Recognizing human behavior in video

Sergio Escalera

1) Action/Gesture recognition
2) Features & methods (handcrafted)
3) Dynamic Time Warping
4) Deep learning modeling of videos

1) Action/Gesture recognition

- Actions/Gestures give us information about people behavior
- Many actions can be determined from a particular pose (**still images**)
- Some actions/gestures are visible analyzing temporal changes in pose (**image sequence**)

Why is action recognition hard?

- Lots of diversity in the data (view-points, appearance, motion, lighting…)

Drinking   Smoking

- Lots of classes and concepts

1) Action/Gesture recognition

Dataset: PASCAL VOC Action Classification

Riding horse   Reading book   Taking photo

Riding bike   Play instrument   Running

Phoning   Use computer   Walking

- Person location given
- Classify into one of 9 categories

Applications: Video Search

- useful for TV production, entertainment, education, social studies, security,…

TV & Web: e.g. "Fight in a parlament"

Home videos: e.g. "My daughter climbing"

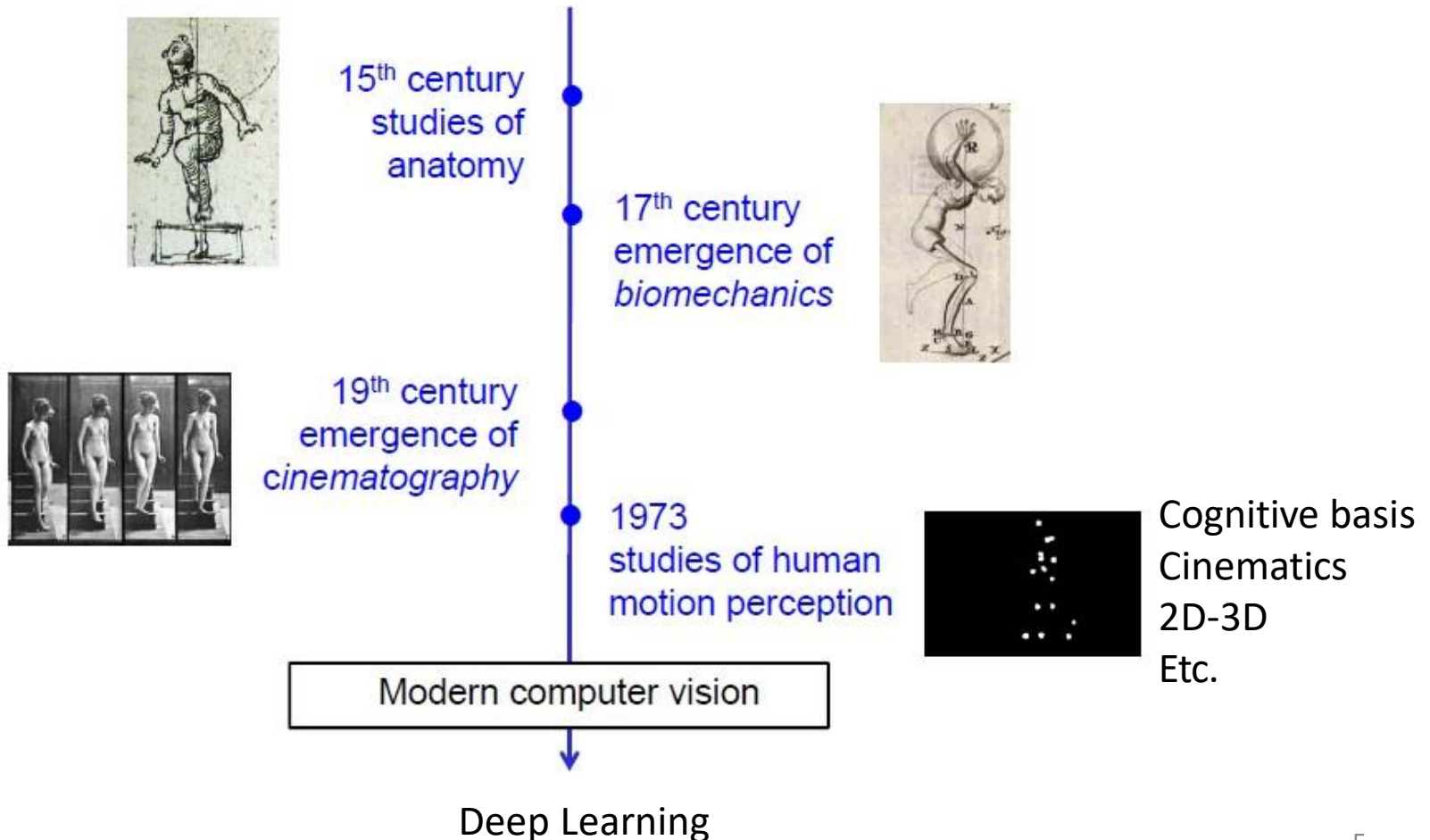Sociology research: e.g. Manually analyzed smoking actions in 900 movies

Surveillance: e.g. "Woman throws cat into wheelie bin" 260K views in 7 days
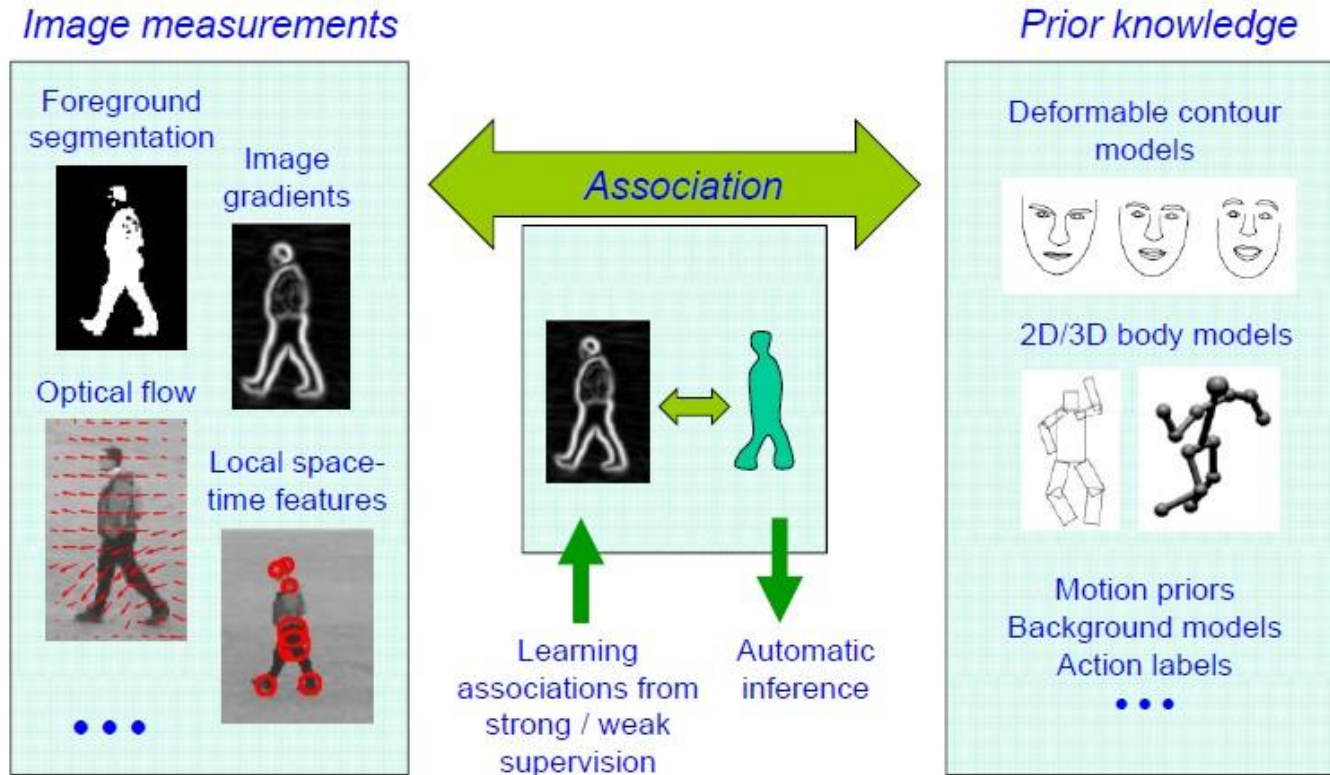
- … and it's mainly about people and human actions

4

1) Action/Gesture recognition

Human actions: Historic overview

15th century
studies of
anatomy

17th century
emergence of
biomechanics

19th century
emergence of
cinematography

1973
studies of human
motion perception

Cognitive basis
Cinematics
2D-3D
Etc.

