# HUMAN ACTION RECOGNITION

# Human Action Recognition

1. Hand crafted feature + Shallow classifier

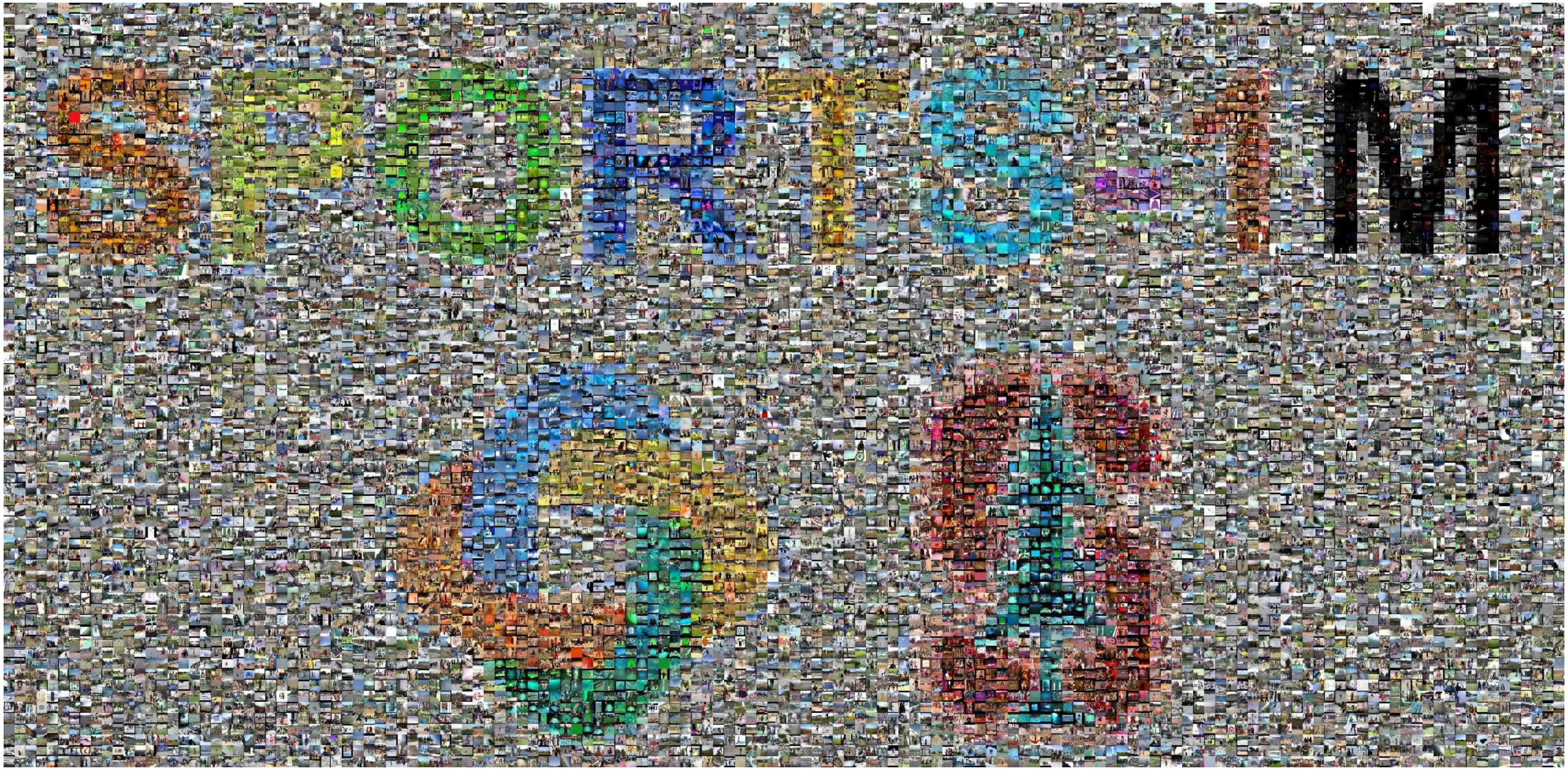
2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

3. 3D CNN
   - Input is a small chunk of video

4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - Two stream (single frame + multi-frame optical flow)

5. ConvNet + RNN
   - 3D CNN + RNN
   - CNN + RNN

# Sports-1M Dataset



# Sports-1M Dataset

- a new dataset of 1 million YouTube videos belonging to 487 classes

# Human Action Recognition

1. **Hand crafted feature + Shallow classifier**

2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

3. 3D CNN
   - Input is a small chunk of video

4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - Two stream (single frame + multi-frame optical flow)

5. ConvNet + RNN
   - 3D CNN + RNN
   - CNN + RNN

# Feature-based approaches to Activity Recognition

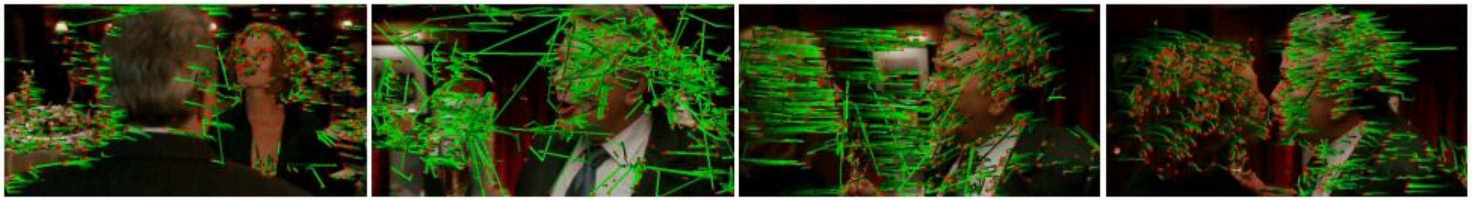
- Dense trajectories and motion boundary descriptors for action recognition

