REAL TIME VIDEO ANALYTICS

Dr TIAN Jing
tianjing@nus.edu.sg

Module objective

Module: Video analytics

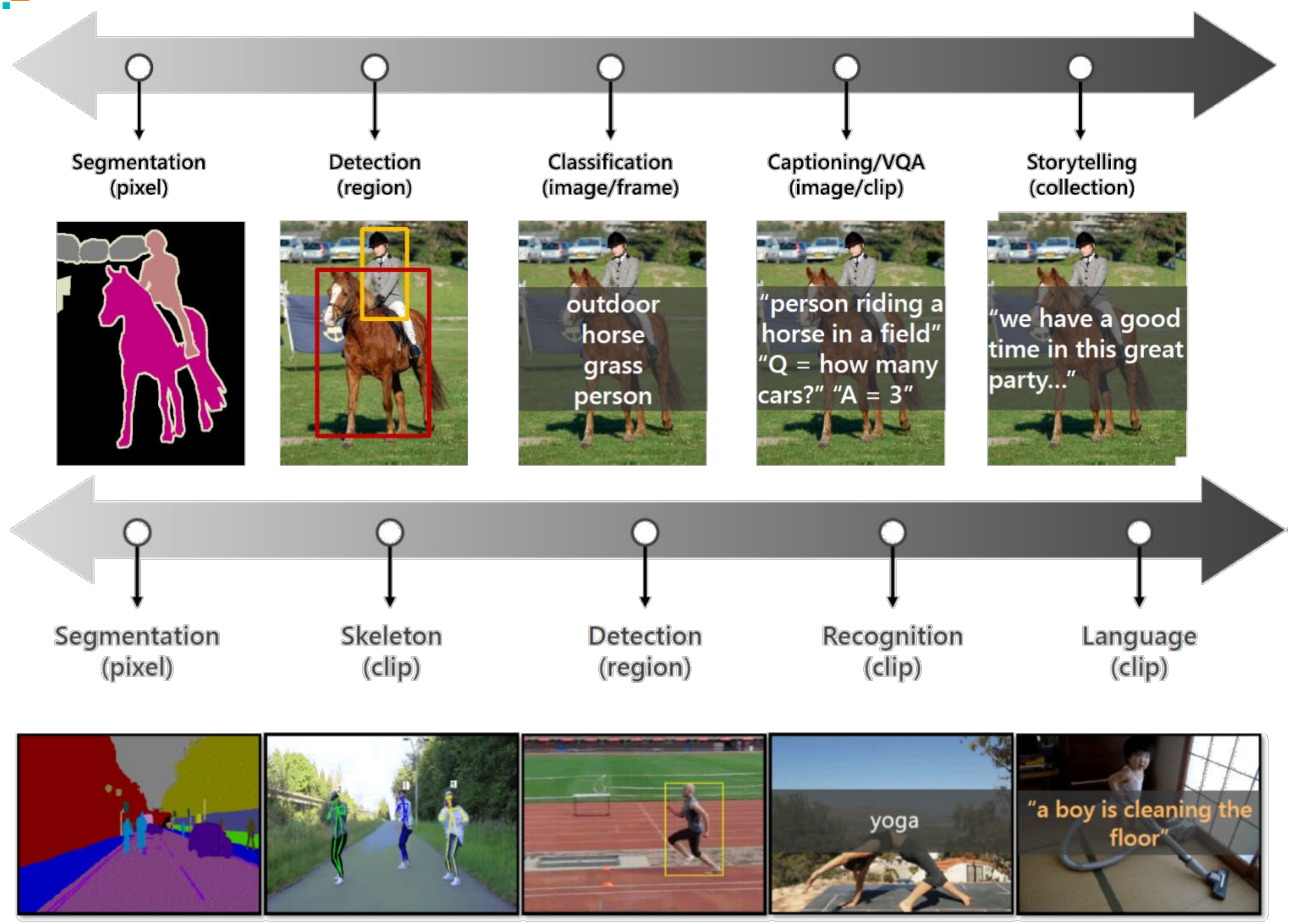
Knowledge and understanding

- Fundamentals of video understanding, action recognition

Key skills

- Design, build, implement and evaluate various action recognition and video understanding systems

From image analytics to video analytics



Reference: Deep Learning for Intelligent Video Analysis, ACM Multimedia 2017, https://www.microsoft.com/en-us/research/wp-content/uploads/2017/10/Tao-Mei-Intelligent-Video-Analysis-ACMMM-2017-Pub.pdf

Video analytics

- Video analytics applications typically address information needs that are typically referred to as four “W” questions:
  - Who (people detection and identification);
  - What (object, activity, event, behavior, and relationship analysis);
  - Where (frame space, 3D space, and world map space); and
  - When (date/day, time-of-day, time-of-year)
- Video analytics can be applied to
  - Retrospective analysis of archives (archive management, search, triage, forensic investigation),
  - Real-time analysis of live video streams (situation awareness and alerting, encoding, compression),
  - Predictive analyses leveraging both live video streams and archives as well as data from other correlated domains (prediction based on the past and present, event/activity prediction, anomaly detection).

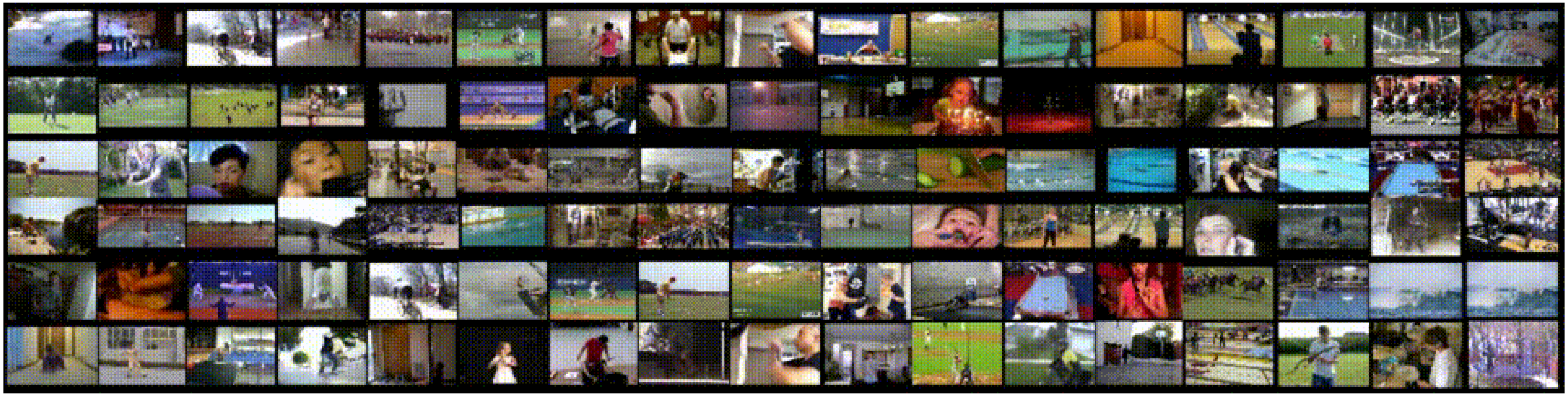
Reference: NIST Video Analytics in Public Safety, https://www.nist.gov/sites/default/files/documents/2017/07/26/ir_8164.pdf

Topics

- Action recognition
- Workshop: Build action recognition systems

Human activity in video

- Action: Atomic motion patterns, gesture-like, single clear-cut trajectory, single nameable behavior (e.g., sit, wave arms)
- Activity: Series or composition of actions (e.g., interactions between people)
- Event: Combination of activities or actions (e.g., a sport game, a traffic accident)



Action recognition: Given an input video sequence, perform some appropriate processing, and output the "action label" of the whole video sequence.

Photo: http://www.thumos.info/

Applications

- Intelligent assisted living
- Crowd analysis in surveillance
- Social activity recognition
- Many other applications

Photo: https://warwick.ac.uk/fac/sci/dcs/people/victor_sanchez/siplab/code/; https://tomhsu1990.github.io/research/ActionRecog.html; https://www.researchgate.net/figure/Examples-of-Our-Senior-Activity-Recognition-Dataset_fig11_221263478;

Challenges in action recognition

- Intra- and inter-class variations: People behave differently for the same actions.
- Cluttered background and camera motion: Indoor controlled environments but not in outdoor uncontrolled environments.
- Insufficient annotated benchmark dataset
- Uneven predictability: Not all frames are equally discriminative.