Modern computer vision

Deep Learning

5

1) Action/Gesture recognition
(can be represented within the Pattern Recognition pipeline)

Action understanding: Key components

Image measurements

Foreground segmentation

Image gradients

Optical flow

Local space-time features

• • •

Association

Learning associations from strong / weak supervision

Automatic inference

Prior knowledge

Deformable contour models

2D/3D body models

Motion priors
Background models
Action labels
• • •

2) Features & methods

- Based on optical flow

    Optical flow (OF) vectors indicate the translation of pixels between a pair of subsequent frames (n, $n + 1$).
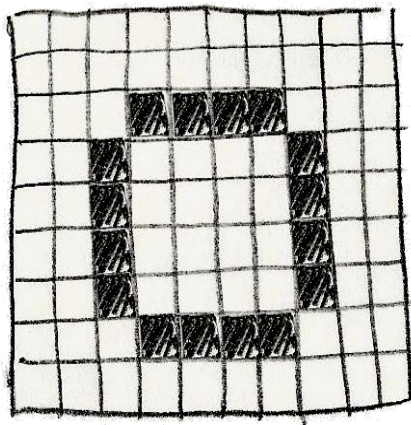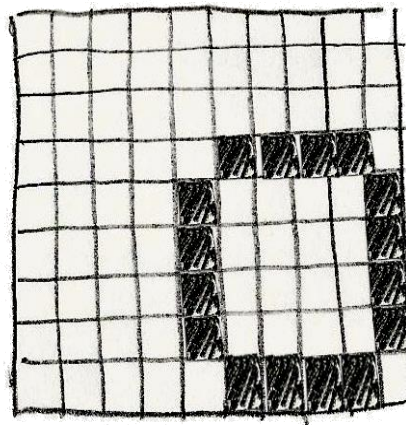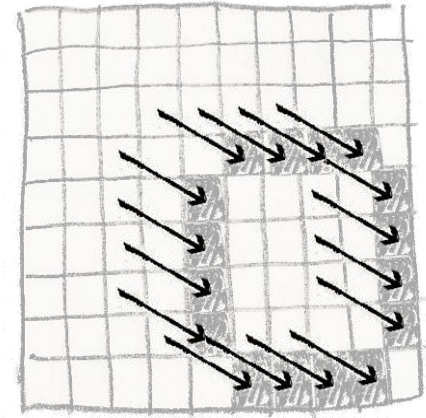
frame n   frame n+1   optical flow
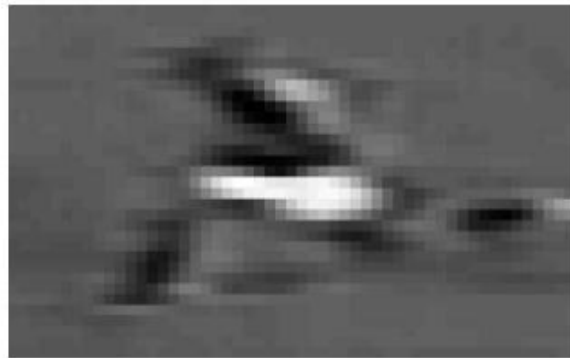
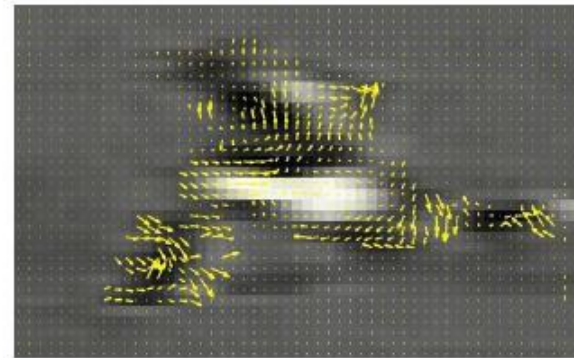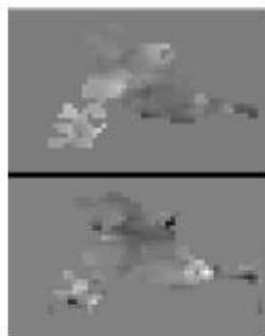# 2) Features & methods
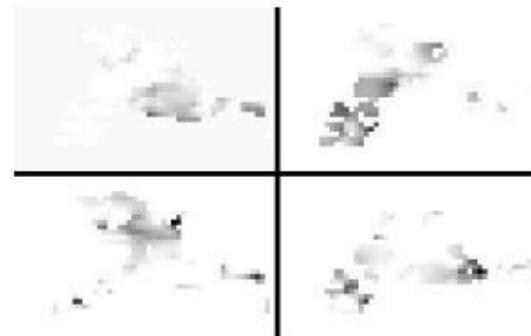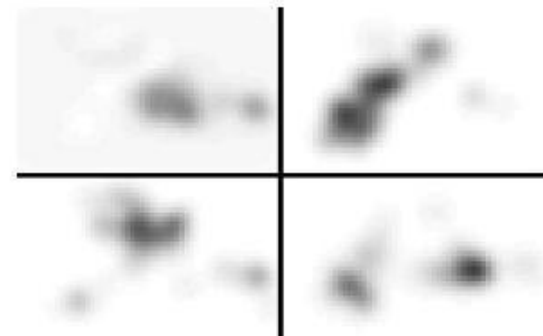
- Based on optical flow