  https://hal.inria.fr/hal-00725627/document

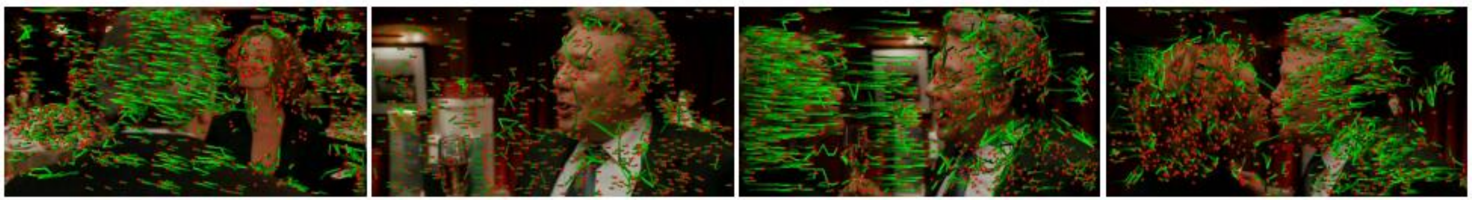
- Action Recognition with Improved Trajectories

  https://hal.inria.fr/hal-00873267v2/document

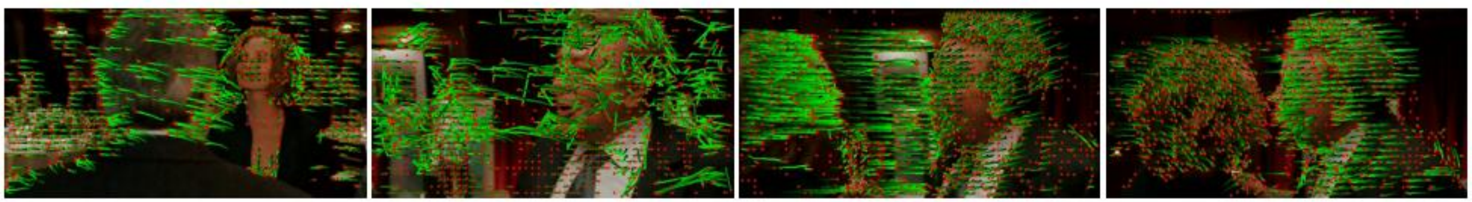
# Trajectories for a "kiss" action



KLT trajectories



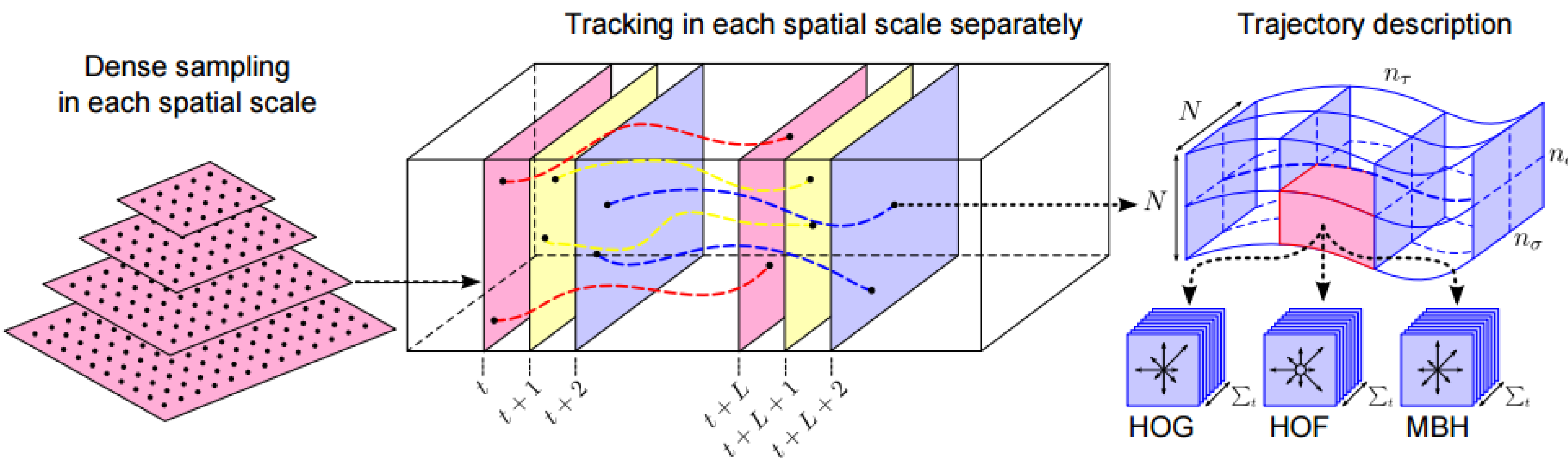
SIFT trajectories



Dense trajectories

Figure 1: Visualization of KLT, SIFT and dense trajectories for a "kiss" action. Red dots indicate the point positions in the current frame. Compared to KLT trajectories, dense trajectories are more robust to fast irregular motions, in particular at shot boundaries (second column). SIFT trajectories can also handle shot boundaries, but are not able to capture the complex motion patterns accurately.

# Feature-based approaches to Activity Recognition



# Human Action Recognition

1. Hand crafted feature + Shallow classifier

2. **Human localization + Hand crafted features + 3D CNN**
   - Input is a small chunk of video

3. 3D CNN
   - Input is a small chunk of video

4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - Two stream (single frame + multi-frame optical flow)

5. ConvNet + RNN
   - 3D CNN + RNN
   - CNN + RNN

# Dataset

- TRECVID 2008
  - 49 hours videos captured at the London Gatwick Airport using 5 cameras
    - 720x576 at 25fps
- 3 action classes
  - CellToEar/ObjectPut/Pointing
- Head location:
  - Human detection + a detection-driven tracker