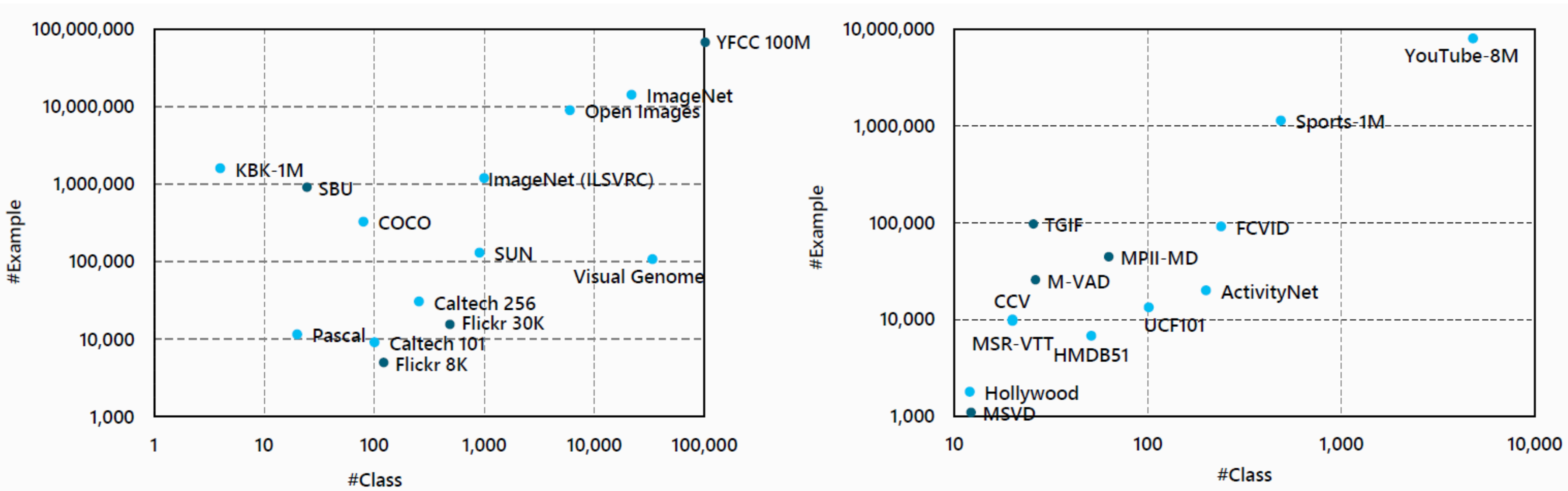


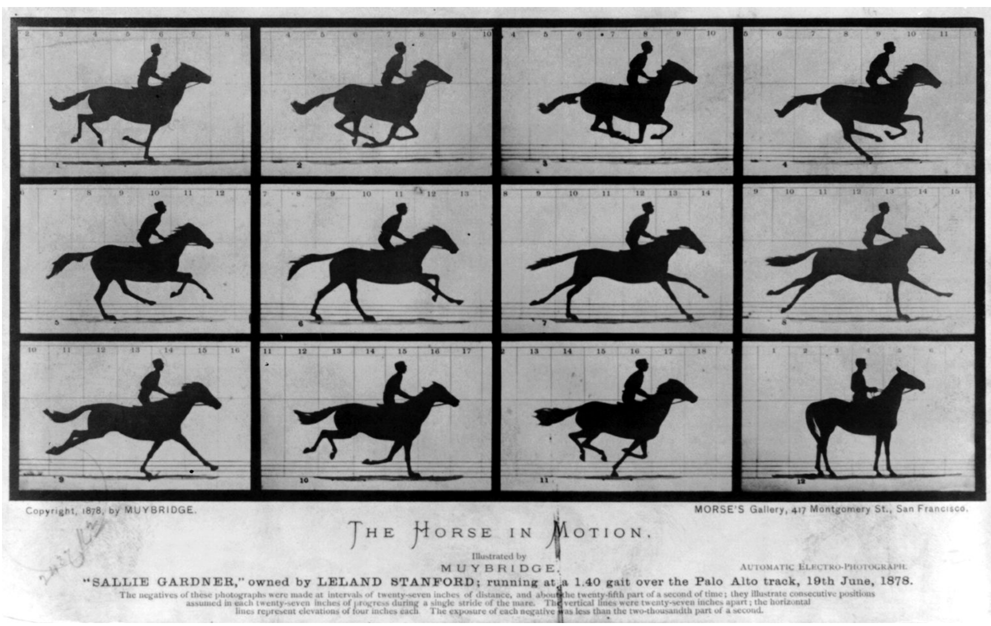
Figure: Deep Learning for Intelligent Video Analysis, ACM Multimedia 2017, https://www.microsoft.com/en-us/research/wp-content/uploads/2017/10/Tao-Mei-Intelligent-Video-Analysis-ACMMM-2017-Pub.pdf

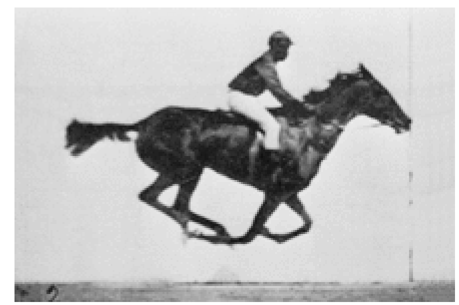
Human action recognition in history

A study in 1878

The Horse in Motion is a series of six cabinet cards by Eadweard Muybridge, each showing a sequential series of six photos depicting the movement of a horse.

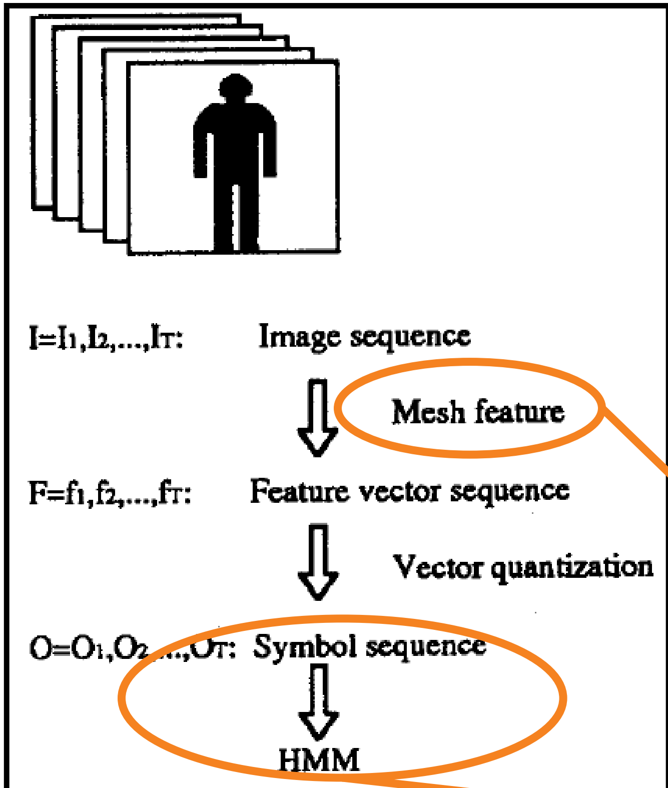
Reference: https://en.wikipedia.org/wiki/Eadweard_Muybridge

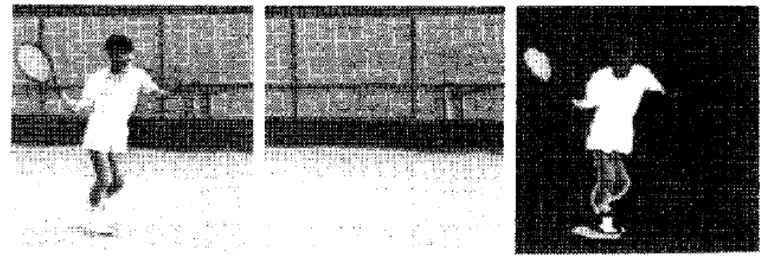




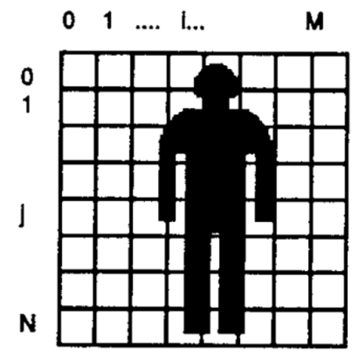
An animation generated from photos

Human action recognition in history





original background foreground

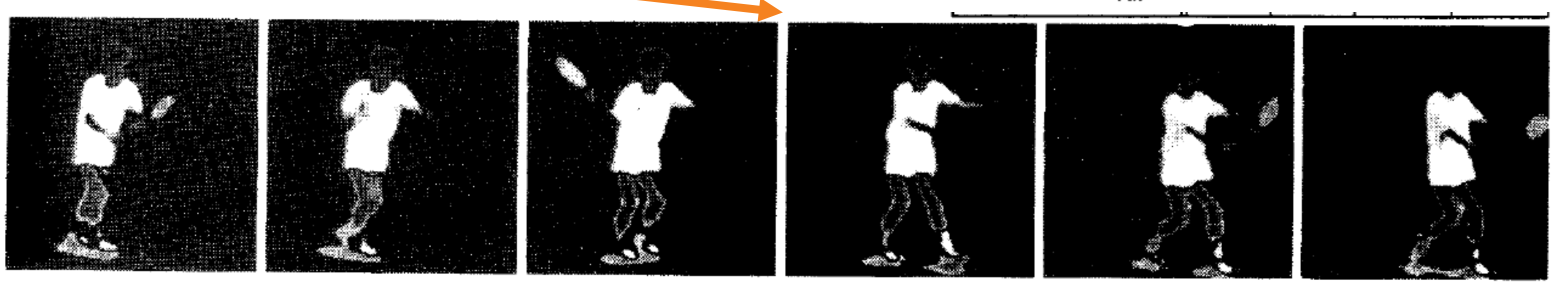


$f=(a_{00},a_{01},\ldots,a_{ij},\ldots a_{MN})$

a_{ij}=number of black mesh(ij)/$M_m N_m$

A study in 1992

J. Yamato, et al., Recognizing human action in time-sequential images using hidden Markov model, CVPR 1992, https://www.cs.sfu.ca/~mori/courses/cmpt888/summer10/papers/yamato_cvpr92.pdf



Symbol sequence	60 61 61 62 62 62 63 63 64 64 65 66 66 66 67 68 68 69 69 70 70 70 71 71

Action: Single frame based methods

Key idea

- Action can be recognized by posture (see the horse in motion slide)
- So many research works in image recognition particularly in deep learning

CNN for action recognition

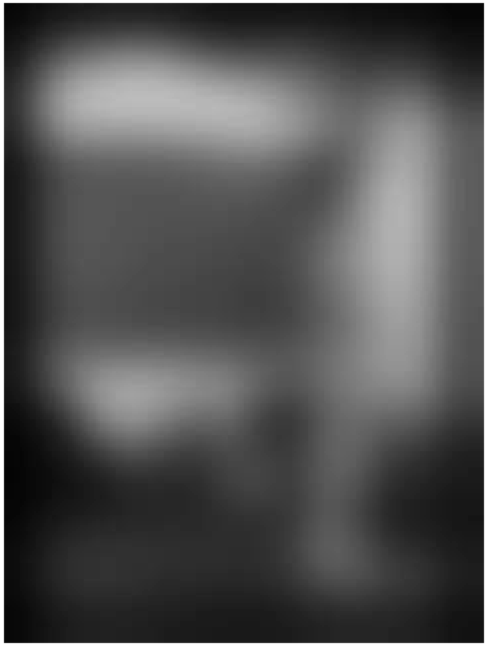
- Input: Region containing the actor
- Output: Action category labels

MPII human pose dataset (http://human-pose.mpi-inf.mpg.de/)

- 410 actions and 40000 instances

Multiple frames based methods

- Intuitive idea: If we can convert multiple frame into a single frame, then we can apply previously-studied single frame based methods
- Question: How to convert video sequence (a set of images) into a single frame?



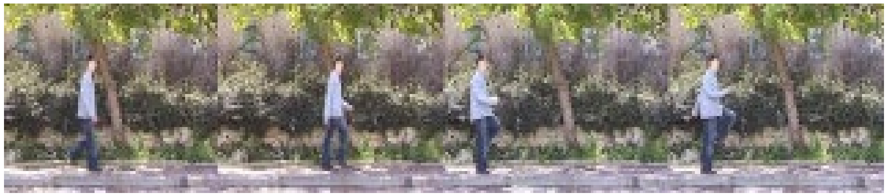
When viewing the motion in a blurred sequence, two distinct patterns are apparent.