Image frame

Optical flow $F_{x,y}$
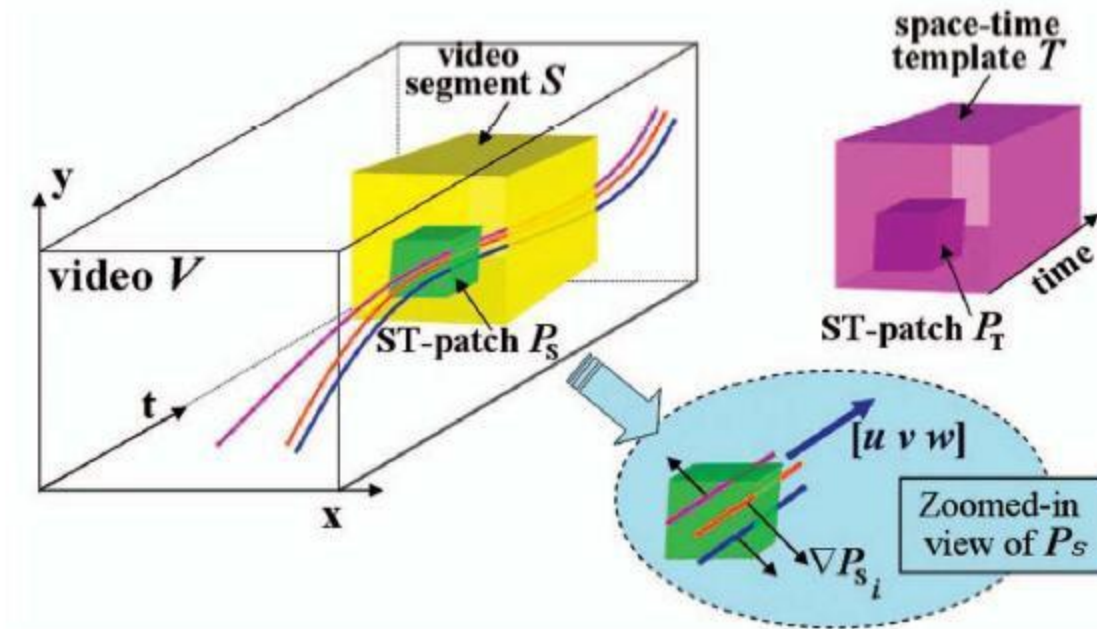
F_x, F_y

$F_x^-, F_x^+, F_y^-, F_y^+$

blurred $F_x^-, F_x^+, F_y^-, F_y^+$

[Efros, Berg, Mori, Malik, ICCV 2003]
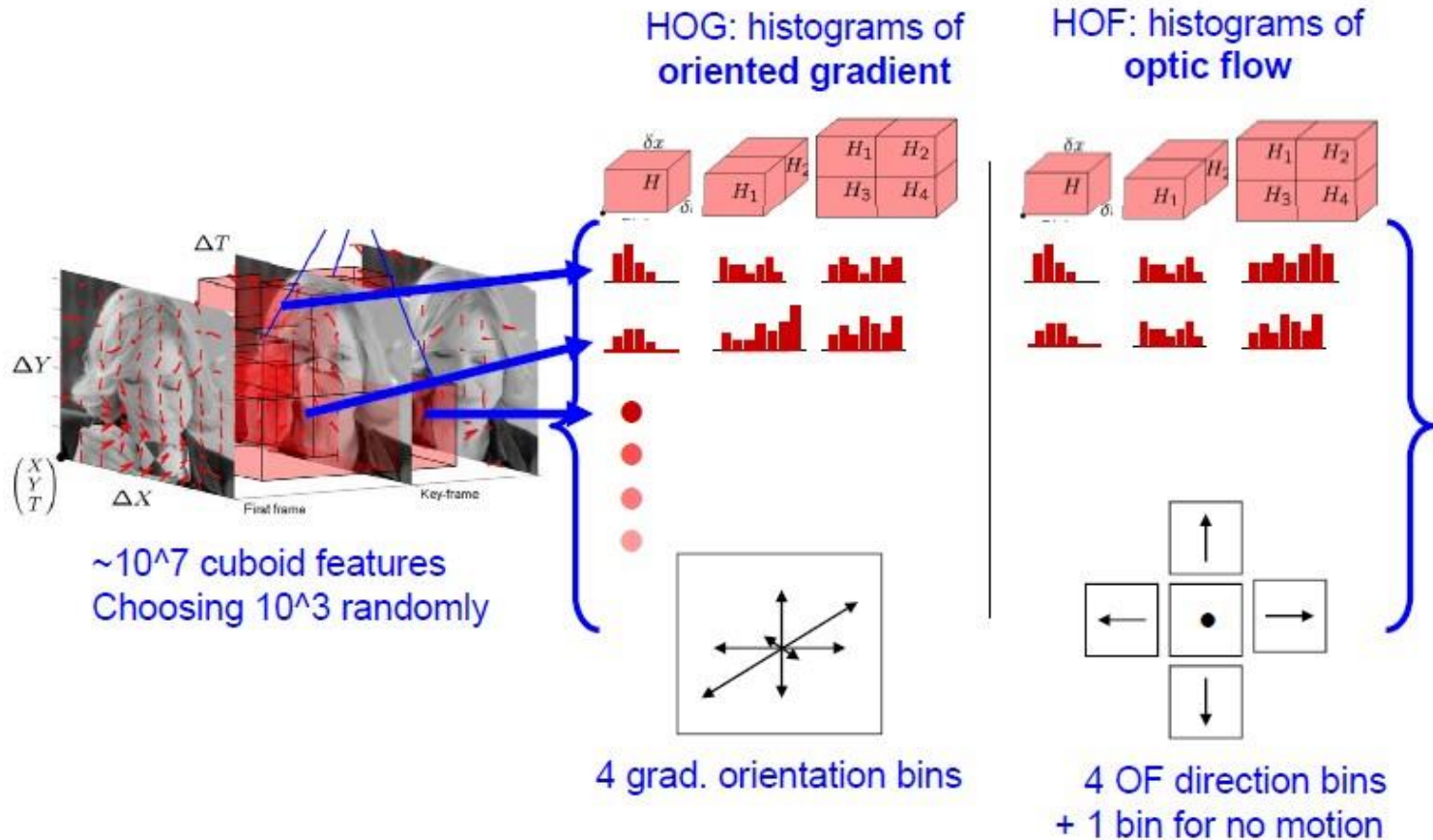
2) Features & methods

- Spatio-Temporal

  - Motion estimation from video is a often noisy/unreliable
  - Measure motion consistency between a template and test video
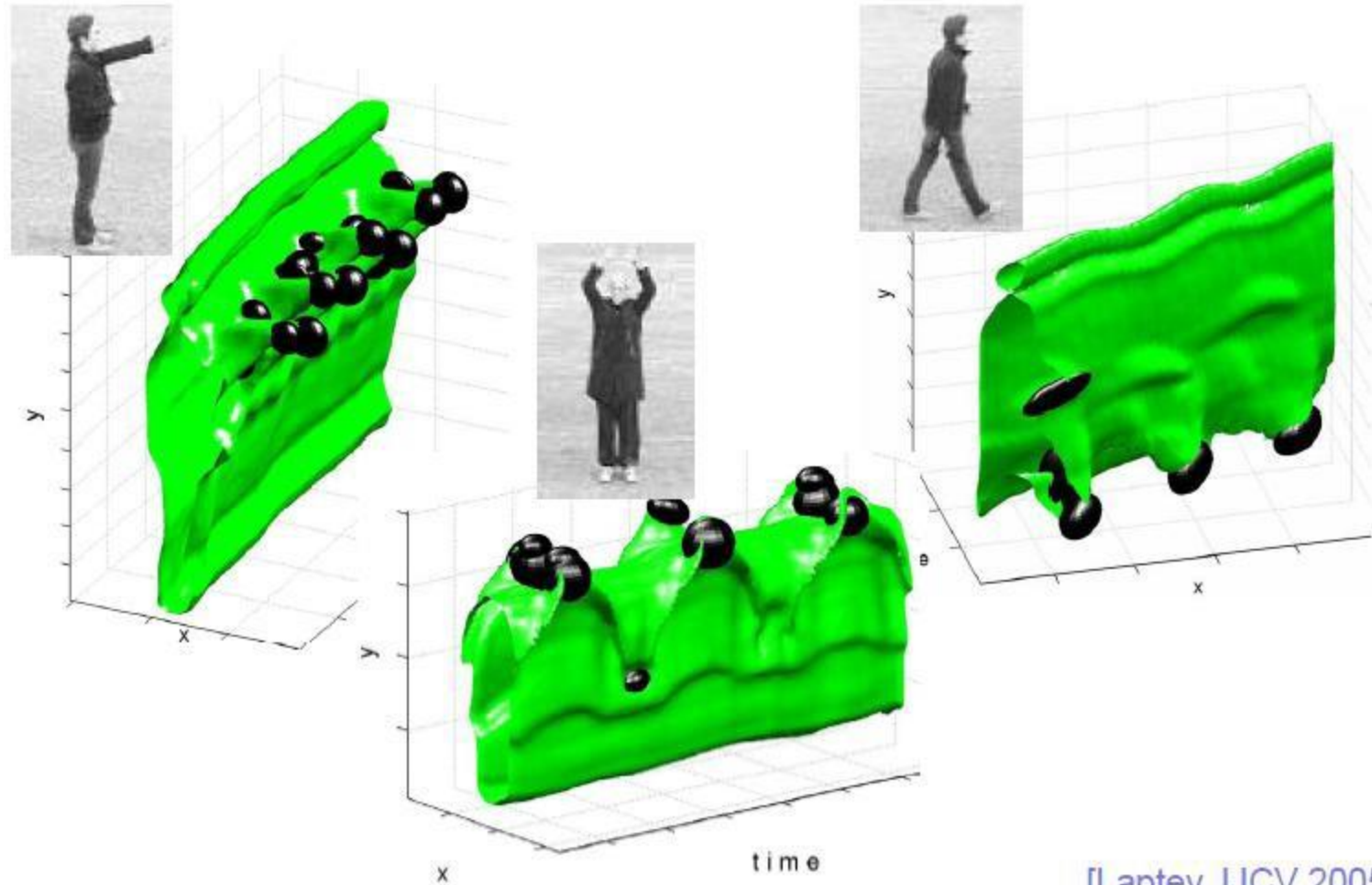

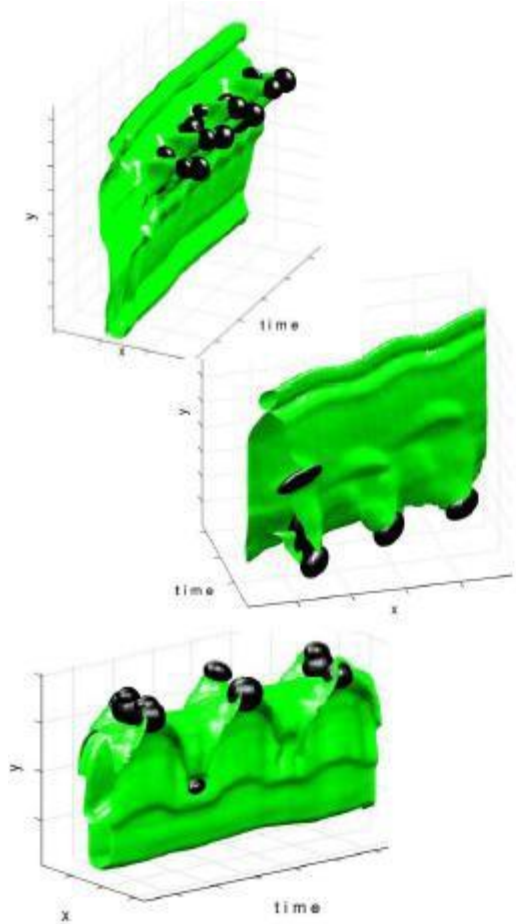[Schechtman and Irani, PAMI 2007]

2) Features & methods

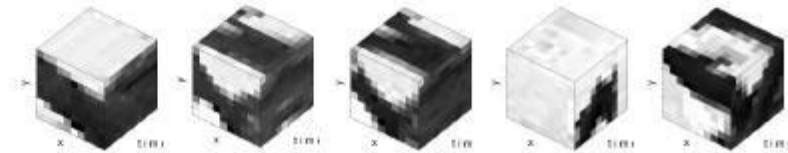# 2) Features & methods
STIPS

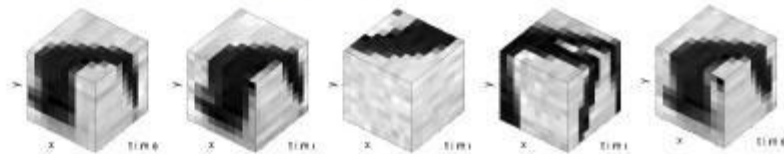[Laptev, IJCV 2005]

2) Features & methods

boxing

walking

hand waving
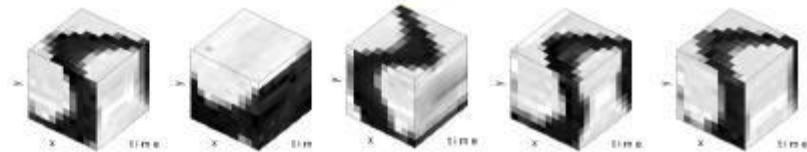
[Laptev, IJCV 2005]

2) Features & methods

Where are we so far ?

