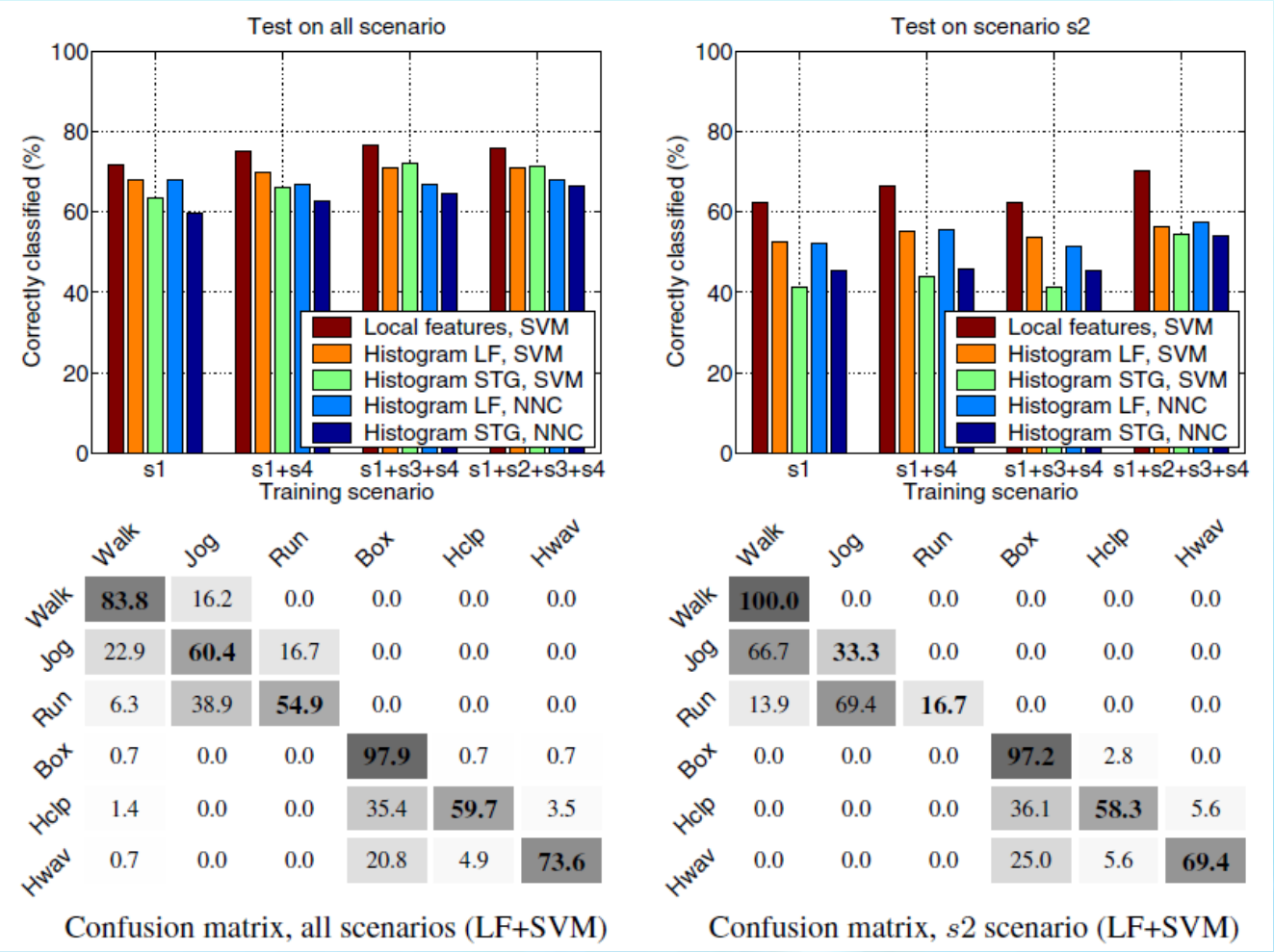
Figure from Schuldt et al.