- The first is the spatial region in which the motion is occurring. The pattern is defined by the area of pixels where something is changing largely independent of how it is moving.
- The second pattern is how the motion itself is behaving within these regions (e.g. an expanding or rotating field in a particular location).

We need to exploit these notions of where and how, since these observations capture significant motion properties of actions that can be used for recognition.

Reference: http://web.cse.ohio-state.edu/~davis.1719/CVL/Research/MHI/mhi.html

Silhouette



Extract silhouette by background subtraction

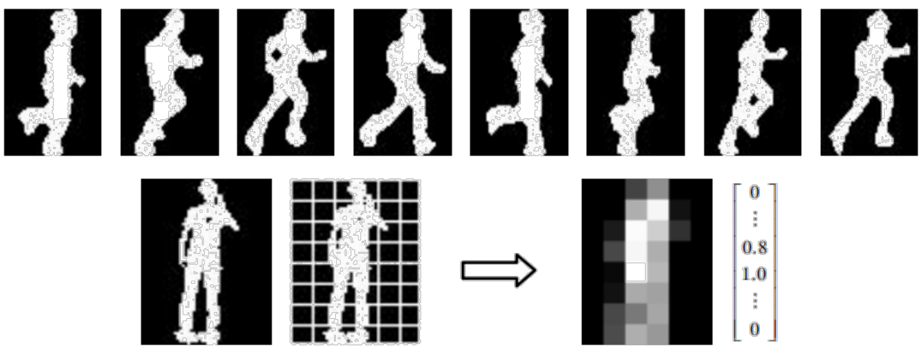


Figure 2. The normalized silhouette sequences of running (*top*) and the illustration of block-based feature representation (*bottom*)

- Given an action video with a set of frames, extract the associated sequence of moving silhouettes.
- The silhouette images are centred and normalized on the basis of keeping the aspect ratio property of the silhouette.
- Divide each silhouette image into $h \times w$ non-overlapping sub-blocks. The silhouette feature is defined as the normalized value of each sub-block by $F_i = \frac{N(B_i)}{hw}, i = 1,2,\cdots,hw$, where $N(B_i)$ is the number of foreground pixels in the i-th sub-block.

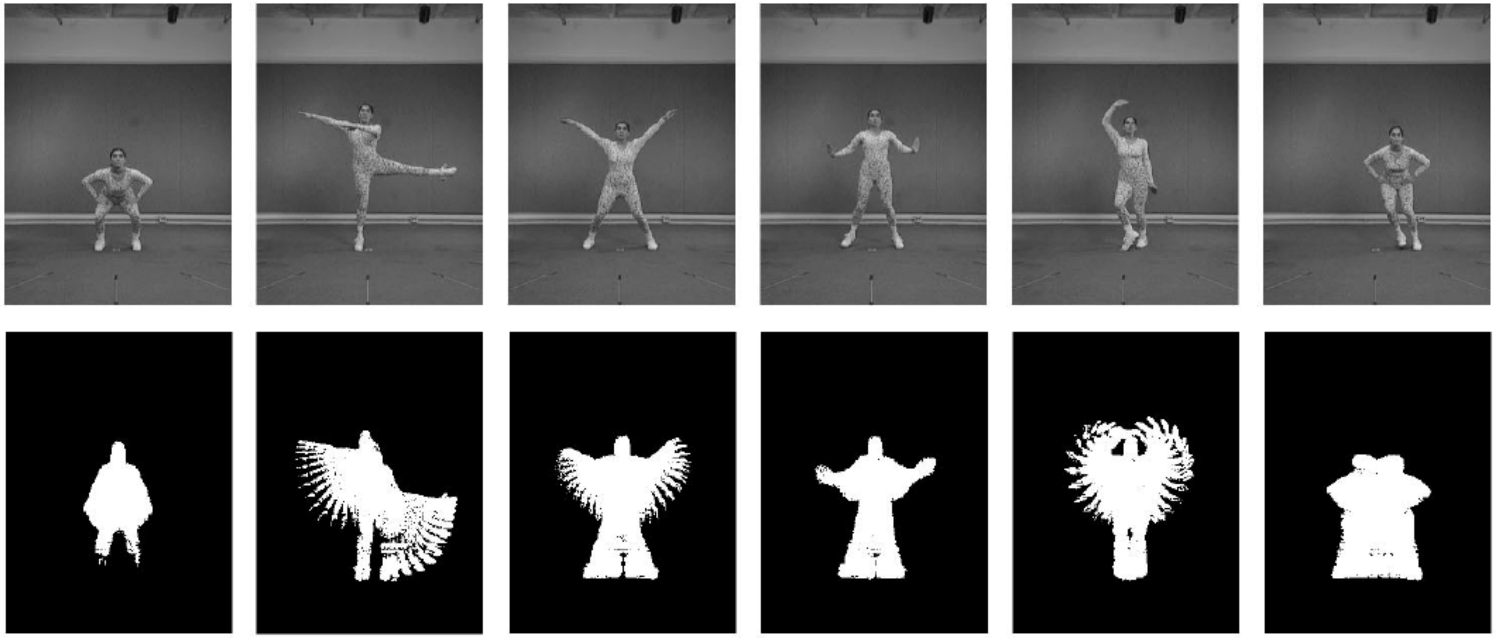
Reference: Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model, CVPR 2007, http://vigir.missouri.edu/~gdesouza/Research/Conference_CDs/IEEE_CVPR_2007/data/papers/0330.pdf

Motion energy image

- Motion energy image: A single binary cumulative motion image, where the shape of the brighter region (location of the motion) can be used to suggest the movement.

$$H_N(x, y, t) = \bigcup_{i=0}^{N-1} D(x, y, t-1)$$

- $I(x, y, t)$: Image sequence
- $D(x, y, t)$: Binary image indicating regions of motion
- $H_N(x, y, t)$: Motion energy image calculated based on past N frames

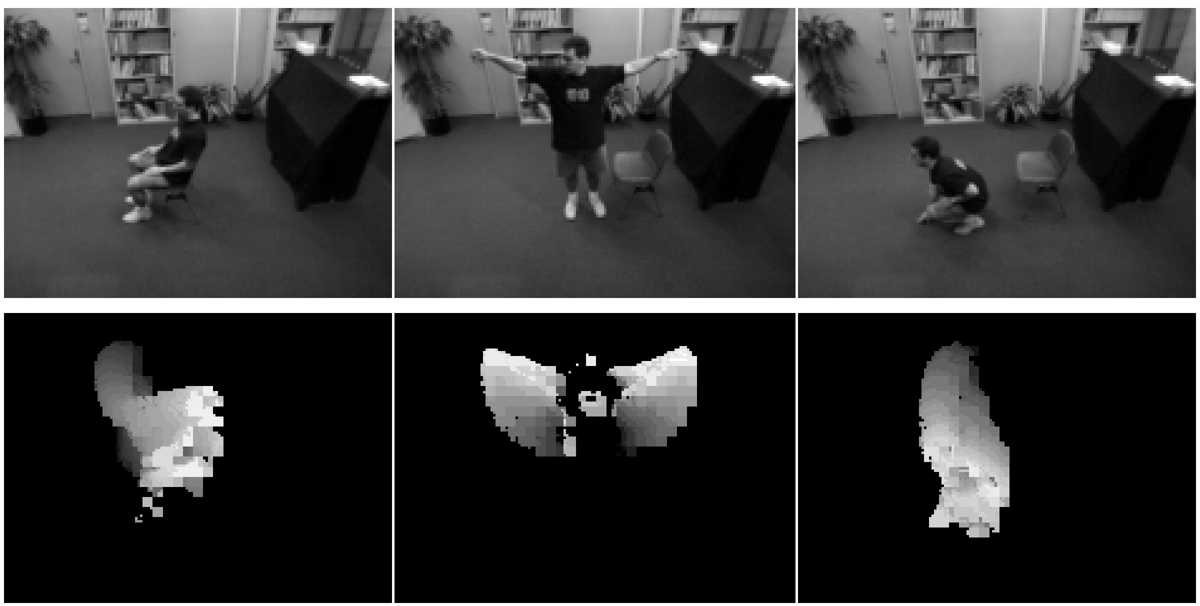


Reference: The recognition of human movement using temporal templates, IEEE Trans on PAMI, 2001, https://www.researchgate.net/publication/3193234_The_recognition_of_human_movement_using_temporal_templates

Motion history image

- Action can be characterized by “changing of shape”.
- Motion history image: Pixel intensity is a function of the motion history at that location, where brighter values correspond to more recent motion.

$$H_N(x,y,t) = \begin{cases} N & \text{if } D(x,y,t) = 1 \\ \max(0, H_N(x,y,t-1)-1) & \text{otherwise} \end{cases}$$

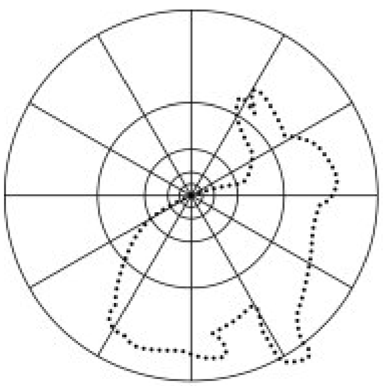


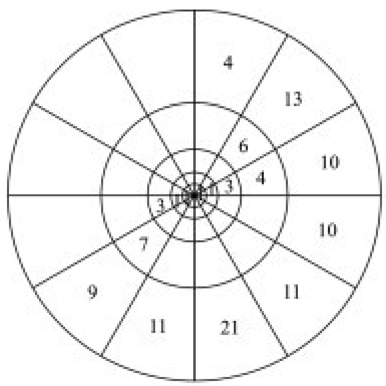
- $I(x,y,t)$: Image sequence
- $D(x,y,t)$: Binary image indicating regions of motion
- $H_N(x,y,t)$: Motion history image calculated based on past N frames

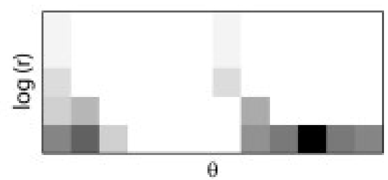
Reference: The recognition of human movement using temporal templates, IEEE Trans on PAMI, 2001, https://www.researchgate.net/publication/3193234_The_recognition_of_human_movement_using_temporal_templates