Temporal templates:
+ simple, fast
- sensitive to
  segmentation errors

Active shape models:
+ shape regularization
- sensitive to
  initialization and
  tracking failures
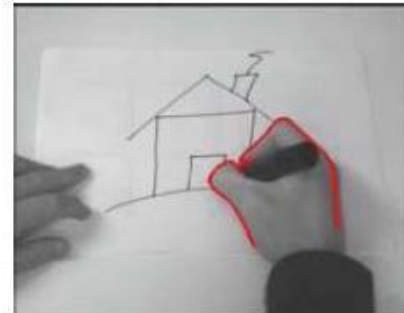
Tracking with motion priors:
+ improved tracking and
  simultaneous action recognition
- sensitive to initialization and
  tracking failures

Motion-based recognition:
+ generic descriptors;
  less depends on
  appearance
- sensitive to
  localization/tracking
  errors
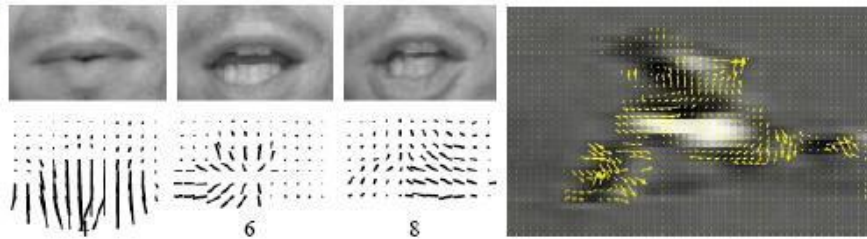
2) Features & methods
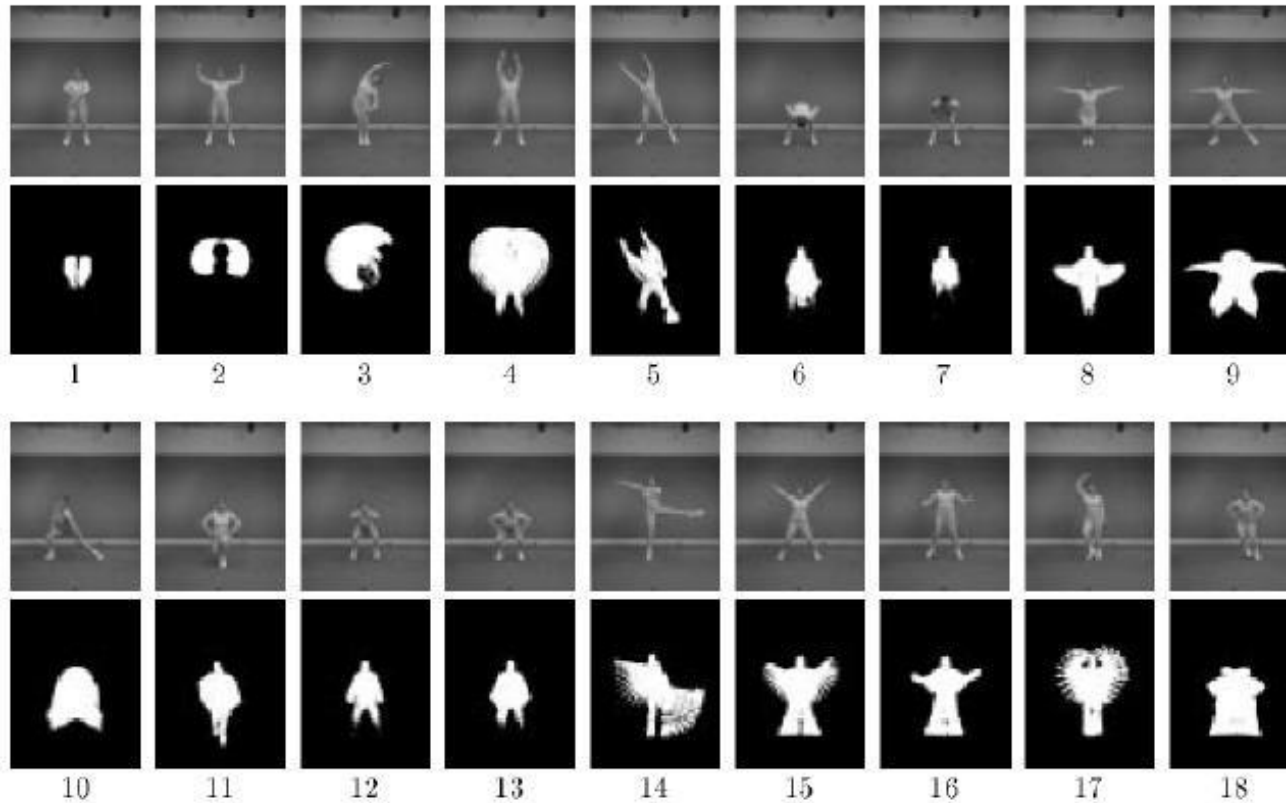
Nearest Neighbor classifier: 66% accuracy

[A.F. Bobick and J.W. Davis, PAMI 2001]

2) Features & methods

Action from pose
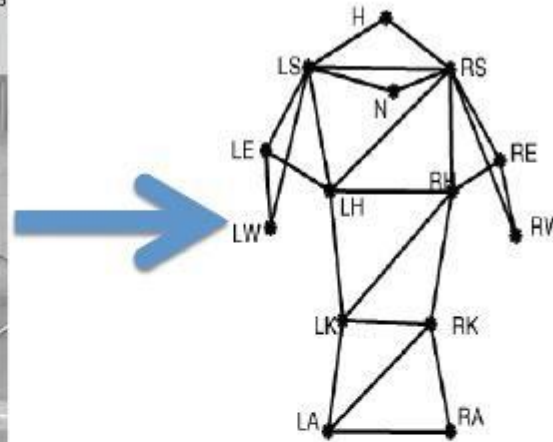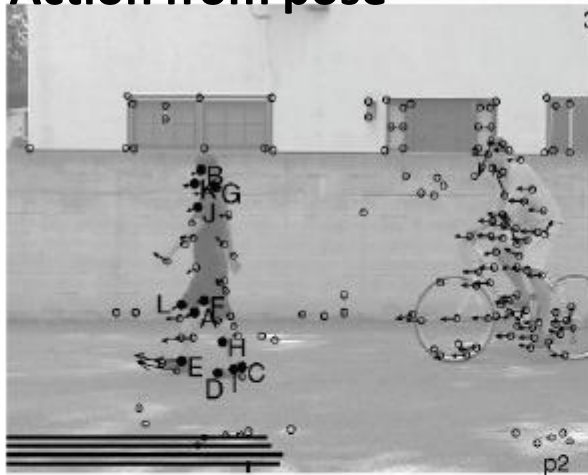
- Detect corners in images/video
- Assess likelihood under action-specific pose model
- Discriminate between walking directions, bicycle riding

Song, Goncalves & Perona NIPS 2001, PAMI 2003

15

2) Features & methods

Action learning

$$H(z) = \text{sgn}(\sum_{t=1}^{T} \alpha_t h_t(f_t))$$

selected features

weak classifier

boosting
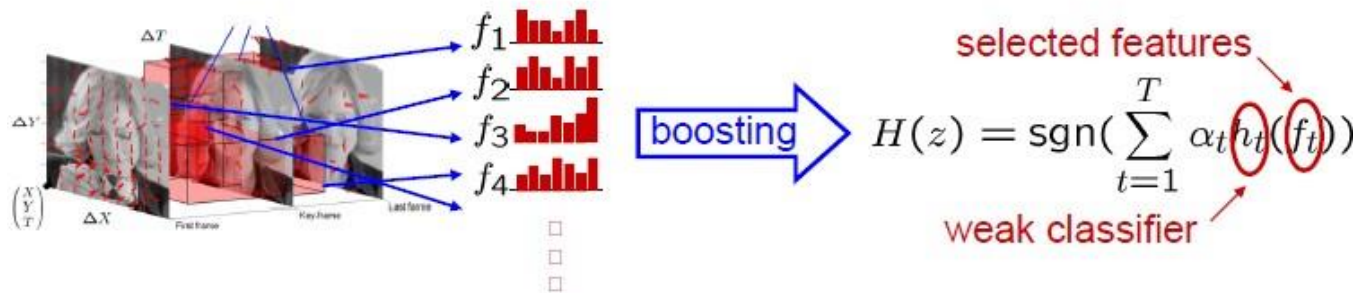
AdaBoost:
- Efficient discriminative classifier [Freund&Schapire'97]
- Good performance for face detection [Viola&Jones'01]