The Dataset

- Video dataset with a few thousand instances.
  - 25 people each:
    - perform 6 different actions
      - Walking, jogging, running, boxing, hand waving, clapping
    - in 4 different scenarios
      - Outdoors, outdoors w/scale variation, outdoors w/different clothes, indoors
    - (several times)
- Backgrounds are mostly free of clutter.
- Only one person performing a single action per video.

credit: Benzaid

Results



Confusion matrix, all scenarios (LF+SVM)

Confusion matrix, $s2$ scenario (LF+SVM)

Figure from Schuldt et al.

Results

- Experimental results:
  - Local Feature (*jet*s) + SVM performs the best
  - SVM outperforms NN
  - HistLF (histogram of *jet*s) is slightly better than HistSTG (histogram of spatio-temporal gradients)

- Average classification accuracy on all scenarios = 71.72%

Analysis

- Some categories can be confused with others (running vs. jogging vs. walking / waving vs. boxing) due to different ways that people perform these tasks.

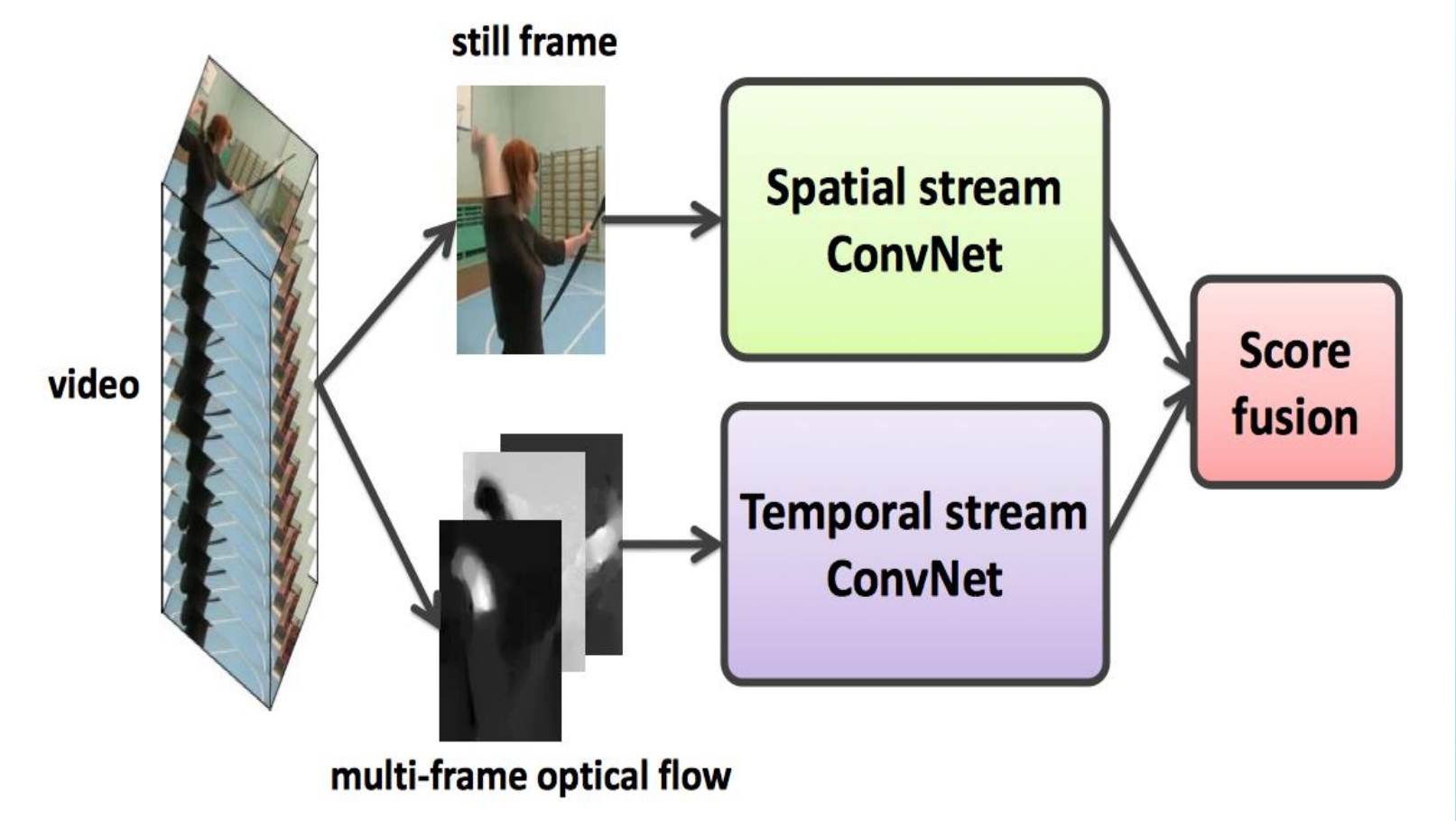
- Local Features (raw jet descriptors without histograms) combined with SVMs was the best-performing technique for all tested scenarios.