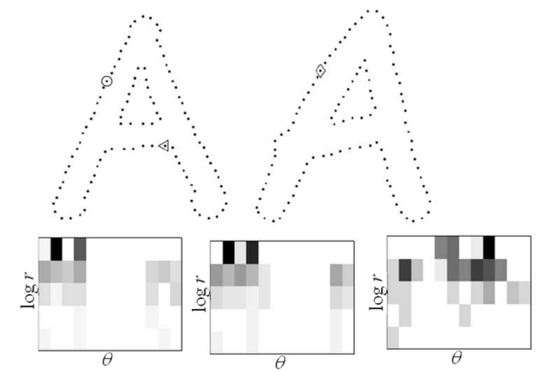
Feature extraction

Shape representation			
Contour-based		Region-based	
Structural	Global	Global	Structural
• Chain code	• Shape descriptors	• Histogram	• Geometrical features









Reference:
- Motion history image: Its variants and applications, https://www.researchgate.net/publication/225968468_Motion_history_image_Its_variants_and_applications;
- Shape descriptors, https://medium.com/machine-learning-world/shape-context-descriptor-and-fast-characters-recognition-c031eac726f9
- Matching with Shape Contexts, https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/shape/sc_digits.html

Action recognition: Milestones

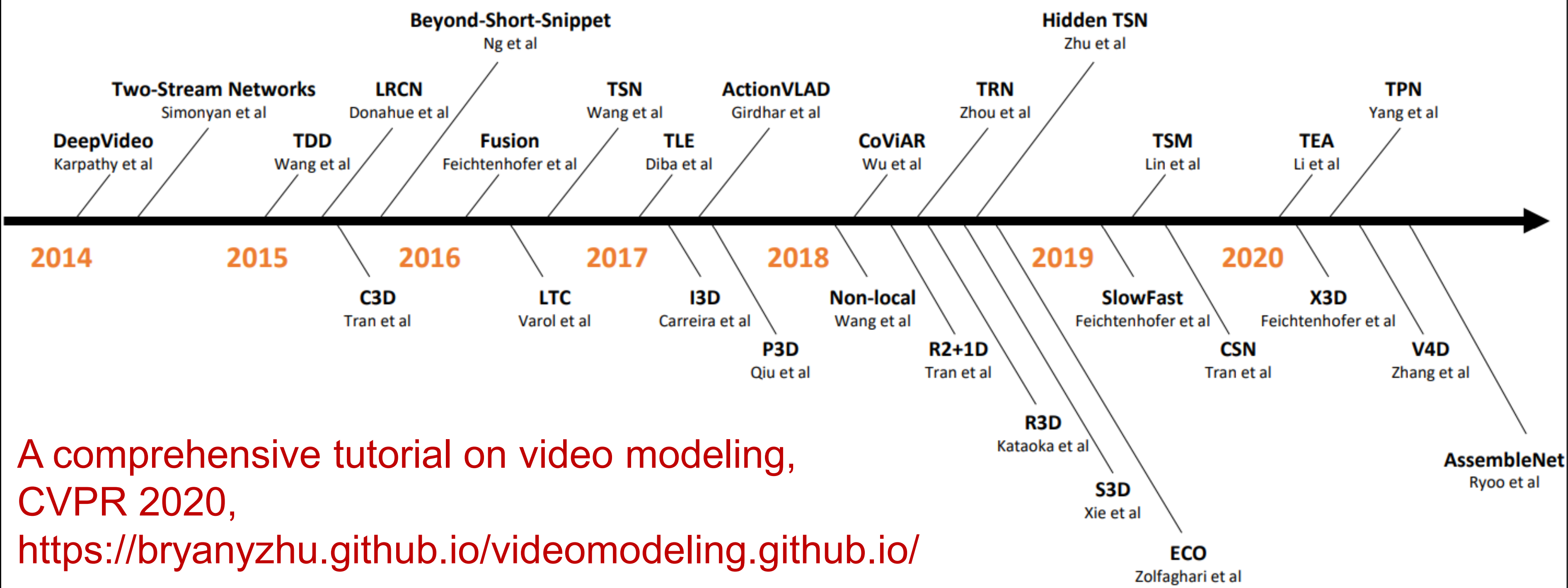
Assumption

- Treat video sequence as a bag of fixed-size clip.
- This is called “trimmed video classification”.
- We can apply sliding window, or automatically segment the long video sequence into short sequence (called action localization)

Key questions

- Where the features should be extracted? Every pixel or key points?
- What features need to be extracted?
- How does the additional motion information influence the predictions of a CNN and how much does it improve performance overall?
- What temporal connectivity pattern in a CNN architecture is best at taking advantage of local motion information present in the video?

Action recognition: Milestones



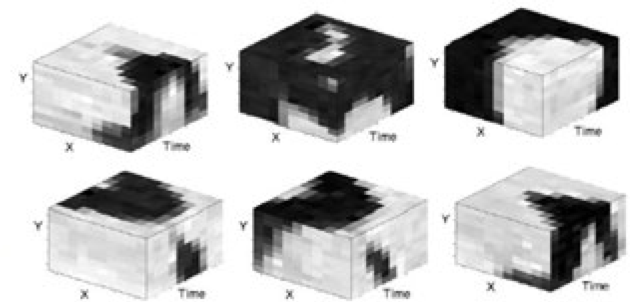
A comprehensive tutorial on video modeling, CVPR 2020, https://bryanyzhu.github.io/videomodeling.github.io/

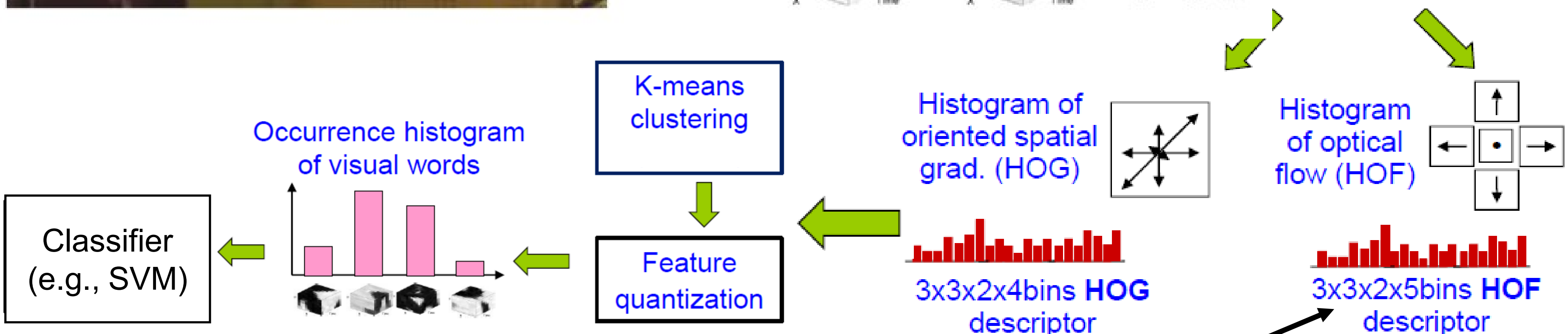
Action recognition: Milestones

Non-deeply learned representation	Sparse feature	HoF	I. Laptev, et al., “Learning realistic human actions from movies,” CVPR 2008, https://www.di.ens.fr/~laptev/actions/
	Dense feature	DT	H. Wang, et al., “Action recognition by dense trajectories,” CVPR 2011, https://hal.inria.fr/hal-00725627v2/document
		iDT	H. Wang, et al., “Action recognition with improved trajectories,” ICCV 2013, https://hal.inria.fr/hal-00873267v2/document.
Deeply learned representation	Single-stream	CNN	A. Karpathy, et al., “Large-scale video classification with convolutional neural networks,” CVPR 2014, https://cs.stanford.edu/people/karpathy/deepvideo/
		LRCN	J. Donahue, et al., "Long-term recurrent convolutional networks for visual recognition and description," CVPR 2015, https://arxiv.org/abs/1411.4389
	Two-stream	Two-stream	K. Simonyan, et al., “Two-stream convolutional networks for action recognition in videos,” NIPS 2014, https://arxiv.org/abs/1406.2199
		Two-stream fusion	C. Feichtenhofer, et al., “Convolutional two-stream network fusion for video action recognition,” CVPR 2016, https://arxiv.org/abs/1604.06573
		TDD	L. Wang, et al., “Action recognition with trajectory-pooled deep-convolutional descriptors,” CVPR 2015, https://arxiv.org/abs/1505.04868
	3D CNN	C3D	D. Tran, et al., “Learning spatiotemporal features with 3D convolutional networks“, ICCV 2015, https://arxiv.org/abs/1412.0767
		i3D	Carreira, et al., “Action recognition? A new model and the Kinetics dataset”, CVPR 2017, https://arxiv.org/abs/1705.07750
		P3D	Qiu et al., Learning spatio-temporal representation with Pseudo-3D residual networks, ICCV 2017, https://arxiv.org/abs/1711.10305
		SlowFast	C. Feichtenhofer, et al., SlowFast networks for video recognition, ICCV 2019, https://arxiv.org/abs/1812.03982

HoF

- Detect local structures in space-time where the image values have significant local variations in both space and time.
- At every keypoint, extract its space-time parch in its neighborhood.
- Each space-time patch is subdivided into a (n_x, n_y, n_t) grid $(3 \times 3 \times 2$ in the paper) of cuboids; for each cuboid we compute histograms of oriented gradient (HoG) and optic flow (HoF).