pre-aligned samples

Haar features

optimal threshold

h_t
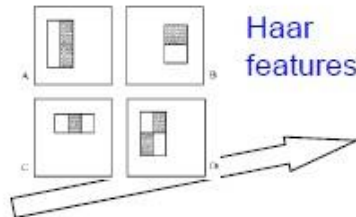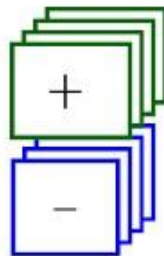
Histogram features
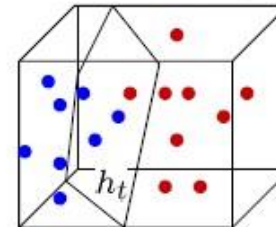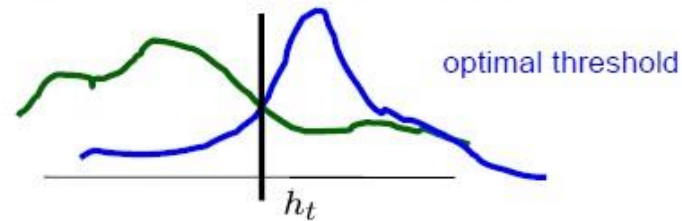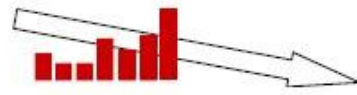
Fisher discriminant

see [Laptev BMVC'06] for more details

2) Features & methods

Bag of space-time features + SVM [Schuldt'04, Niebles'06, Zhang'07,…]

2) Features & methods

A: Tennis forehand, Croquet shot, Volleyball smash ...

O: Tennis racket, Croquet mallet, Volleyball ...

H: Intra-class variations
- More than one H for each A;
- **Unobserved** during training.
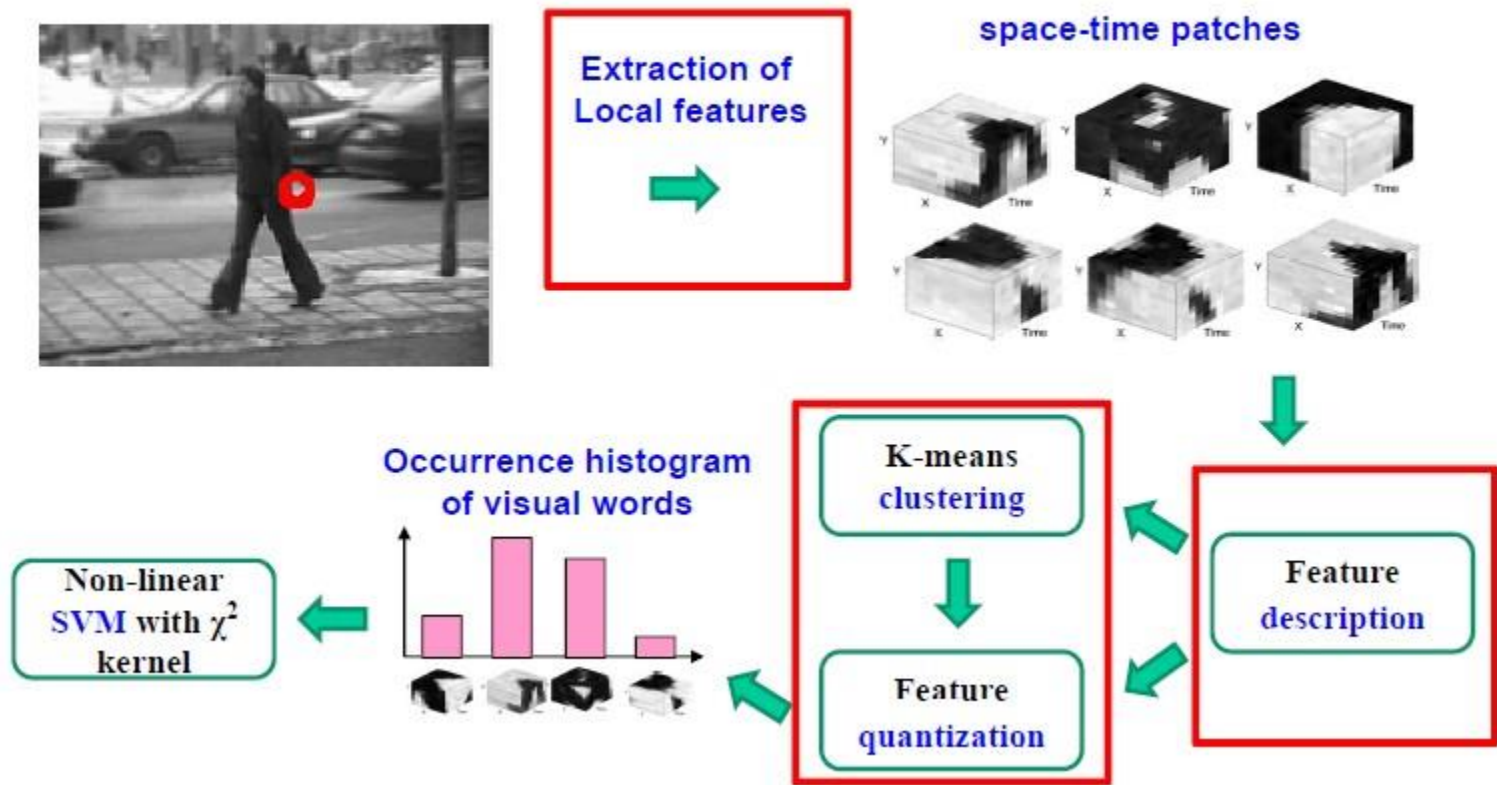
P: l_P: location; θ_P: orientation; s_P: scale.

f: Shape context. [Belongie et al, 2002]

Activity — A

Human pose — H

Object — O

Body parts — P_1, P_2 ... P_N

f_O, f_1, f_2 ... f_N

Image evidence
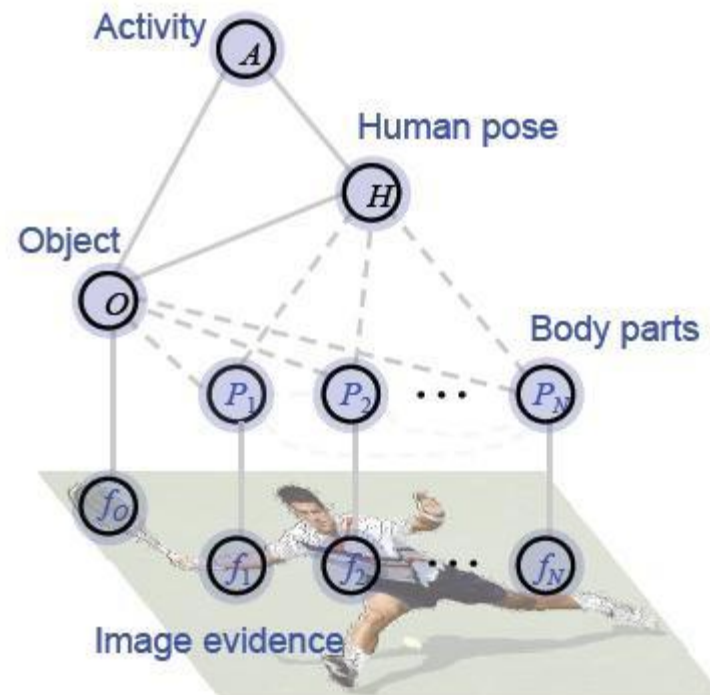
Yao & Fei-Fei CVPR 2010

2) Features & methods
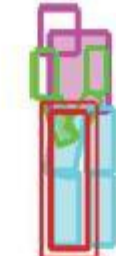
2) Features & methods

Perform Gesture Recognition from hand tracking

- Pruning Methods
  - Matching discriminative feature vectors with gesture models by eliminating large number of hypothesis → Reduces time complexity subproblems
- Deal with the Subgesture problem
- Dynamic Time Warping Based approach (DTW)
  - We will see how DTW works next

Jonathan Alon, Vassilis Athitsos, Quan Yuan and Stan Sclaroff , "A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 31, No. 9, pp 1685–1699, 2009*

3) Dynamic Time Warping

- Dynamic programming

- **Dynamic programming** is a method for solving complex problems by breaking them down into simpler subproblems

- The term *dynamic programming* was originally used in the 1940s by Richard Bellman to describe the process of solving problems where one needs to find the best decisions one after another.