Paper Overview

- Two-Stream Convolutional Networks for Action Recognition in Videos (NIPS 2014)

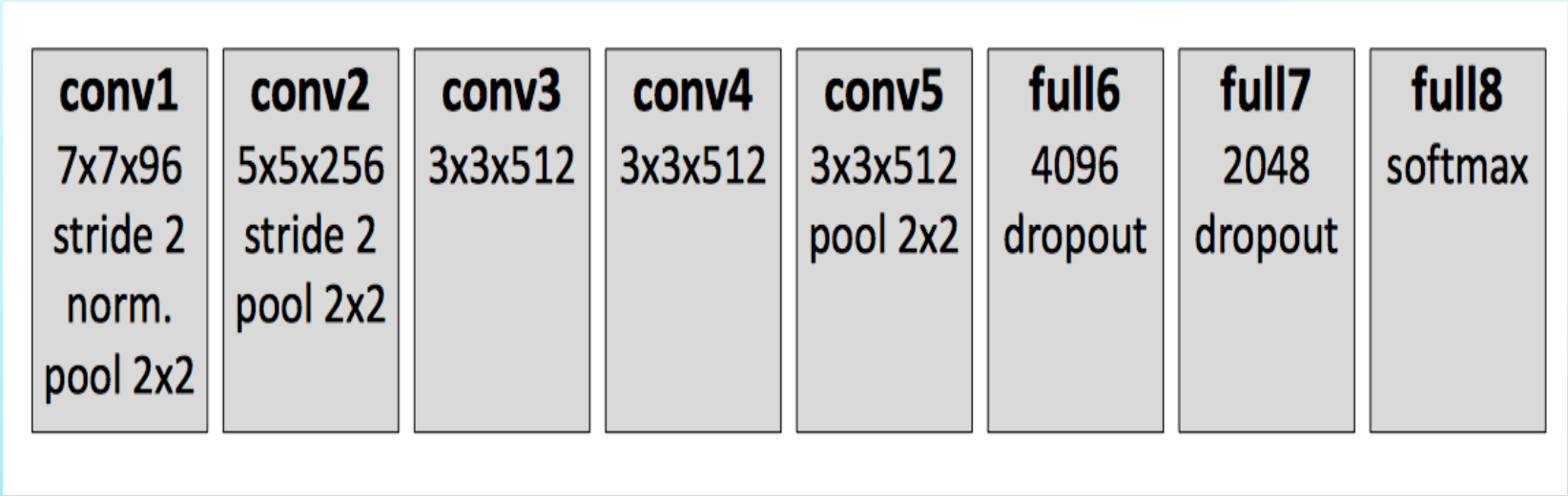
- propose a two-stream ConvNet architecture Two-stream architecture for video classification
  - Temporal stream – motion recognition ConvNet
  - Spatial stream – appearance recognition ConvNet

Architecture



Simonyan and Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos

Convnet Layer Conviguration



- 8 weight layers (5 convolutional and 3 fully-connected)
- used for both spatial & temporal streams

Simonyan and Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos

Spatial Stream

Predicts action from still images – image classification

- Input

  Individual RGB frames

- Training

  Leverages large amounts of outside image data by pre-training on ILSVRC (1.2M images, 1000 classes)