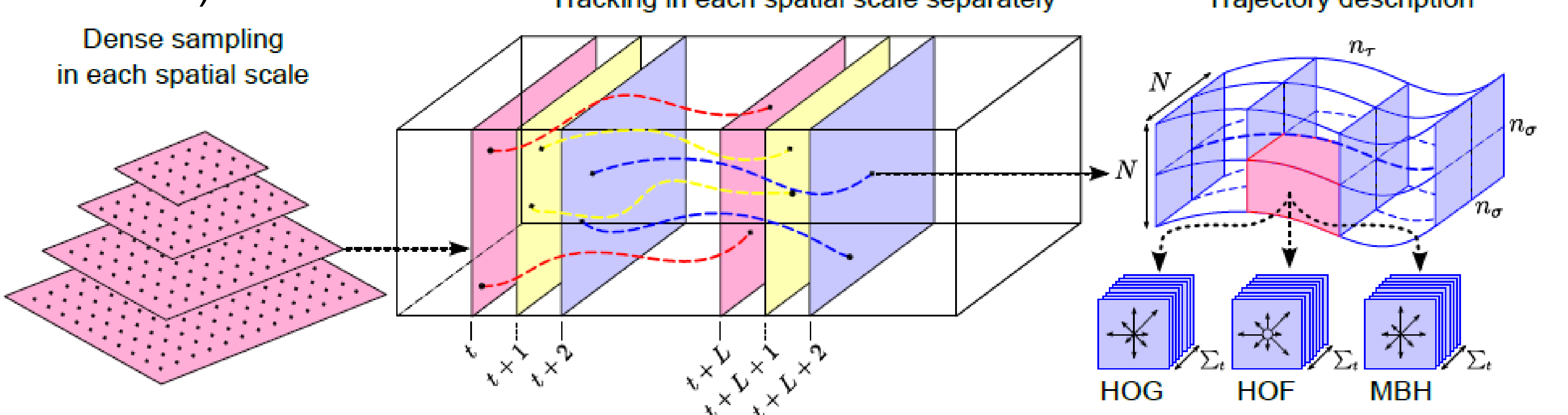
Bag-of-words representation

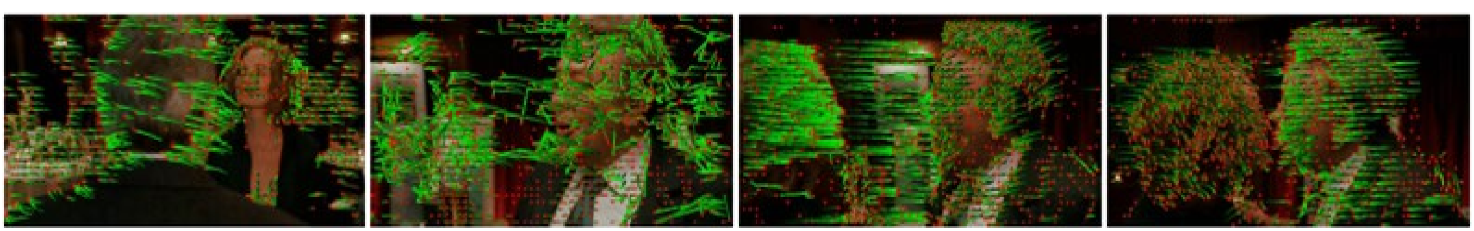
Note: An additional bin is added for histogram of optical flow. It accounts for pixels whose optical flow magnitudes are lower than a threshold (i.e., static).

Reference: I. Laptev, et al., "Learning realistic human actions from movies," CVPR 2008, https://www.di.ens.fr/~laptev/actions/

DT

- Densely sample feature points on a grid spaced by a few pixels. The volume is subdivided into a spatio-temporal grid of size $n_\sigma \times n_\sigma \times n_t$ ($2 \times 2 \times 2$ in the paper). We compute descriptor in each cell of the spatial-temporal grid.
- Both HOG and HOF, orientations are quantized into 8 bins, magnitudes are used for weighting. An additional zero bin is added for HOF (i.e., in total 9 bins). It accounts for pixels whose optical flow magnitudes are lower than a threshold. The final descriptor size is 96 for HOG (i.e., $2 \times 2 \times 3 \times 8$) and 108 for HOF (i.e., $2 \times 2 \times 3 \times 9$).

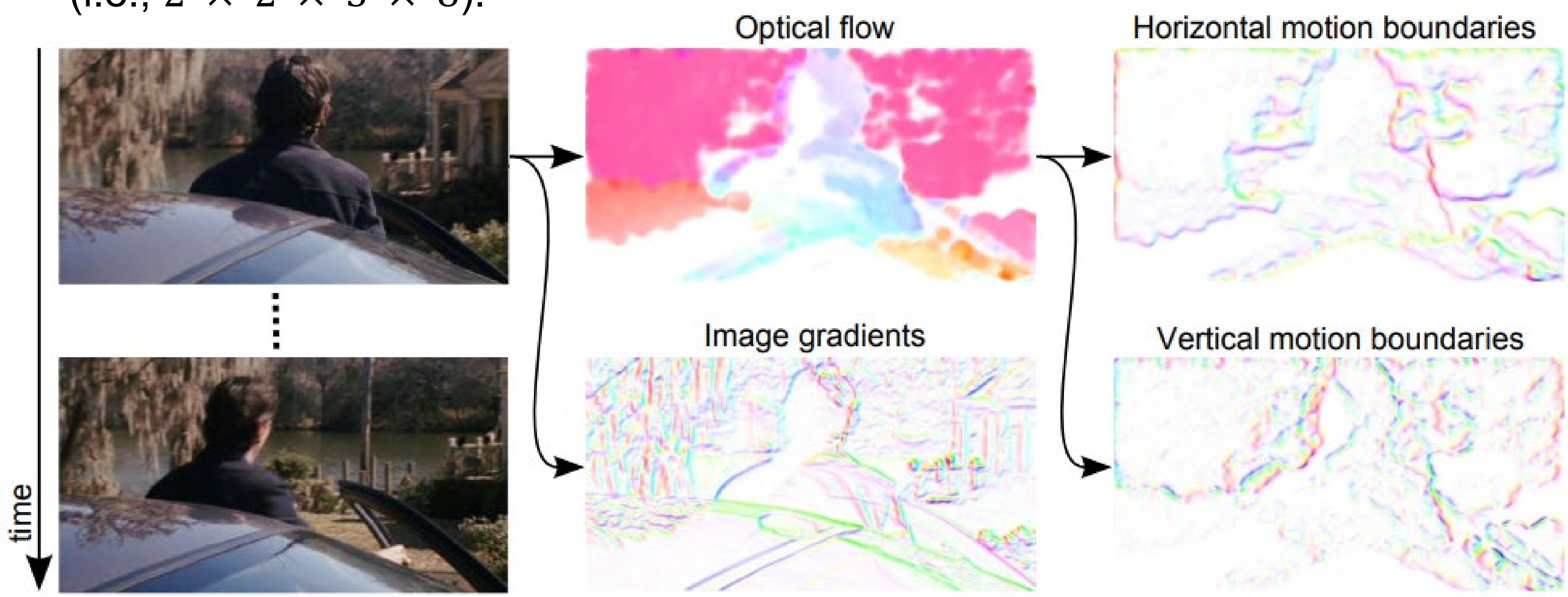




MBH: see next slide

Motion boundary histograms (MBH) descriptor

- Compute optical flow for two consecutive frames.
- Compute derivatives separately for the horizontal and vertical components of the optical flow.
- Compute spatial derivatives for each of them and orientation information is quantized into histograms. The magnitude is used for weighting.
- We obtain a 8-bin histogram for each component, each has a dimension of 96 (i.e., $2 \times 2 \times 3 \times 8$).



Reference: H. Wang, et al., "Action recognition by dense trajectories," CVPR 2011, https://hal.inria.fr/hal-00725627v2/document

iDT

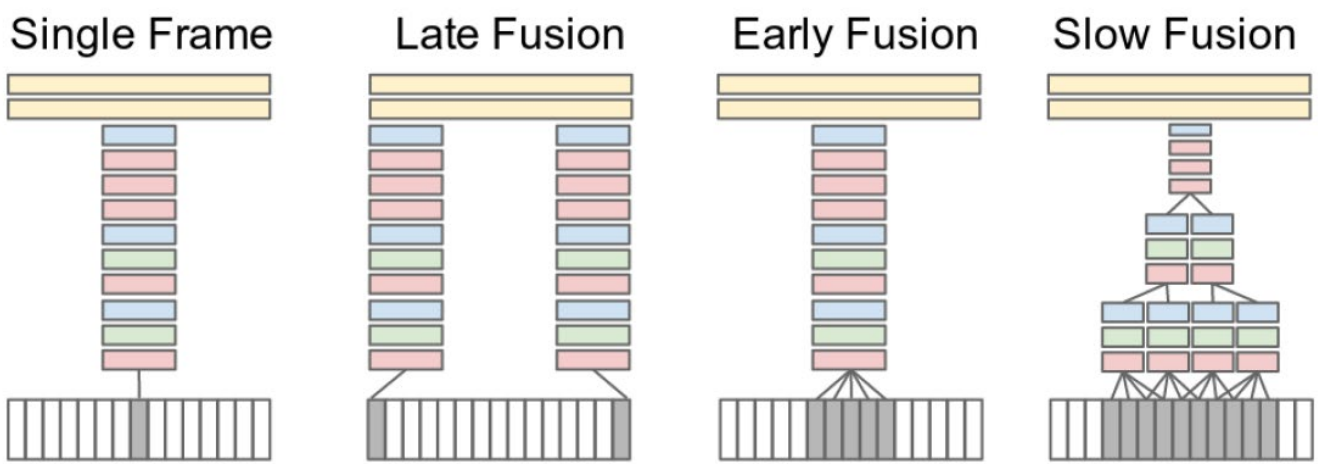
- Estimate camera motion to remove background optical flows.

- Detect human body to remove spurious trajectories.

Reference: H. Wang, et al., “Action recognition with improved trajectories,” ICCV 2013, https://hal.inria.fr/hal-00873267v2/document.

CNN

- Single Frame: Apply image-based CNN on a randomly chosen single frame.
- Early Fusion: Combines information across an entire time window immediately on the pixel level. The filters on the first convolutional layer is modified (e.g., $11 \times 11 \times 3 \times T$) for T frames.
- Late Fusion: Places two separate single-frame networks with shared parameters and then merges the two streams in the first fully connected layer.
- Slow Fusion: Slowly fuses temporal information throughout the network such that higher layers get access to progressively more global information in both spatial and temporal dimensions.



Legend: Red, green and blue boxes indicate convolutional, normalization and pooling layers respectively. In the Slow Fusion model, the depicted columns share parameters.