3D Convolutional Neural Networks for Human Action Recognition, Ji et al., 2010

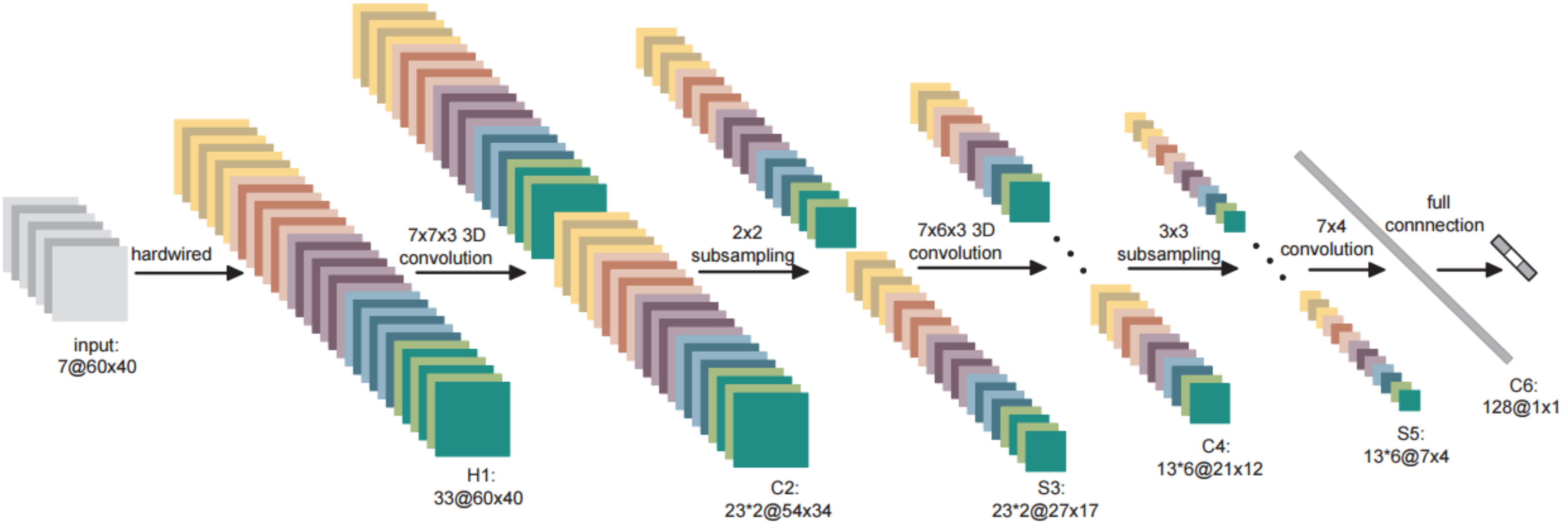
# Dataset

*Figure 4.* Sample human detection and tracking results from camera numbers 1, 2, 3, and 5, respectively from left to right.

*Table 1.* The number of samples in each class on each of the five dates extracted from the TRECVID 2008 development data set. The total number of samples on each date and in each class are also shown.

| Date\Class | CellToEar | ObjectPut | Pointing | Negative | Total |
|---|---|---|---|---|---|
| 20071101 | 2692 | 1349 | 7845 | 20056 | 31942 |
| 20071106 | 1820 | 3075 | 8533 | 22095 | 35523 |
| 20071107 | 465 | 3621 | 8708 | 19604 | 32398 |
| 20071108 | 4162 | 3582 | 11561 | 35898 | 55203 |
| 20071112 | 4859 | 5728 | 18480 | 51428 | 80495 |
| Total | 13998 | 17355 | 55127 | 149081 | 235561 |

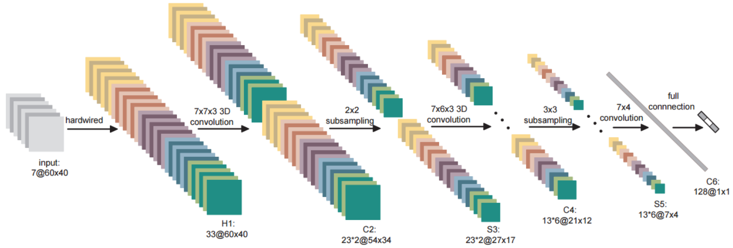
# Spatio-Temporal ConvNet




Ji et al. “3D Convolutional Neural Networks for Human Action Recognition”

# 3D convolution

| | #(parameters) | #(parameters) | 비고 |
|---|---|---|---|
| H1-C2 | (7x7x3+1)x5x2 | 1,480 | 7x7x3 filter |
| C2-S3 | 23x2x2 | 92 | 2 para. per samp. |
| S3-C4 | (7x6x3+1)x5x6 | 3,810 | |
| C4-S5 | 13x6x2 | 156 | 2 para. per samp. |
| S5-C6 | (7x4+1)x78x128 | 289,536 | Conv+FC layer |
| C6-output | 128x3 | 384 | 3 classes |
| Total | | 295,458 | |




- Hard wired feature maps
  - Gray, gradient-x, gradient-y, optical flow-x, optical flow-y (5)

# Human Action Recognition

1. Hand crafted feature + Shallow classifier

2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

3. **3D CNN**
   - Input is a small chunk of video

4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - Two stream (single frame + multi-frame optical flow)

5. ConvNet + RNN
   - 3D CNN + RNN
   - CNN + RNN

# LEARNING SPATIOTEMPORAL FEATURES WITH 3D CONVOLUTIONAL NETWORKS

Tran et al. 2015

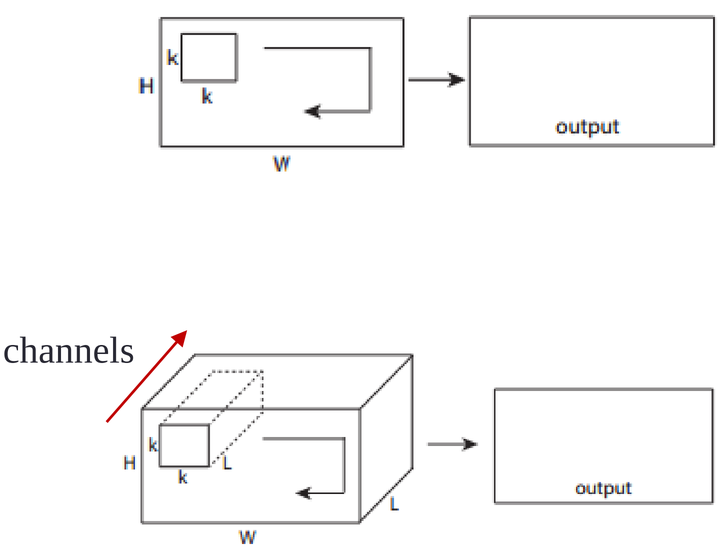
# UCF101- action recognition dataset

- 5 categories
  - 1)Human-Object Interaction 2) Body-Motion Only 3) Human-Human Interaction 4) Playing Musical Instruments 5) Sports.