  Classification layer re-trained on video frames

- Evaluation

  Applied to 25 evenly sampled frames in each clip

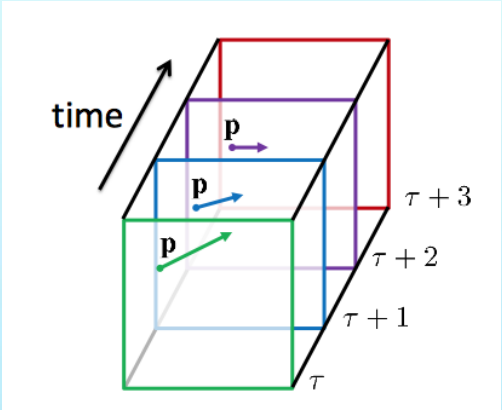
  Resulting scores averaged

Simonyan and Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos

Optical Flow

- Displacement vector field between a pair of consecutive frames
- Each flow – 2 channels: horizontal & vertical components
- Computed using [Brox et al., ECCV 2004]
  - based on generic assumptions of constancy and smoothness
  - pre-computed on GPU (17fps), JPEG-compressed
- Global (camera) motion compensated by mean flow subtraction

Temporal Stream



Predicts action from motion

Input

- Explicitly describes motion in video
- Stacked optical flow over several frames

Training

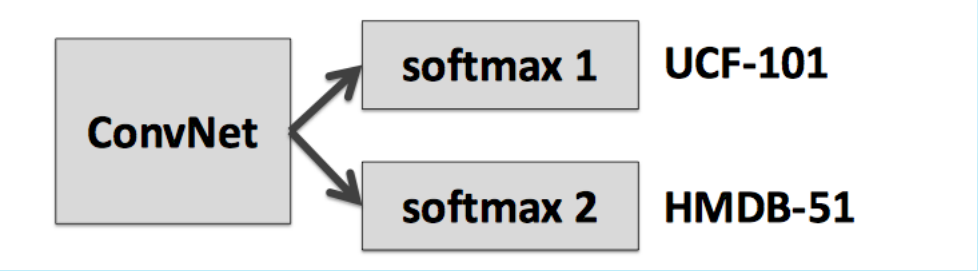
- From scratch with high drop-out (90%)

Multi-task learning to reduce over-fitting

- Video datasets (UCF-101, HMDB-51) are small
- Merging datasets is problematic due to semantic overlap
- Multi-task learning: each dataset defines a separate task (loss)

Simonyan and Zisserman, Two-Stream Convolutional Networks for Action Recognition in Videos

Evaluation



Video action classification datasets

- UCF-101 (101 class, 13K videos)
- HMDB-51 (51 class, 6.8K videos)

Results

Table 3: **Two-stream ConvNet accuracy on UCF-101 (split 1).**

Spatial ConvNet	Temporal ConvNet	Fusion Method	Accuracy
Pre-trained + last layer	bi-directional	averaging	85.6%
Pre-trained + last layer	uni-directional	averaging	85.9%
Pre-trained + last layer	uni-directional, multi-task	averaging	86.2%
Pre-trained + last layer	uni-directional, multi-task	SVM	**87.0%**