- e.g. path finding:
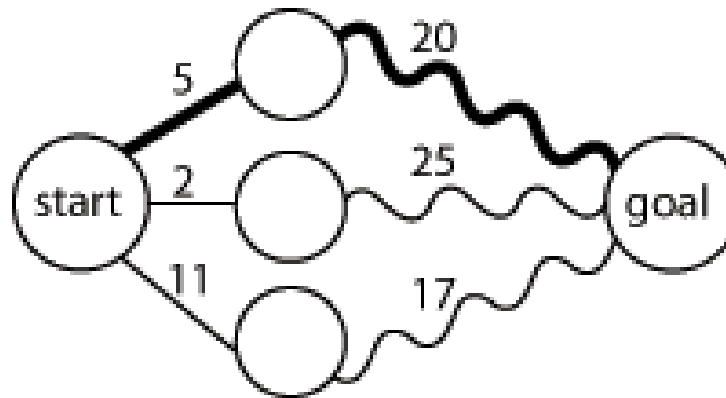
3) Dynamic Time Warping

- Example: text editing

```
for  i = 0, 1, 2, ..., m:
    E(i, 0) = i
for  j = 1, 2, ..., n:
    E(0, j) = j
for  i = 1, 2, ..., m:
    for  j = 1, 2, ..., n:
        E(i, j) = min{E(i − 1, j) + 1, E(i, j − 1) + 1, E(i − 1, j − 1) + diff(i, j)}
return  E(m, n)
```

$$\begin{array}{cccccccccc} E & X & P & O & N & E & N & - & T & I & A & L \\ - & - & P & O & L & Y & N & O & M & I & A & L \end{array}$$

$$O(mn)$$

		P	O	L	Y	N	O	M	I	A	L
	0	1	2	3	4	5	6	7	8	9	10
E	1	1	2	3	4	5	6	7	8	9	10
X	2	2	2	3	4	5	6	7	8	9	10
P	3	2	3	3	4	5	6	7	8	9	10
O	4	3	2	3	4	5	5	6	7	8	9
N	5	4	3	3	4	4	5	6	7	8	9
E	6	5	4	4	4	5	5	6	7	8	9
N	7	6	5	5	5	4	5	6	7	8	9
T	8	7	6	6	6	5	5	6	7	8	9
I	9	8	7	7	7	6	6	6	6	7	8
A	10	9	8	8	8	7	7	7	7	6	7
L	11	10	9	8	9	8	8	8	8	7	6

Human Pose Recovery and Behavior Analysis Group

3) Dynamic Time Warping

```
for i = 0, 1, 2, ..., m:
    E(i, 0) = i
for j = 1, 2, ..., n:
    E(0, j) = j
for i = 1, 2, ..., m:
    for j = 1, 2, ..., n:
        E(i, j) = min{E(i − 1, j) + 1, E(i, j − 1) + 1, E(i − 1, j − 1) + diff(i, j)}
return E(m, n)
```

P O L Y N O M I A L
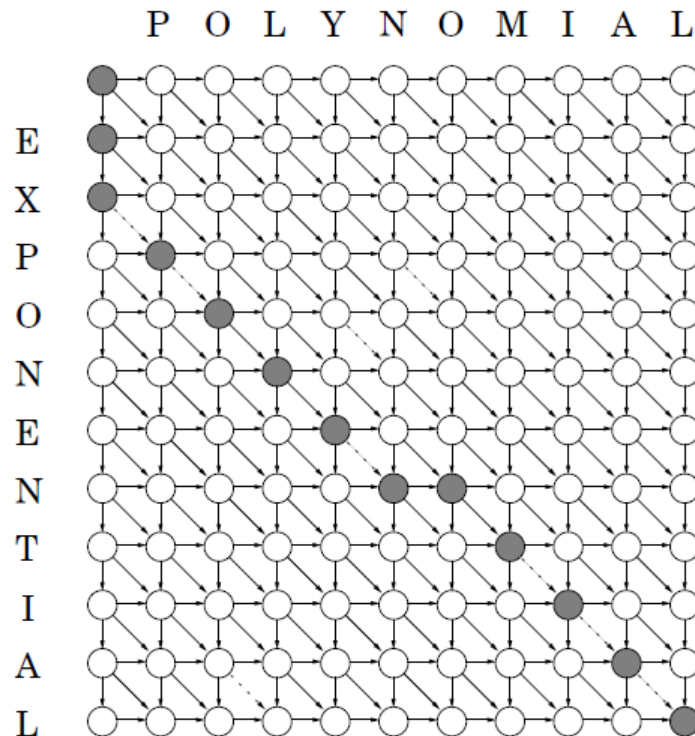
E X P O N E N T I A L

```
E  X  P  O  N  E  N  −  T  I  A  L
−  −  P  O  L  Y  N  O  M  I  A  L
```

It can exist different ways
(**working paths**)

23

3) Dynamic Time Warping

- Dynamic Time Warping

- Algorithm for measuring similarity between two sequences

  - Considering variations in time or speed

  - Objective is to find the optimal match

  - Data which can be analyzed

    - Any linear representation

    - *Audio*

    - *Video*

    - *Graphics*

```
int DTWDistance(char s[1..n], char t[1..m]) {
    declare int DTW[0..n, 0..m]
    declare int i, j, cost

    for i := 1 to m
        DTW[0, i] := infinity
    for i := 1 to n
        DTW[i, 0] := infinity
    DTW[0, 0] := 0

    for i := 1 to n
        for j := 1 to m
            cost:= d(s[i], t[j])
            DTW[i, j] := cost + minimum(DTW[i-1, j  ],    // insertion
                                        DTW[i  , j-1],    // deletion
                                        DTW[i-1, j-1])    // match

    return DTW[n, m]
}
```

Sakoe, H. and Chiba, S., *Dynamic programming algorithm optimization for spoken word recognition*, IEEE Transactions on Acoustics, Speech and Signal Processing, 26(1) pp. 43– 49, 1978, ISSN: 0096–3518.
C. S. Myers and L. R. Rabiner. A comparative study of several dynamic time-warping algorithms for connected word recognition. The Bell System Technical Journal, 60(7):1389–1409, September 1981.
L. R. Rabiner and B. Juang. Fundamentals of speech recognition. Prentice-Hall, Inc., 1993 (Chapter 4)
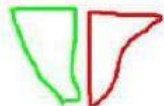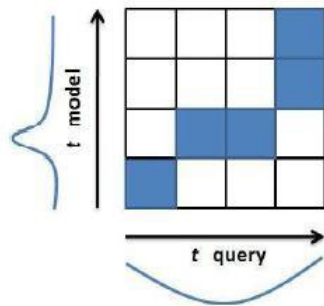
Human Pose Recovery and Behavior Analysis Group

3) Dynamic Time Warping

- ## Sign Language Recognition

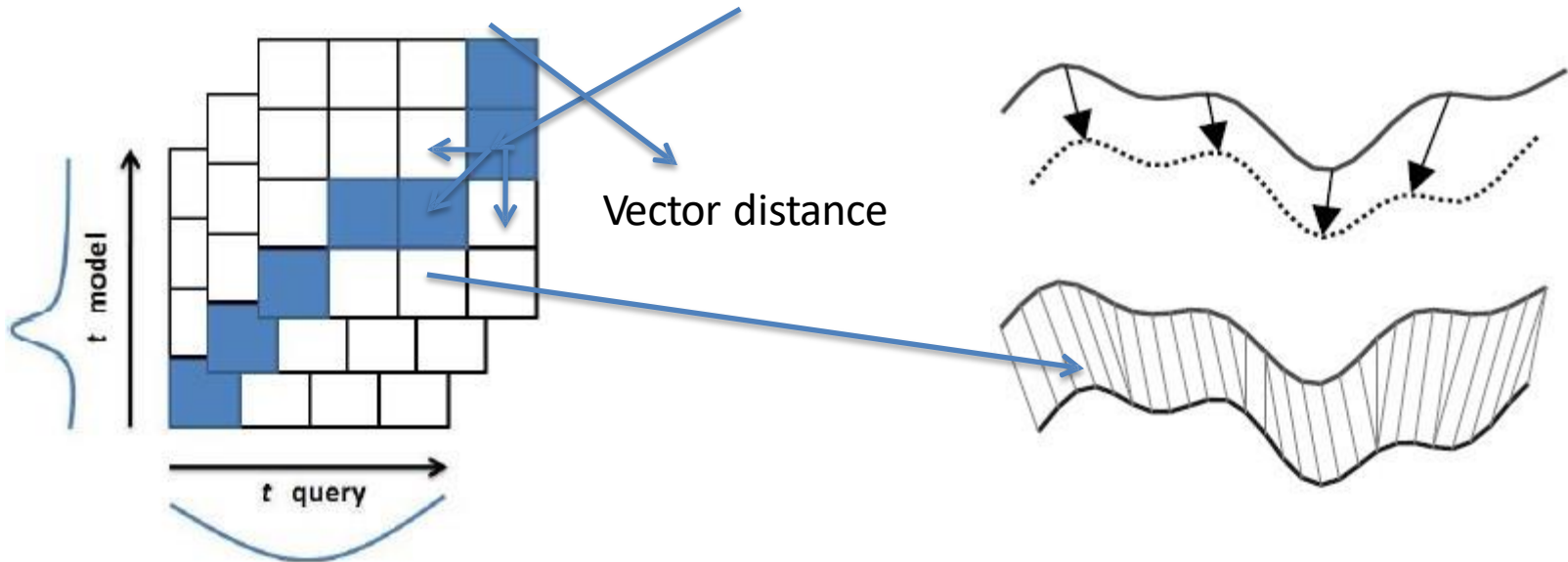Face detection
Skin color modeling Noise
removing
Blob detection and tracking Dynamic Time
Warping