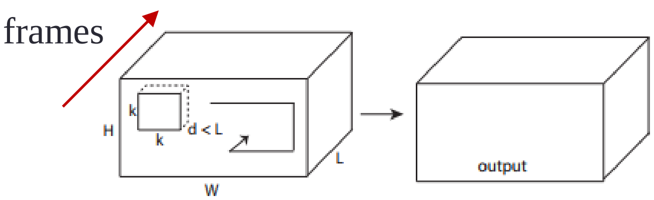
# 101 actions

- Apply Eye Makeup, Apply Lipstick, Archery, Baby Crawling, Balance Beam, Band Marching, Baseball Pitch, Basketball Shooting, Basketball Dunk, Bench Press, Biking, Billiards Shot, Blow Dry Hair, Blowing Candles, Body Weight Squats, Bowling, Boxing Punching Bag, Boxing Speed Bag, Breaststroke, Brushing Teeth, Clean and Jerk, Cliff Diving, Cricket Bowling, Cricket Shot, Cutting In Kitchen, Diving, Drumming, Fencing, Field Hockey Penalty, Floor Gymnastics, Frisbee Catch, Front Crawl, Golf Swing, Haircut, Hammer Throw, Hammering, Handstand Pushups, Handstand Walking, Head Massage, High Jump, Horse Race, Horse Riding, Hula Hoop, Ice Dancing, Javelin Throw, Juggling Balls, Jump Rope, Jumping Jack, Kayaking, Knitting, Long Jump, Lunges, Military Parade, Mixing Batter, Mopping Floor, Nun chucks, Parallel Bars, Pizza Tossing, Playing Guitar, Playing Piano, Playing Tabla, Playing Violin, Playing Cello, Playing Daf, Playing Dhol, Playing Flute, Playing Sitar, Pole Vault, Pommel Horse, Pull Ups, Punch, Push Ups, Rafting, Rock Climbing Indoor, Rope Climbing, Rowing, Salsa Spins, Shaving Beard, Shotput, Skate Boarding, Skiing, Skijet, Sky Diving, Soccer Juggling, Soccer Penalty, Still Rings, Sumo Wrestling, Surfing, Swing, Table Tennis Shot, Tai Chi, Tennis Swing, Throw Discus, Trampoline Jumping, Typing, Uneven Bars, Volleyball Spiking, Walking with a dog, Wall Pushups, Writing On Board, Yo Yo.

# 3D convolution




2D convolution




3D convolution

# 3D convolution

## tf.nn.conv2d(input, filter, strides, padding, use_cudnn_on_gpu=None, data_format=None, name=None)

Computes a 2-D convolution given 4-D `input` and `filter` tensors.

Given an input tensor of shape `[batch, in_height, in_width, in_channels]` and a filter / kernel tensor of shape `[filter_height, filter_width, in_channels, out_channels]`, this op performs the following:

1. Flattens the filter to a 2-D matrix with shape `[filter_height * filter_width * in_channels, output_channels]`.
2. Extracts image patches from the input tensor to form a virtual tensor of shape `[batch, out_height, out_width, filter_height * filter_width * in_channels]`.
3. For each patch, right-multiplies the filter matrix and the image patch vector.

In detail, with the default NHWC format,

```
output[b, i, j, k] =
    sum_{di, dj, q} input[b, strides[1] * i + di, strides[2] * j + dj, q] *
                    filter[di, dj, q, k]
```

Must have `strides[0] = strides[3] = 1`. For the most common case of the same horizontal and vertices strides, `strides = [1, stride, stride, 1]`.

Args:

- `input`: A `Tensor`. Must be one of the following types: `half`, `float32`, `float64`.
- `filter`: A `Tensor`. Must have the same type as `input`.
- `strides`: A list of `ints`. 1-D of length 4. The stride of the sliding window for each dimension of `input`. Must be in the same order as the dimension specified with format.
- `padding`: A `string` from: `"SAME", "VALID"`. The type of padding algorithm to use.
- `use_cudnn_on_gpu`: An optional `bool`. Defaults to `True`.
- `data_format`: An optional `string` from: `"NHWC", "NCHW"`. Defaults to `"NHWC"`. Specify the data format of the input and output data. With the default format "NHWC", the data is stored in the order of: [batch, in_height, in_width, in_channels]. Alternatively, the format could be "NCHW", the data storage order of: [batch, in_channels, in_height, in_width].
- `name`: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `input`.

## tf.nn.conv3d(input, filter, strides, padding, name=None)

Computes a 3-D convolution given 5-D `input` and `filter` tensors.

In signal processing, cross-correlation is a measure of similarity of two waveforms as a function of a time-lag applied to one of them. This is also known as a sliding dot product or sliding inner-product.

Our Conv3D implements a form of cross-correlation.

Args:

- `input`: A `Tensor`. Must be one of the following types: `float32`, `float64`, `int64`, `int32`, `uint8`, `uint16`, `int16`, `int8`, `complex64`, `complex128`, `qint8`, `quint8`, `qint32`, `half`. Shape `[batch, in_depth, in_height, in_width, in_channels]`.
- `filter`: A `Tensor`. Must have the same type as `input`. Shape `[filter_depth, filter_height, filter_width, in_channels, out_channels]`. `in_channels` must match between `input` and `filter`.
- `strides`: A list of `ints` that has length `>= 5`. 1-D tensor of length 5. The stride of the sliding window for each dimension of `input`. Must have `strides[0] = strides[4] = 1`.
- `padding`: A `string` from: `"SAME", "VALID"`. The type of padding algorithm to use.
- `name`: A name for the operation (optional).