Reference: A. Karpathy, et al., “Large-scale video classification with convolutional neural networks,” CVPR 2014, https://cs.stanford.edu/people/karpathy/deepvideo/

CNN

Class cores: C

Early fusion

Other layers (Conv, Pooling, Fully-connected)

2D CNN

Input: $T \times 3 \times H \times W$
$\rightarrow$ reshaped to be $3T \times H \times W$
output: $D \times H \times W$

2D CNN | 2D CNN | 2D CNN | 2D CNN | 2D CNN | 2D CNN

Average pooling over space and time

MLP

Class cores: C

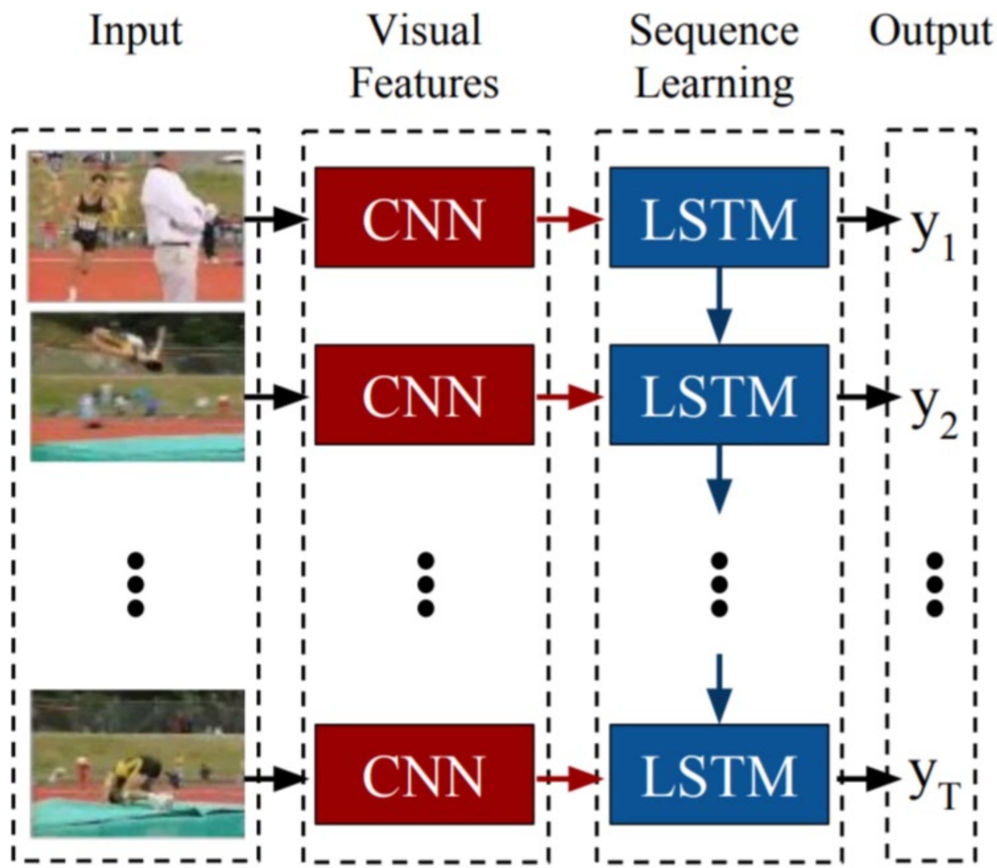
Late fusion

Input to 2D CNN: $3 \times H \times W$ each frame
After 2D CNN to obtain frame feature: $D \times H \times W$
Frame features for T frames: $T \times D \times H \times W$
After space-time pooling to obtain: $D \times 1$

Reference: A. Karpathy, et al., "Large-scale video classification with convolutional neural networks," CVPR 2014, https://cs.stanford.edu/people/karpathy/deepvideo/

LRCN

LRCN processes the (possibly) variable-length visual input (left) with a CNN (middle-left), whose outputs are fed into a stack of recurrent sequence models (LSTMs, middle-right), which finally produce a variable-length prediction (right). Both the CNN and LSTM weights are shared across time, resulting in a representation that scales to arbitrarily long sequences.



Reference: J. Donahue, et al., "Long-term recurrent convolutional networks for visual recognition and description," CVPR 2015, https://arxiv.org/abs/1411.4389

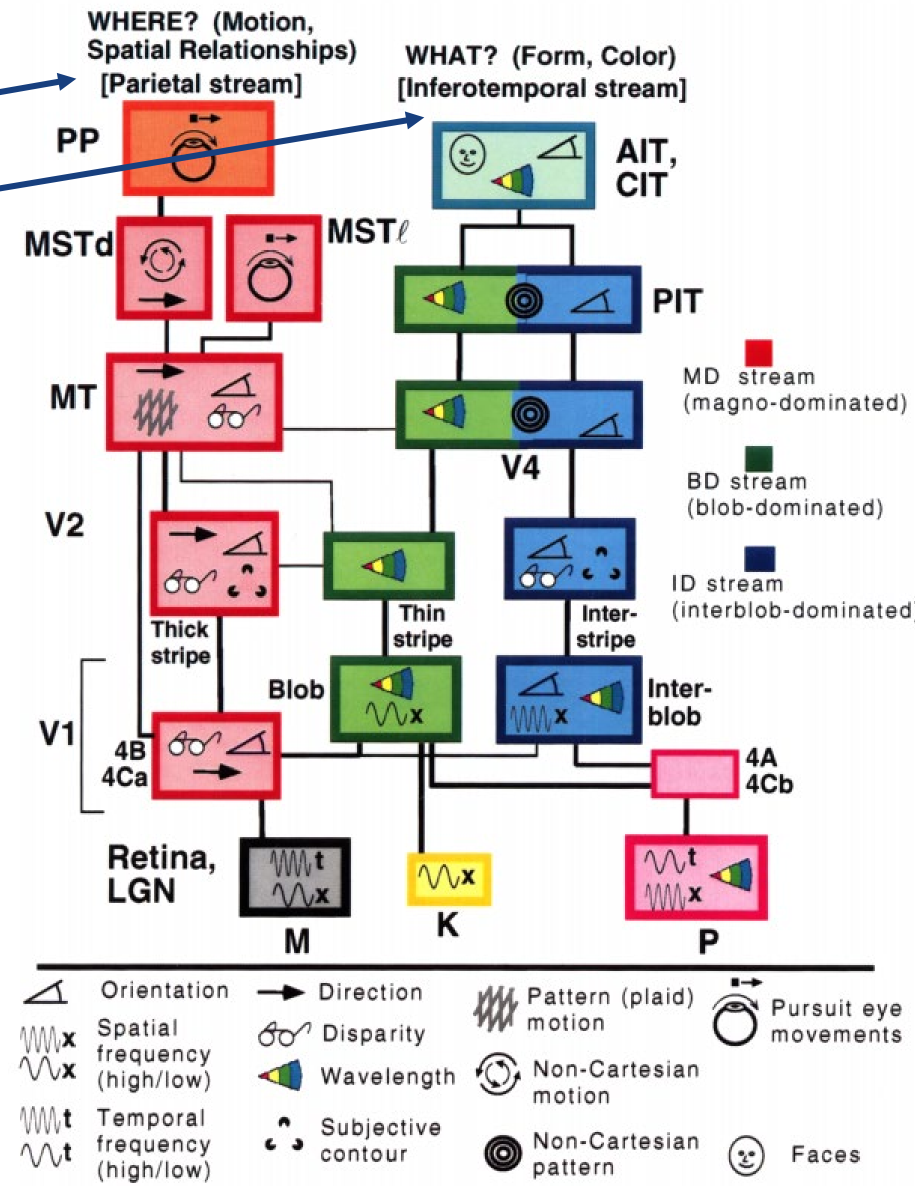
Two-stream

Temporal ConvNet

Spatial ConvNet

The human visual cortex has two hierarchical pathways

- Ventral stream performs object recognition
- Dorsal stream recognized motion and locates objects



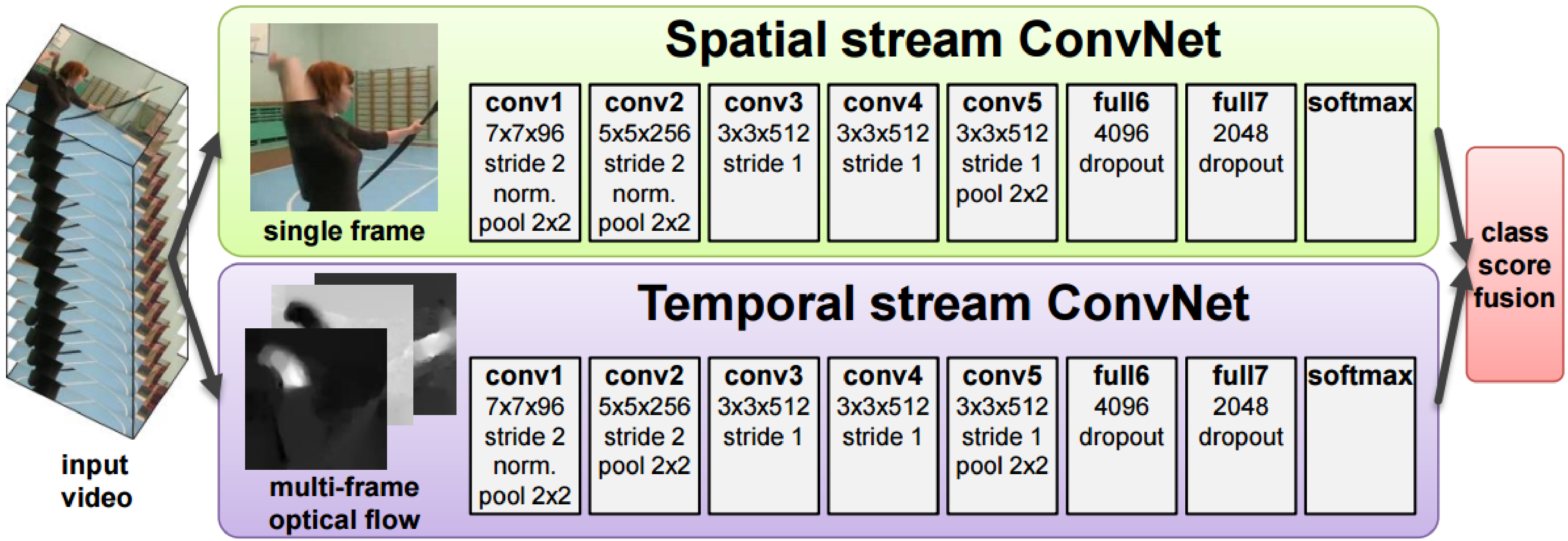
Demo: https://www.youtube.com/watch?v=1F5ICP9SYLU

Reference: David C. Van Essen, Jack L. Gallant, Neural mechanisms of form and motion processing in the primate visual system, Neuron, Vol. 13, Jul. 1994, pp. 1-10

Two-stream

Video: Appearance + Motion

- Single RGB frame: Static appearance
- Multiple motion frames: Optical flow, pixel displacement as motion information
- Spatial stream: Operates on (randomly) individual video frames, effectively performing action recognition from still images.
- Temporal stream: Input to the model is formed by stacking optical flow displacement fields between several consecutive frames.