3) Dynamic Time Warping

- ## Multiple candidates

$$\gamma(i, j, k) = d(i, j, k) + \min\{\gamma((i-1, j-1), (i-1, j), (i, j-1) \times \{1, .., K\})\}$$

Vector distance

t model

t query

- Multiple candidates
  - Candidates can appear and disappear
- Sub-patterns can be detected from a large sequence
- Distances can be changed to cost or probabilities
- Gesture match sequence requires from threshold distance/cost
  - → Learnt or empirically set

25

4) Moving to Deep learning

Until the apparition of the Two-stream ConvNet, hand-crafted methods dominated state-of-the-art of action classification.
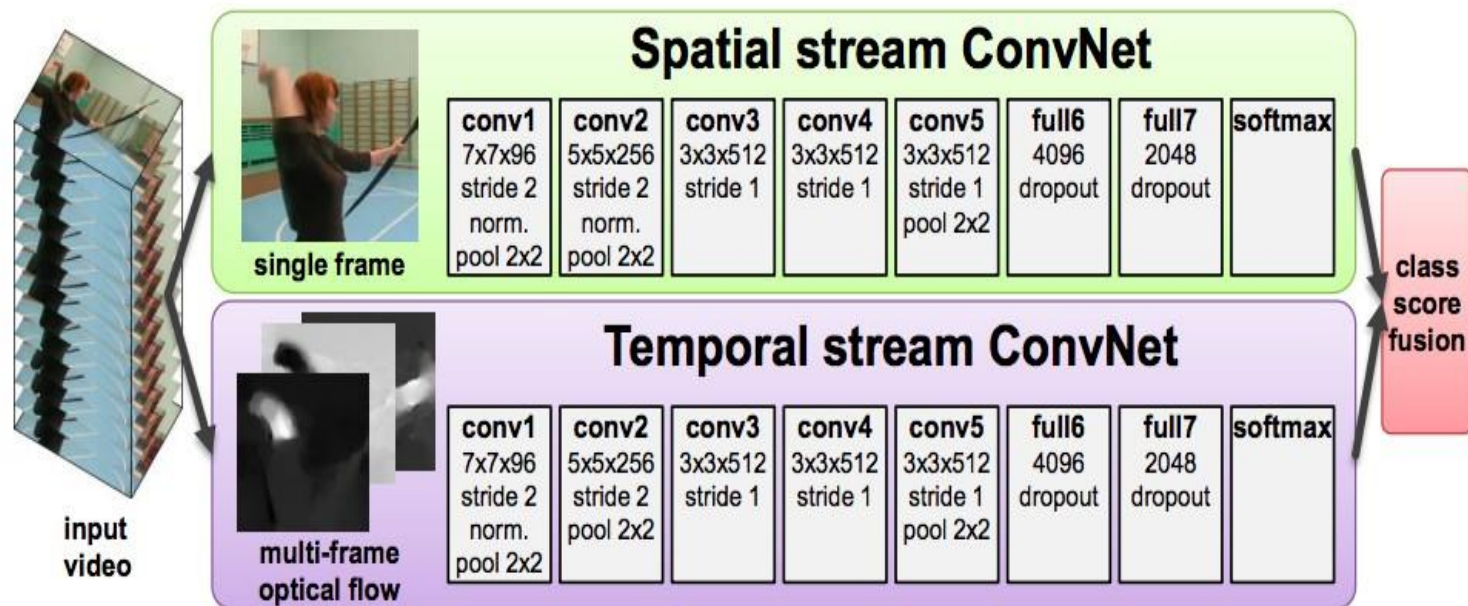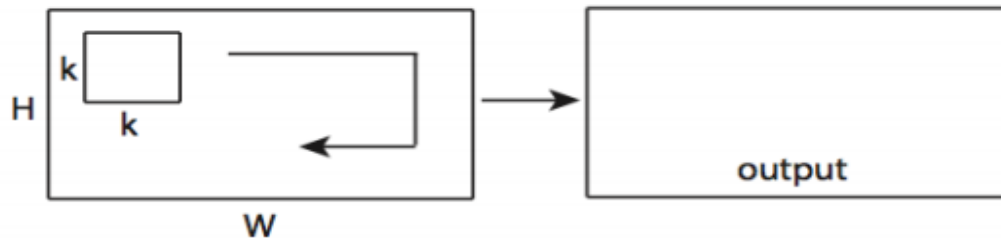

Karen Simonyan and Andrew Zisserman. "Two-stream convolutional networks for action recognition in videos". In: Advances in Neural Information Processing Systems. 2014, pp. 568–576.

4) Moving to Deep learning

Must know: 2D vs 3D convolutions

Convolving a $k \times k$ filter (or kernel) on a $H \times W$ grayscale image produces a 2D response map (output).
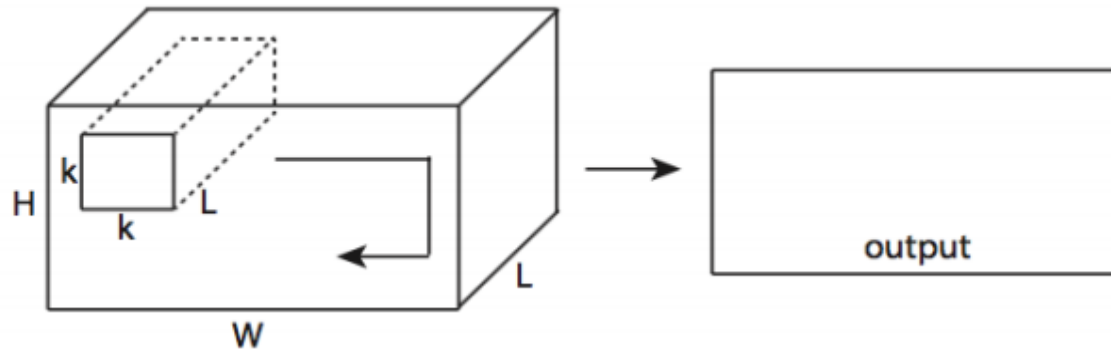

2D convolution

4) Moving to Deep learning

Must know: 2D vs 3D convolutions

Convoling a $k \times k \times L$ filter (or kernel) on a $H \times W \times L$ stack of grayscale images also produces a 2D response map (output).

2D convolution on multiple frames

The temporal information is merged (or "lost") after the first convolutional layer.

4) Moving to Deep learning

Must know: 2D vs 3D convolutions

Convoling a $k \times k \times d$ filter (or kernel) on a $H \times W \times L$ stack of grayscale images, where $d < L$, produces a 3D response map (output).
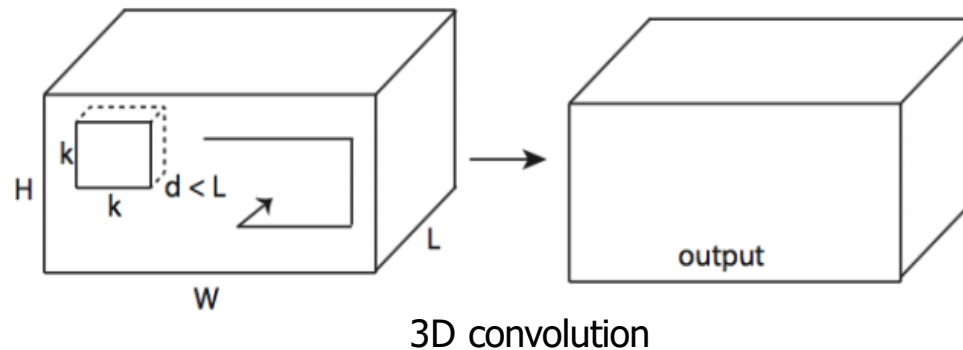

3D convolution

The temporal structure is mantained throughout subsequent network layers.

4) Moving to Deep learning

Taxonomy: action classification architectures

In the current SOTA, action classification (AC) methods can be roughly categorized following this taxonomy:

1. Two-stream ConvNets
2. 3D ConvNets
3. ConvNet + LSTM
4. Two-stream Inflated 3D ConvNets
5. Transformer-like architectures

Taxonomy (1/5): Two-stream ConvNets

Two separate ConvNets – namely *streams* – process, respectively, appearance (RGB frames) and motion (pre-computed *optical flow* stacks). Whereas RGB frames are $H \times W \times 3$, OF stacks are $H \times W \times 2$. During **training**, the network learns to *classify individual RGB frames or OF stacks* centered at the corresponding frame.

During **test**, the *video-level prediction* is got from averaging class scores from several frames from each stream and performing a weighted sum of both streams.