Returns:

A `Tensor`. Has the same type as `input`.

# Learning Spatiotemporal Features with 3D Convolutional Networks

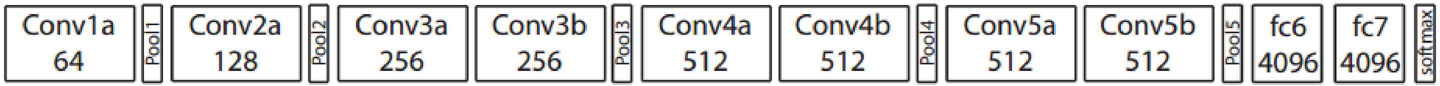
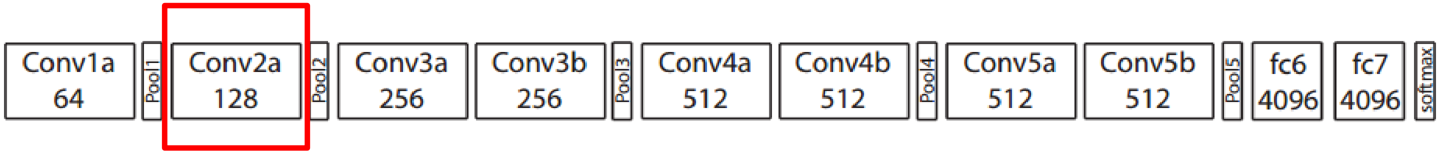



Figure 3. **C3D architecture**. C3D net has 8 convolution, 5 max-pooling, and 2 fully connected layers, followed by a softmax output layer. All 3D convolution kernels are $3 \times 3 \times 3$ with stride 1 in both spatial and temporal dimensions. Number of filters are denoted in each box. The 3D pooling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


1 ice_skating:0.98
2 speed_skating:0.01


darts:0.23
fives:0.05
badminton:0.05
juggling_club:0.03
skeleton_(sport):0.03
SUPER DARTS 2011
SINGLE 19
TRIPLE 19
SINGLE 18
REMOVE DARTS
19
40


1 basketball:1.00
2 streetball:0.00
FREE THROWS
BULLS 4/7
HEAT 13/19
2011 EAST FINALS
CHI 40 MIA 42
2ND 1:41 16
Learn How To Dunk At
www.50inchvertical.com