Reference: K. Simonyan, et al., “Two-stream convolutional networks for action recognition in videos,” NIPS 2014, https://arxiv.org/abs/1406.2199

Two-stream fusion

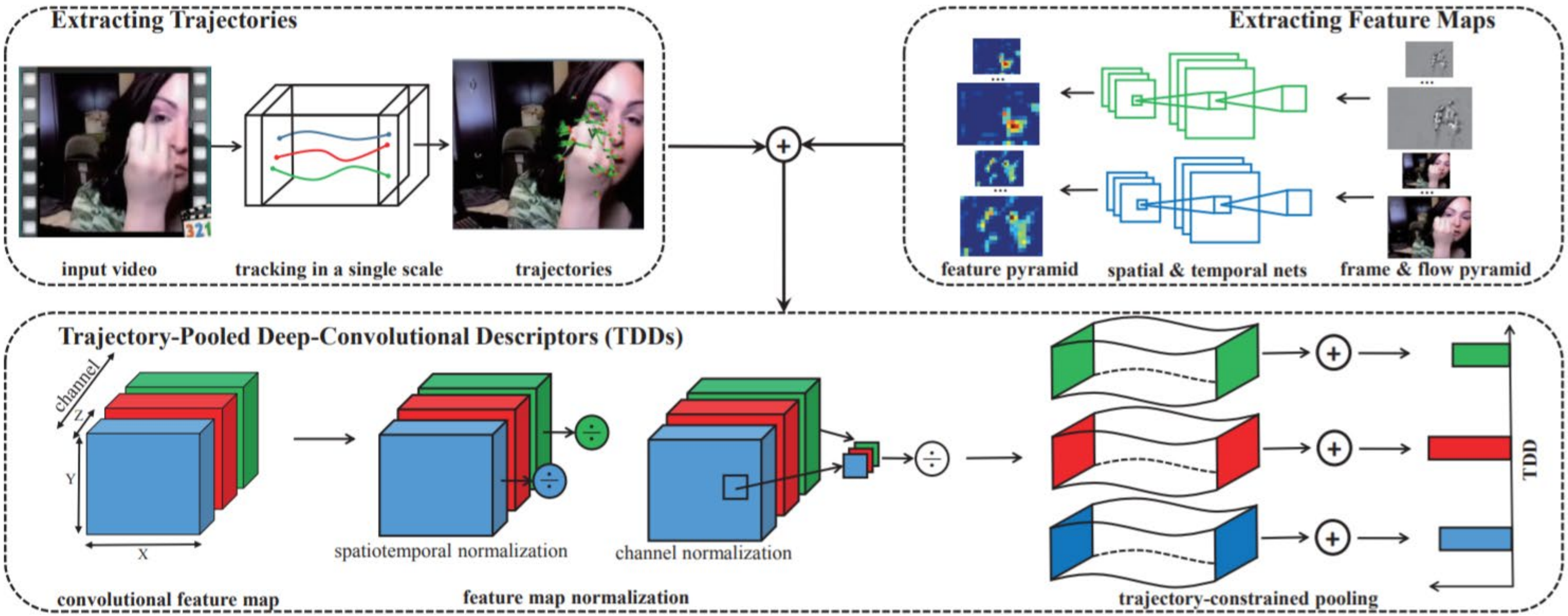
- Objective: Fuse the two networks (at a particular convolutional layer) such that channel responses at the same pixel position are put in correspondence.
- Sum fusion: Computes the sum of two feature maps at the same spatial locations i, j and feature channel d: $y_{i,j,d}^{\text{sum}} = x_{i,j,d}^{a} + x_{i,j,d}^{b}$
- Max fusion: Takes maximum of the two feature map, $y_{i,j,d}^{\max} = \max\{x_{i,j,d}^{a}, x_{i,j,d}^{b}\}$
- Concatenation fusion: Stacks the two feature maps at the same spatial locations i, j across the feature channels d: $y_{i,j,2d}^{\text{cat}} = x_{i,j,d}^{a}$, $y_{i,j,2d-1}^{\text{cat}} = x_{i,j,d}^{b}$
- Bilinear fusion: Computes a matrix outer product of two features at each pixel location, followed by a summation over the locations $y_{i,j,d}^{\text{bil}} = \sum_i \sum_j \mathbf{x}_{i,j}^{a\,T} \mathbf{x}_{i,j}^{b}$

Input	Method	Output
Two feature maps, each has a dimension of $H \times W \times D$	Sum fusion	$H \times W \times D$
	Max fusion	$H \times W \times D$
	Concatenation fusion	$H \times W \times 2D$
	Bilinear fusion	• Outer product at each pixel location: $H \times W \times D \times D$ • Summation over pixel locations: $D \times D$

Reference: C. Feichtenhofer, et al., "Convolutional two-stream network fusion for video action recognition," CVPR 2016, https://arxiv.org/abs/1604.06573

TDD

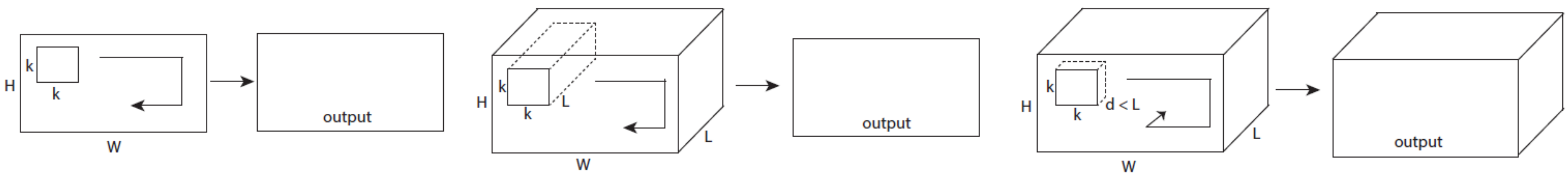
- Treat the learned two-stream ConvNets as generic feature extractors, and use them to obtain multi-scale convolutional feature maps for each video.
- Detect a set of point trajectories with the method of dense trajectories.
- Pool the local ConvNet responses over the spatiotemporal tubes centered at the trajectories, based on convolutional feature maps and trajectories.



Reference: L. Wang, et al., "Action recognition with trajectory-pooled deep-convolutional descriptors," CVPR 2015, https://arxiv.org/abs/1505.04868

c3d

- 3D ConvNets are more suitable for spatiotemporal feature learning compared to 2D ConvNets.
- A homogeneous architecture with small $3 \times 3 \times 3$ convolution kernels in all layers
- The learned features, namely C3D (Convolutional 3D) can be further integrated with other classifiers.



(a) 2D convolution (b) 2D convolution on multiple frames (c) 3D convolution

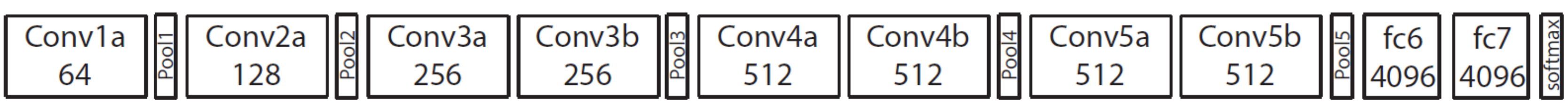


Figure 3. **C3D architecture**. C3D net has 8 convolution, 5 max-pooling, and 2 fully connected layers, followed by a softmax output layer. All 3D convolution kernels are $3 \times 3 \times 3$ with stride 1 in both spatial and temporal dimensions. Number of filters are denoted in each box. The 3D pooling layers are denoted from `pool1` to `pool5`. All pooling kernels are $2 \times 2 \times 2$, except for `pool1` is $1 \times 2 \times 2$. Each fully connected layer has 4096 output units.

Reference: D. Tran, et al., “Learning spatiotemporal features with 3D convolutional networks“, ICCV 2015, https://arxiv.org/abs/1412.0767

c3d

```
c3d_model = Sequential()
c3d_model.add(Conv3D(32, kernel_size=(3, 3, 3), input_shape=(x_train.shape[1:]), padding='same'))
c3d_model.add(Activation('relu'))
c3d_model.add(Conv3D(32, kernel_size=(3, 3, 3), padding='same'))
c3d_model.add(Activation('softmax'))
c3d_model.add(MaxPooling3D(pool_size=(3, 3, 3), padding='same'))
c3d_model.add(Dropout(0.25))

c3d_model.add(Conv3D(64, kernel_size=(3, 3, 3), padding='same'))
c3d_model.add(Activation('relu'))
c3d_model.add(Conv3D(64, kernel_size=(3, 3, 3), padding='same'))
c3d_model.add(Activation('softmax'))
c3d_model.add(MaxPooling3D(pool_size=(3, 3, 3), padding='same'))
c3d_model.add(Dropout(0.25))

c3d_model.add(Flatten(name='flatten_feature'))
c3d_model.add(Dense(512, activation='sigmoid'))
c3d_model.add(Dropout(0.2))
c3d_model.add(Dense(nb_classes, activation='softmax'))

c3d_model.compile(loss=categorical_crossentropy, optimizer=Adam(), metrics=['accuracy']
c3d_model.summary()
```

Layer (type)	Output Shape	Param #
conv3d_1 (Conv3D)	(None, 32, 32, 10, 32)	896
activation_1 (Activation)	(None, 32, 32, 10, 32)	0
conv3d_2 (Conv3D)	(None, 32, 32, 10, 32)	27680
activation_2 (Activation)	(None, 32, 32, 10, 32)	0
max_pooling3d_1 (MaxPooling3	(None, 11, 11, 4, 32)	0
dropout_1 (Dropout)	(None, 11, 11, 4, 32)	0
conv3d_3 (Conv3D)	(None, 11, 11, 4, 64)	55360
activation_3 (Activation)	(None, 11, 11, 4, 64)	0
conv3d_4 (Conv3D)	(None, 11, 11, 4, 64)	110656
activation_4 (Activation)	(None, 11, 11, 4, 64)	0
max_pooling3d_2 (MaxPooling3	(None, 4, 4, 2, 64)	0
dropout_2 (Dropout)	(None, 4, 4, 2, 64)	0
flatten_feature (Flatten)	(None, 2048)	0
dense_1 (Dense)	(None, 512)	1049088
dropout_3 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 11)	5643