Table 4: **Mean accuracy (over three splits) on UCF-101 and HMDB-51.**

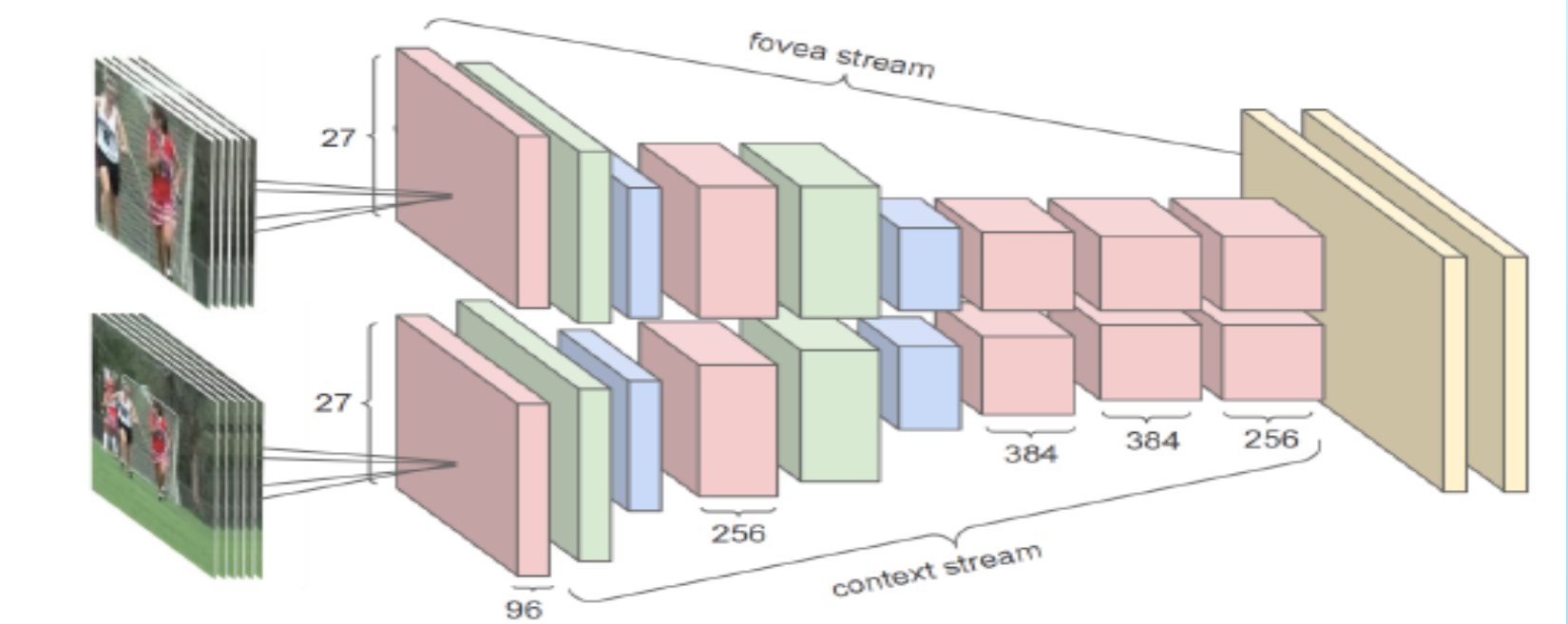
Method	UCF-101	HMDB-51
Improved dense trajectories (IDT) [26, 27]	85.9%	57.2%
IDT with higher-dimensional encodings [20]	**87.9%**	61.1%
IDT with stacked Fisher encoding [21] (based on Deep Fisher Net [23])	-	**66.8%**
Spatio-temporal HMAX network [11, 16]	-	22.8%
"Slow fusion" spatio-temporal ConvNet [14]	65.4%	-
Spatial stream ConvNet	73.0%	40.5%
Temporal stream ConvNet	83.7%	54.6%
Two-stream model (fusion by averaging)	86.9%	58.0%
Two-stream model (fusion by SVM)	**88.0%**	**59.4%**

Conclusions

- From depth and color image sequences to multi-modal visual analytics
- Quality metrics for activity recognition tasks
- New features for depth images and for fusion
- Machine learning framework
- New applications

Next Milestone in Action Recognition?