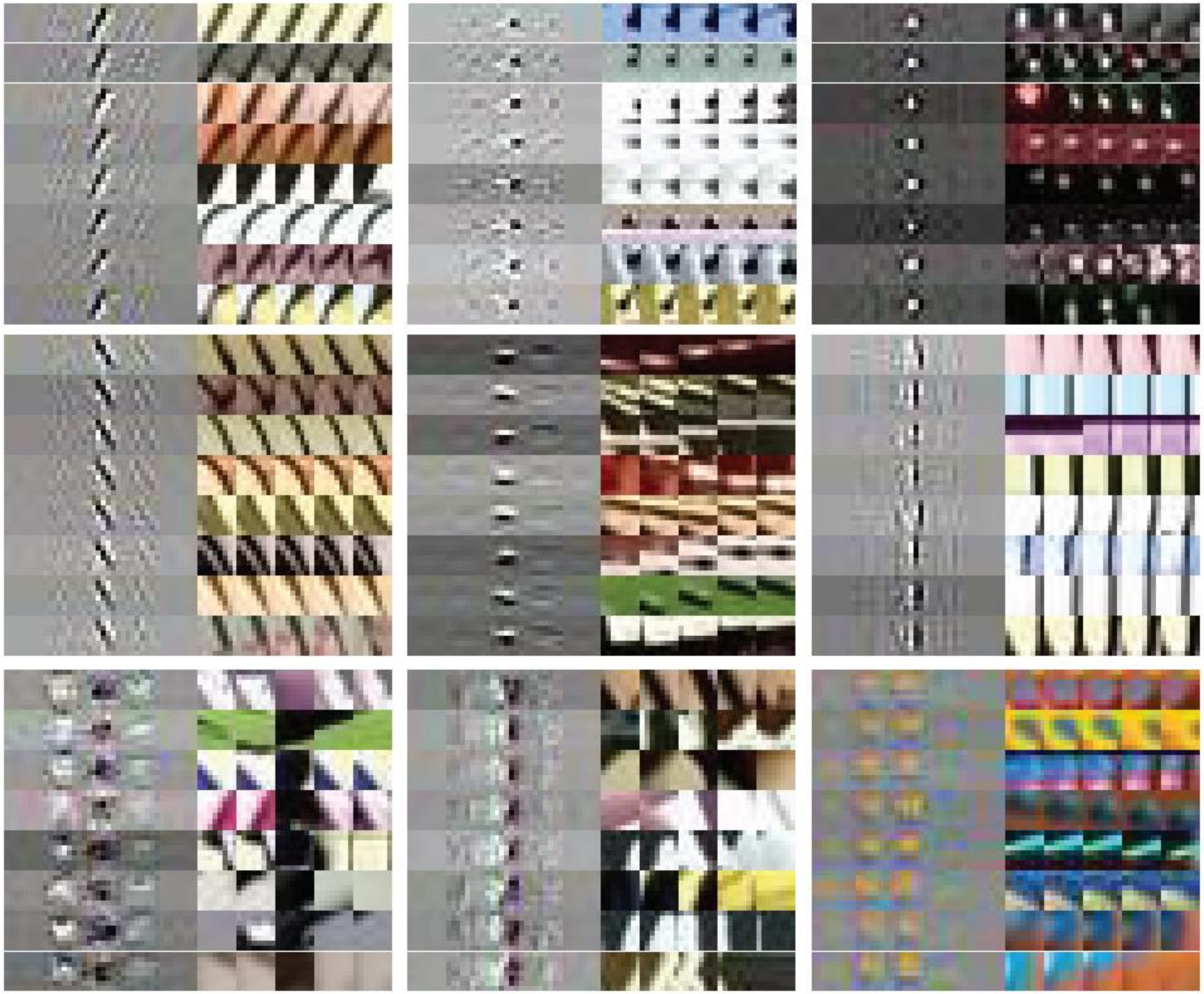
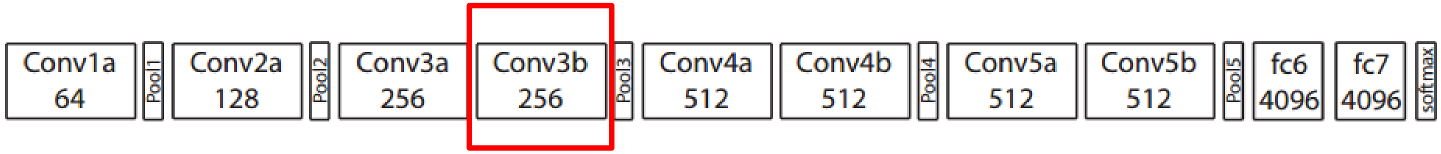
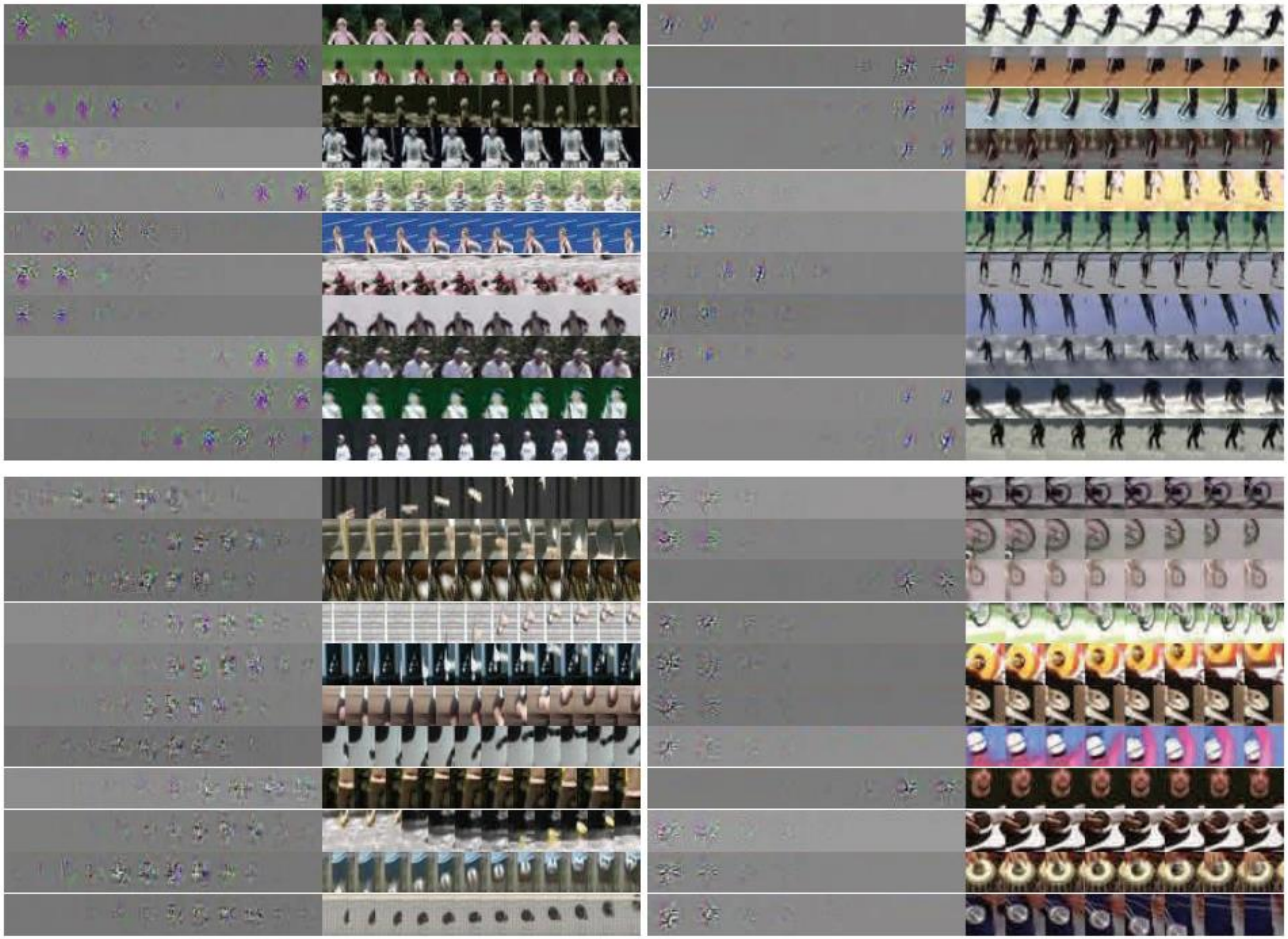



Figure 9. Deconvolutions of C3D `conv2a` feature maps. Each group is a C3D `conv2a` learned feature map. First two rows: the learned filters detect moving edges and blobs. The last row: the learned filters detect shot changes, edge orientation changes, and color changes. Best viewed in a color screen.