Total params: 1,249,323
Trainable params: 1,249,323
Non-trainable params: 0

The c3d model used in our workshop

$$(3 \times 3 \times 3 + 1) \times 32 = 896$$

$$(3 \times 3 \times 3 \times 32 + 1) \times 32 = 27680$$

$$(3 \times 3 \times 3 \times 32 + 1) \times 64 = 55360$$

$$(3 \times 3 \times 3 \times 64 + 1) \times 64 = 110656$$

$$(2048 + 1) \times 512 = 1049088$$

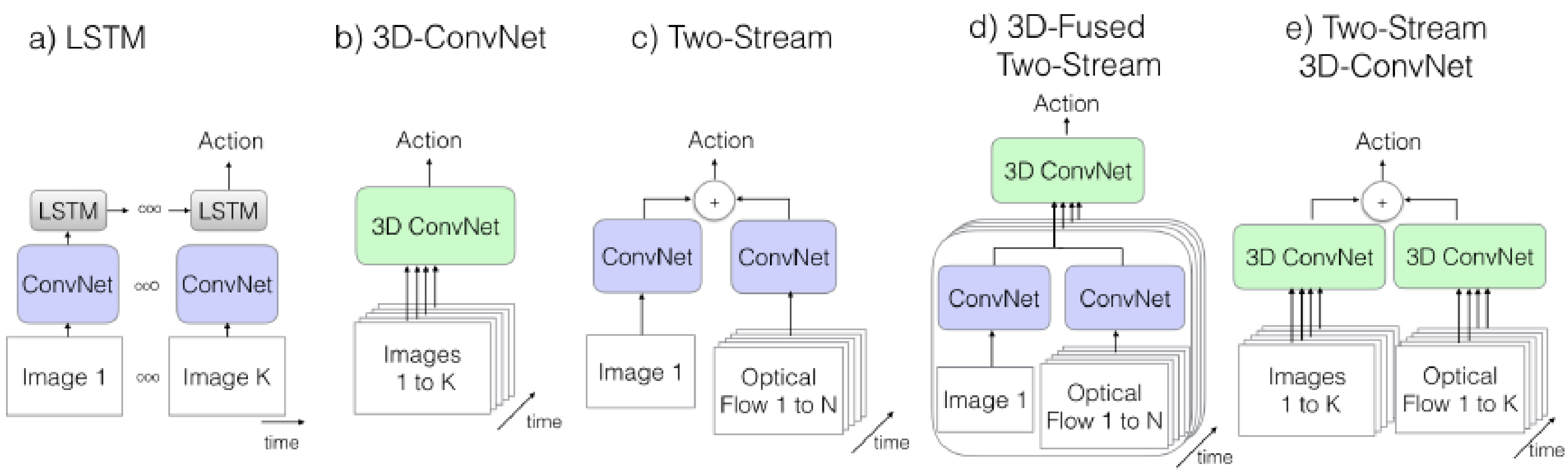
$$(512 + 1) \times 11 = 5643$$

c3d

Method	Layer	Size ($C \times T \times H \times W$)
Early Fusion (2D CNN)	Input	$3 \times 20 \times 64 \times 64$
	Conv2D ($3 \times 3,\ 3 \times 20 \to 12$)	$12 \times 64 \times 64$
	Pool2D (4×4)	$12 \times 16 \times 16$
	Conv2D ($3 \times 3,\ \ 12 \to 24$)	$24 \times 64 \times 64$
	GlobalAvgPool	$24 \times 1 \times 1$
Late Fusion (2D CNN)	Input	$3 \times 20 \times 64 \times 64$
	Conv2D ($3 \times 3,\ 3 \to 12$)	$12 \times 20 \times 64 \times 64$
	Pool2D (4×4)	$3 \times 20 \times 16 \times 16$
	Conv2D ($3 \times 3,\ \ 12 \to 24$)	$24 \times 20 \times 64 \times 64$
	GlobalAvgPool	$24 \times 1 \times 1 \times 1$
3D CNN	Input	$3 \times 20 \times 64 \times 64$
	Conv3D ($3 \times 3 \times 3,\ 3 \to 12$)	$12 \times 20 \times 64 \times 64$
	Pool3D ($4 \times 4 \times 4$)	$12 \times 5 \times 16 \times 16$
	Conv3D ($3 \times 3 \times 3,\ \ 12 \to 24$)	$24 \times 5 \times 64 \times 64$
	GlobalAvgPool	$24 \times 1 \times 1 \times 1$

i3D

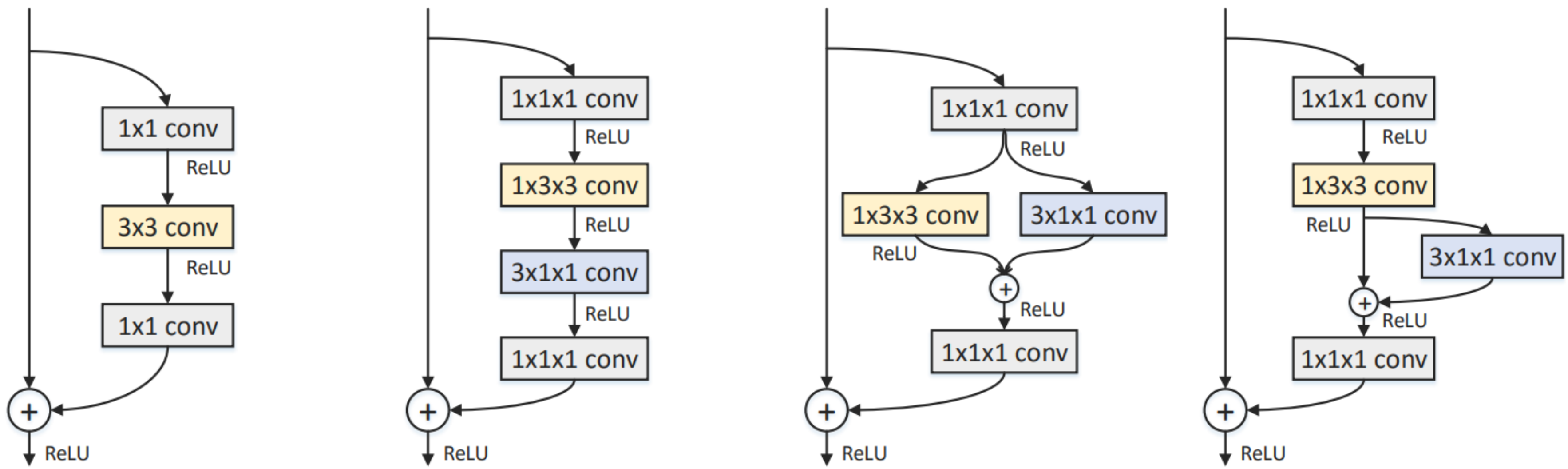
Inflated 3D ConvNet (I3D) is based on 2D ConvNet inflation: Filters and pooling kernels of very deep image classification ConvNets are expanded into 3D, making it possible to learn seamless spatio-temporal feature extractors from video while leveraging successful ImageNet architecture designs and even their parameters. Replace 2D CNN with 3D CNN version. Duplicate pre-trained 2D CNN parameters to be 3D CNN parameters.



Reference: Carreira, et al., “Action recognition? A new model and the Kinetics dataset”, CVPR 2017, https://arxiv.org/abs/1705.07750

P3D

Idea: Propose a *Pseudo-3D Residual Net* (P3D ResNet). Convert $3 \times 3 \times 3$ convolutions with $1 \times 3 \times 3$ convolutional filters on spatial domain (equivalent to 2D CNN) plus $3 \times 1 \times 1$ convolutions to construct temporal connections on adjacent feature maps in time.



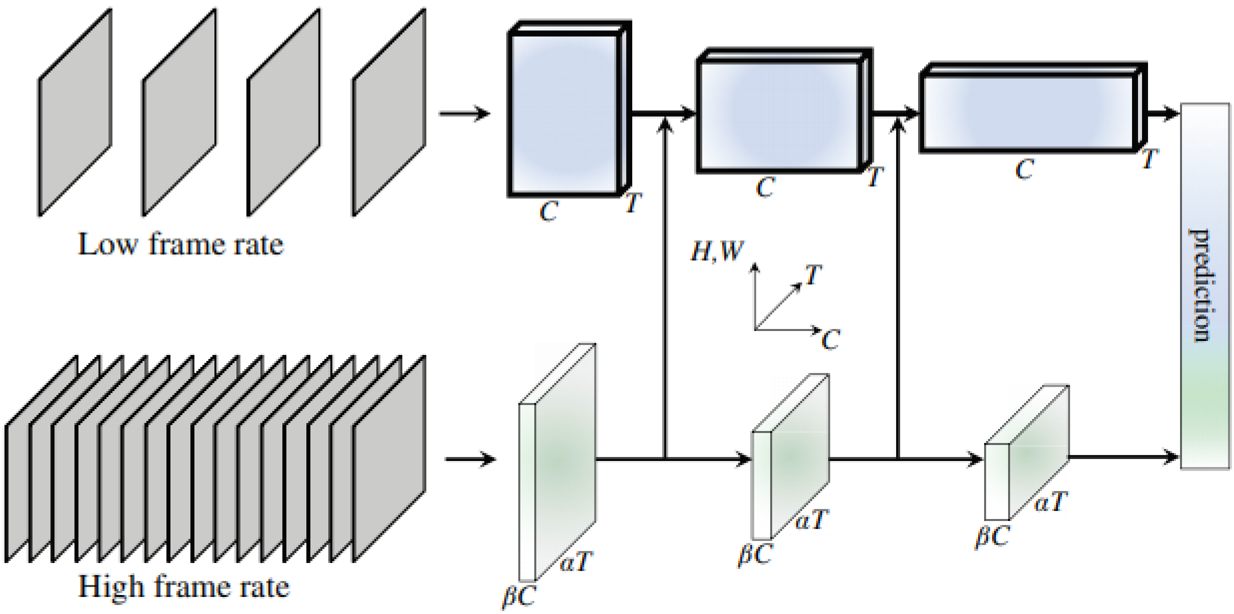
Conventional ResNet   Proposed P3D ResNet with three variants

Reference: Qiu et al., Learning Spatio-Temporal Representation with Pseudo-3D Residual Networks, ICCV 2017, https://arxiv.org/abs/1711.10305

SlowFast

Two streams with different frame rates

- A low frame rate, low temporal resolution Slow pathway
- A high frame rate, higher temporal resolution Fast pathway. The Fast pathway is lightweight by using a fraction of convolutional channels.



α: Adjust frame rate

β: Adjust convolutional channels (for consideration of computational complexity)

Reference:

- C. Feichtenhofer, et al., SlowFast Networks for Video Recognition, ICCV 2019, https://arxiv.org/abs/1812.03982
- https://github.com/facebookresearch/SlowFast

Topics

- Action recognition
- Workshop: Build action recognition systems

Workshop

- Dataset: UCF11 Dataset, https://www.crcv.ucf.edu/data/UCF_YouTube_Action.php.
- It contains 11 action categories: basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog.

Workshop

- Exercise 1: Perform action recognition using histogram of optical flow
- Exercise 2: Perform action recognition using C3D deep learning approach
- Exercise 3: Perform action recognition using C3D+SVM

Submission guideline:
Rename your *.ipynb file to be your name and submit to LumiNUS.

Thank you!

Dr TIAN Jing
Email: tianjing@nus.edu.sg