+ Appearance stream can re-use pre-trained on image classification.
+ Motion stream rapidly trained from scratch.

− Temporal information in motion stream is dropped after the 1st conv layer.
− Ignores long-term temporal information.
− Complexity
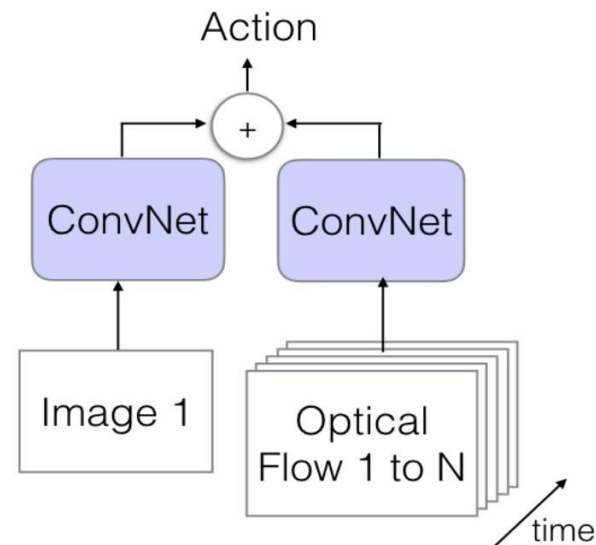
Taxonomy (2/5): 3D ConvNets

Convolutional layers and pooling kernels are extended in time, i.e. inputs are a $K \times H \times W \times 3$ tensor and filter (output) response maps are each a 3-D tensor. During **training**, the network learns to *classify short video snippets*, e.g. $K = 16$ frame clips.

The net is able to model local spatiotemporal information, e.g. motion-based features. During **test**, for *video-level prediction* class scores from several K-frame clips are averaged.

+ Models very rich but local spatiotemporal features.

− Harder to train than Two-stream ConvNets → more data needed.
− Ignores long-term temporal information.
− Do not re-use of powerful pre-trained image-based ConvNets.

Action

3D ConvNet

Images
1 to K

time
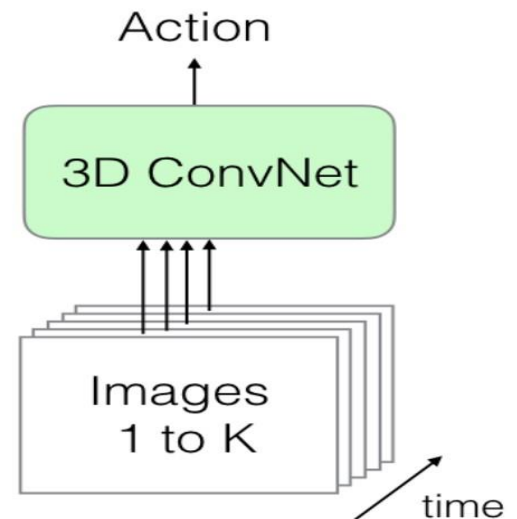
Taxonomy (3/5): ConvNet + LSTM

Individual frames are input to a ConvNet $g(X; \theta_g)$ with shared parameters θ_g. The sequence of outputs $z_i = g(X_i; \theta_g)$, $1 \le i \le K$, is input to a LSTM.

Recursively, the LSTM outputs a hidden state h_i after receiving (z_i, h_{i-1}). During **test**, a video-level prediction in form of class score distribution is produced: $y = \mathrm{softmax}(f(h_K; \theta_f))$, where $f(\cdot)$ is a feed-forward net and $y \in \mathrm{R}^C$.

+ Long-term temporal information is modeled.

− Larger K values → smaller batch size.
− Recurrency difficults GPU-parallelization.
− In practice, K very large does not work properly…
− Useful with HARD sequencial dependencies
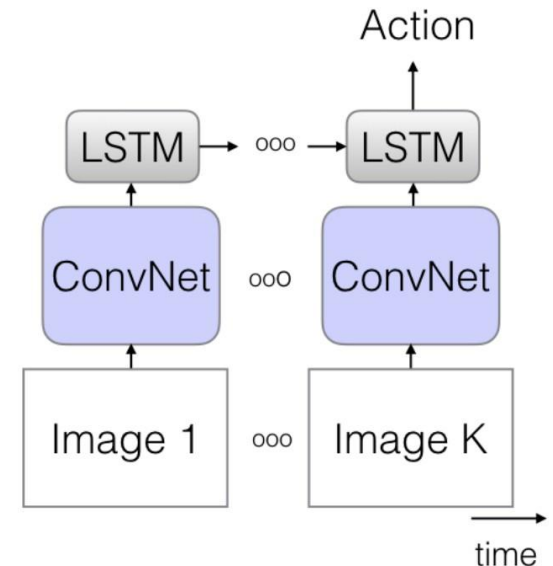
Taxonomy (4/5): Two-stream I3D ConvNets

Put together the best of 3D and Two-stream ConvNets: it *inflates* all the filters and pooling kernels from $N \times N$ to $N \times N \times N$ and includes an OF-based stream.

Given the inflation, the 3D filters are initialized from pre-trained 2D filter weights by repeating them N times along time dimension and rescaling them by dividing by N.
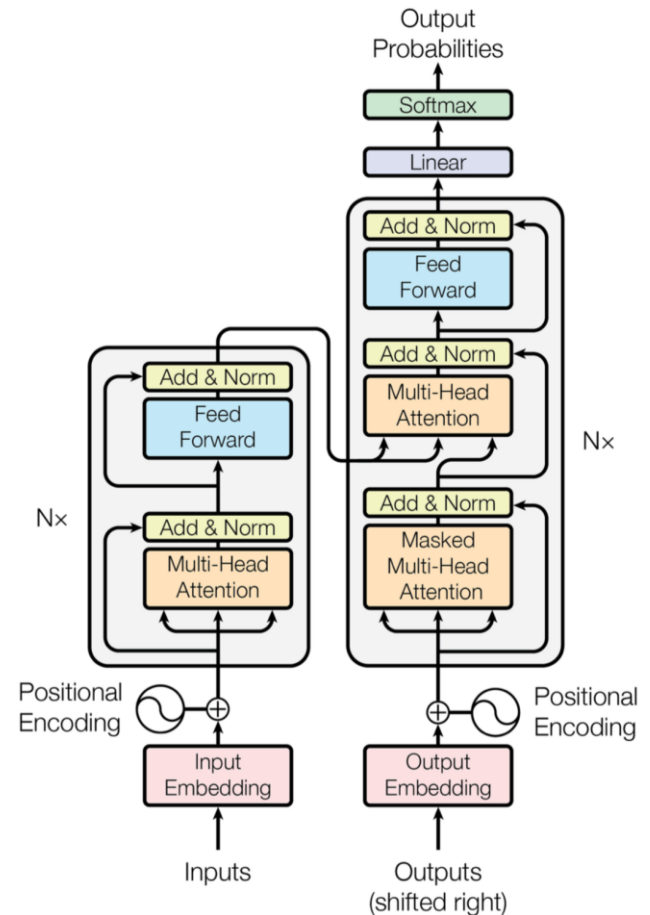
+ All the ones from 3D and Two-stream ConvNets.

+ Performance.

− Ignores long-term temporal information.
− Complexity

35
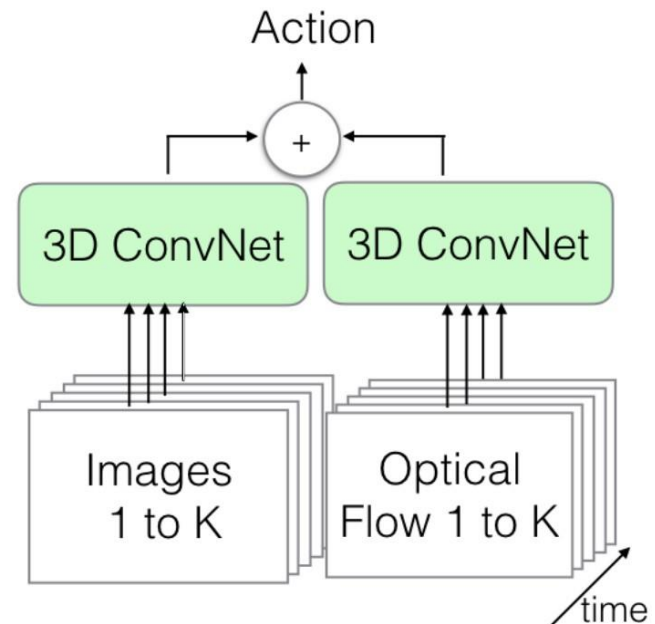
Taxonomy (5/5): Transformer-like architectures

It encodes frame-based or spatio-temporal input features (depending on the input backbones) and take benefit of self-attention mechanism.

+ Can model any kind of input
+ Avoid recurrent modeling
+ Can exploit long term relationships

− Pretrain helps but training is hard and tricky
− Complexity / difficult end-to-end modeling

Recently → check MAMBA !!!!