Conv1a
64
Pool1
Conv2a
128
Pool2
Conv3a
256
Conv3b
256
Pool3
Conv4a
512
Conv4b
512
Pool4
Conv5a
512
Conv5b
512
Pool5
fc6
4096
fc7
4096
softmax



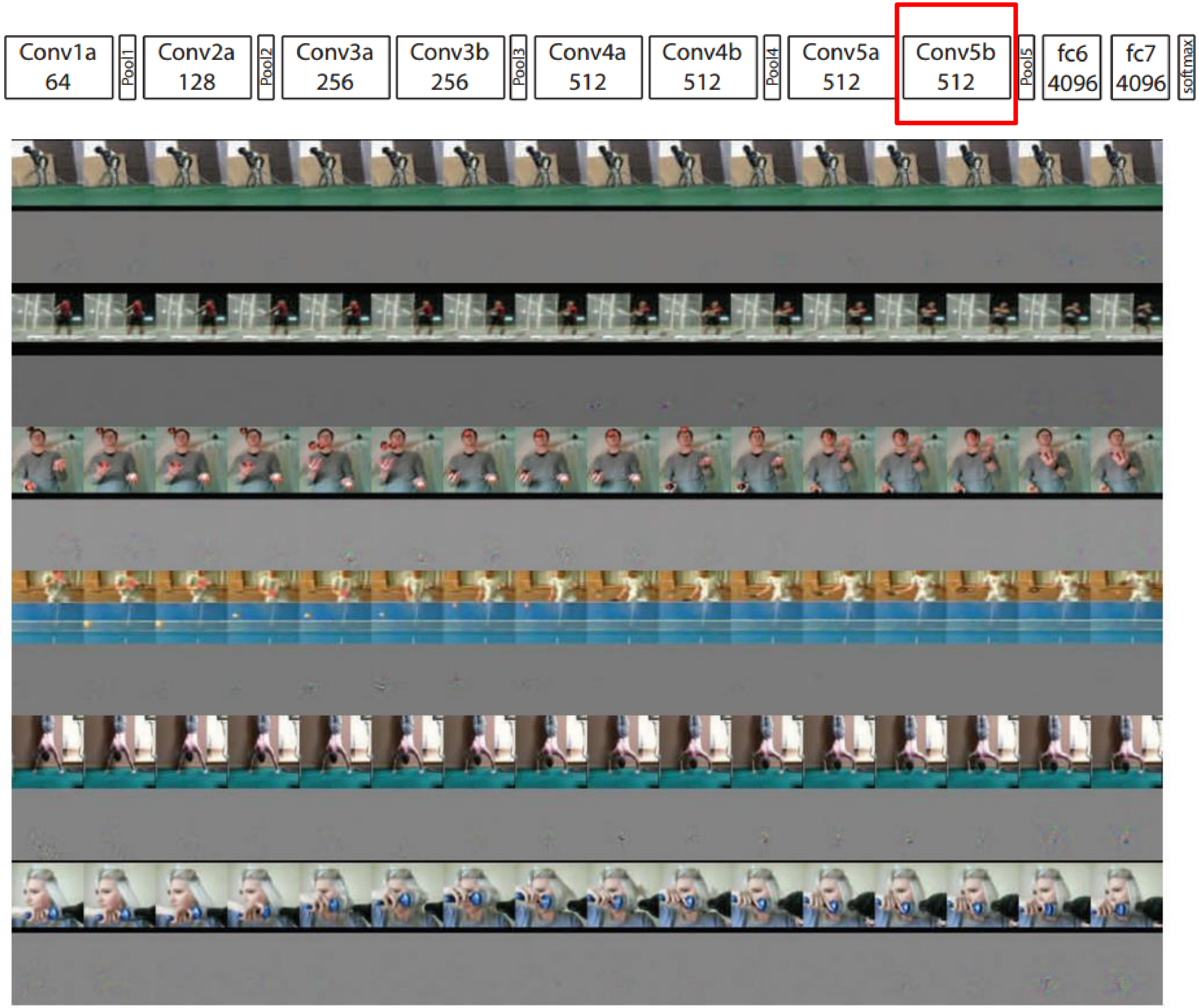
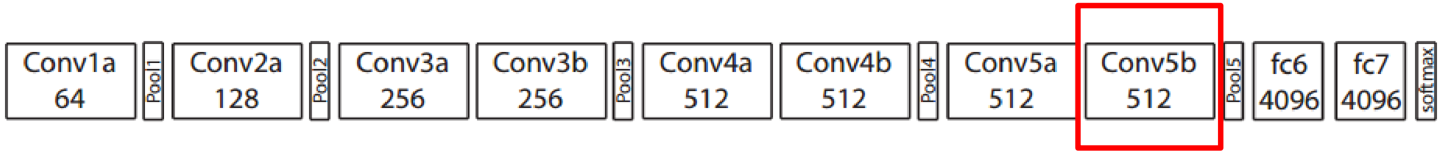



Figure 11. Deconvolutions of a C3D `conv5b` learned feature map which detects moving motions of circular objects. In the second last clip, it detects a moving head while in the last clip, it detects the moving hair-curler. Best viewed in a color screen.