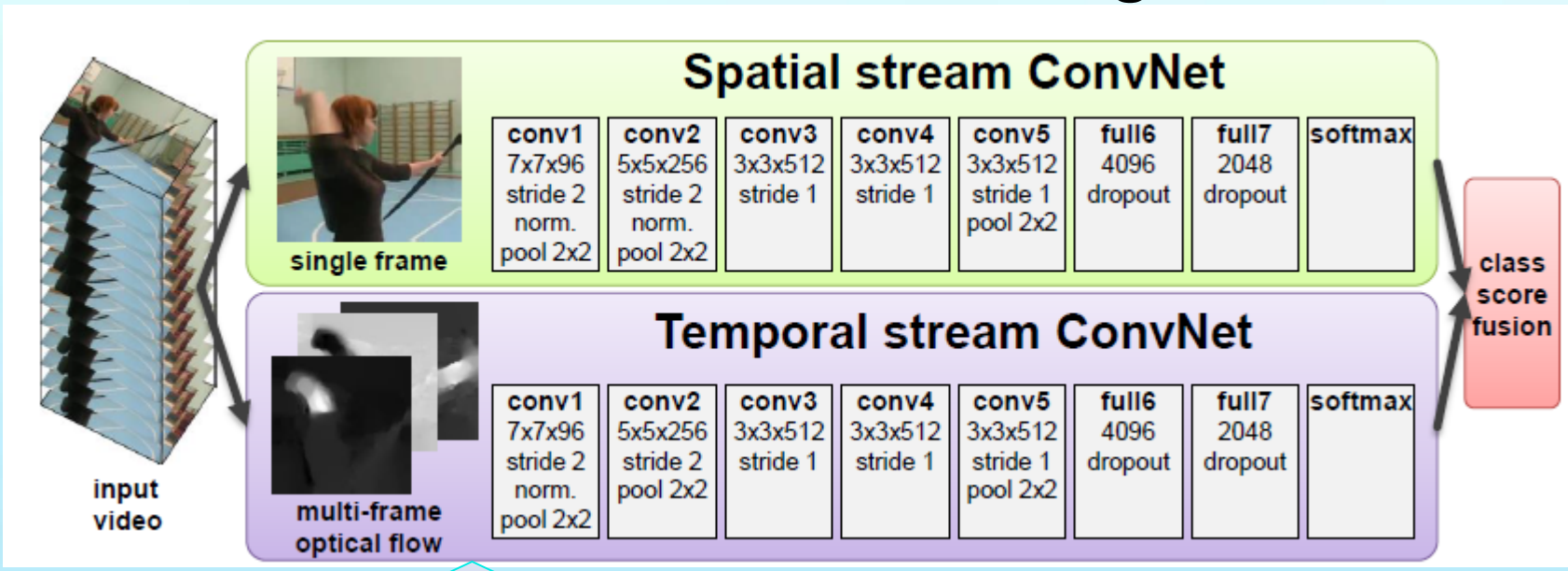
- Will CNNs take over existing action recognition methods?
  - -What's a good network architecture?
  - -Where to get sufficient training data?

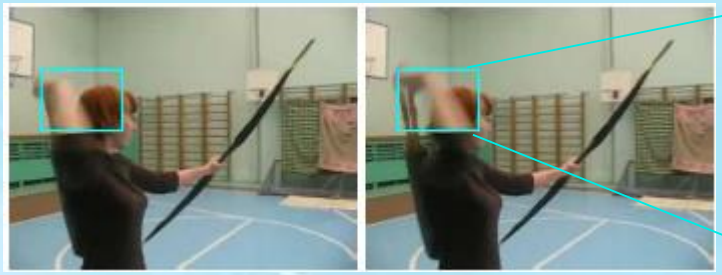


Input frames are fed into two separate streams of processing: a context stream that models low-resolution image and a fovea stream that processes high-resolution center crop. Both streams consist of alternating convolution (red), normalization (green) and pooling (blue) layers. Both streams converge to two fully connected layers (yellow).

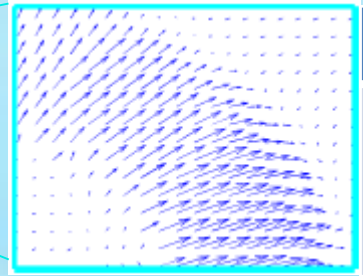
[Karpathy et al. 2014]

Next Milestone in Action Recognition?





two consecutive frames



optic flow

Method	UCF101	HMDB51
Dense Traj. + IFV	87.9%	61.1%
Two Stream ConvNet	**88.0%**	**59.4%**

- Optical flow as input to deep network, i.e., ConvNet
- Spatial stream ConvNet captures shape information
- Temporal stream ConvNet captures motion information
- Achieve state-of-the-art accuracy