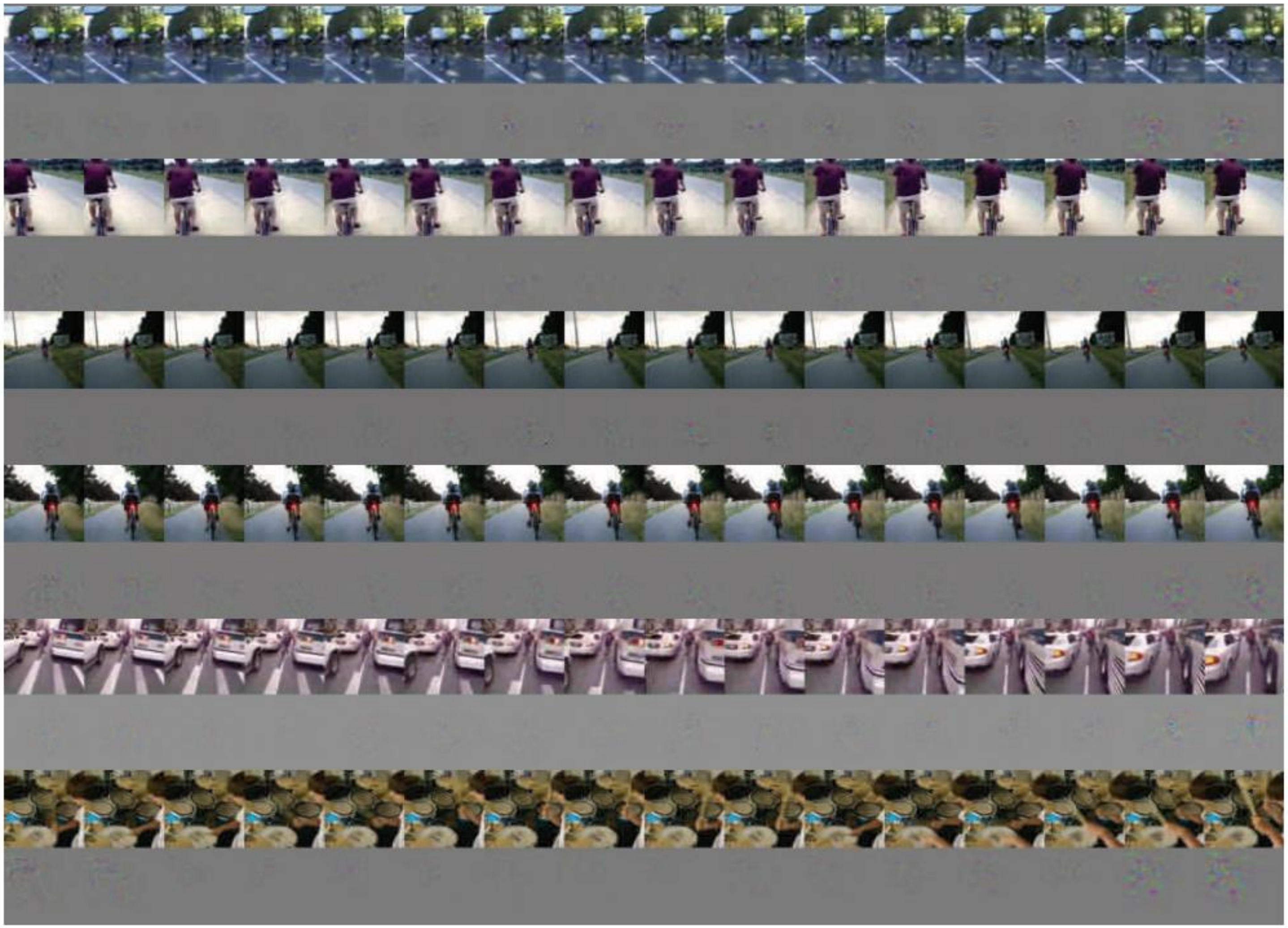



Figure 12. Deconvolutions of a C3D `conv5b` learned feature map which detects biking-like motions. Note that the last two clips have no biking but their motion patterns are similar to biking motions. Best viewed in a color screen.


Figure 14. Deconvolutions of a C3D `conv5b` learned feature map which detects balance-beam-like motions. In the last clip, it detects hammering which shares similar motion patterns with balance beam. Best viewed in a color screen.

# Human Action Recognition

1. Hand crafted feature + Shallow classifier

2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

3. 3D CNN
   - Input is a small chunk of video

4. Other combinations?
   - **Single frame/late fusion/slow fusion (3D CNN)**
   - Two stream (single frame + multi-frame optical flow)

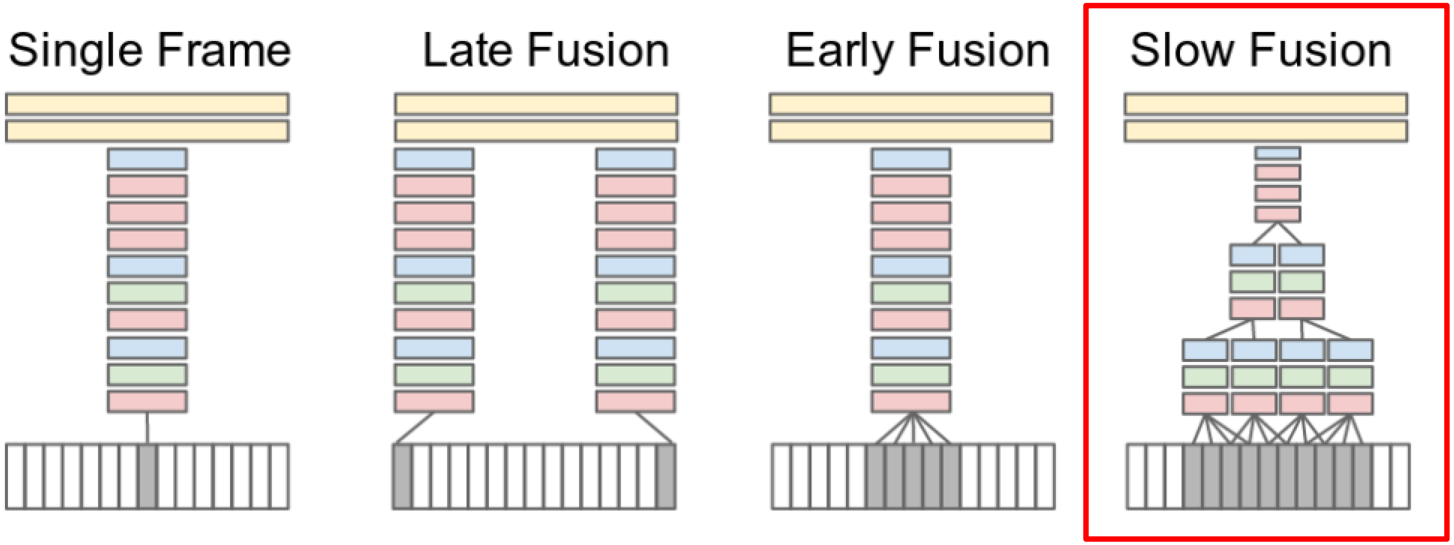
5. ConvNet + RNN
   - 3D CNN + RNN
   - CNN + RNN

# LARGE-SCALE VIDEO CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS

Karpathy et al., 2014

# Time-information fusion in CNNs

- Explored approaches
  - Explored approaches for fusing information over temporal dimension through the network. Red, green and blue boxes indicate convolutional, normalization and pooling layers respectively




3D CNN

# Multi-resolution CNNs

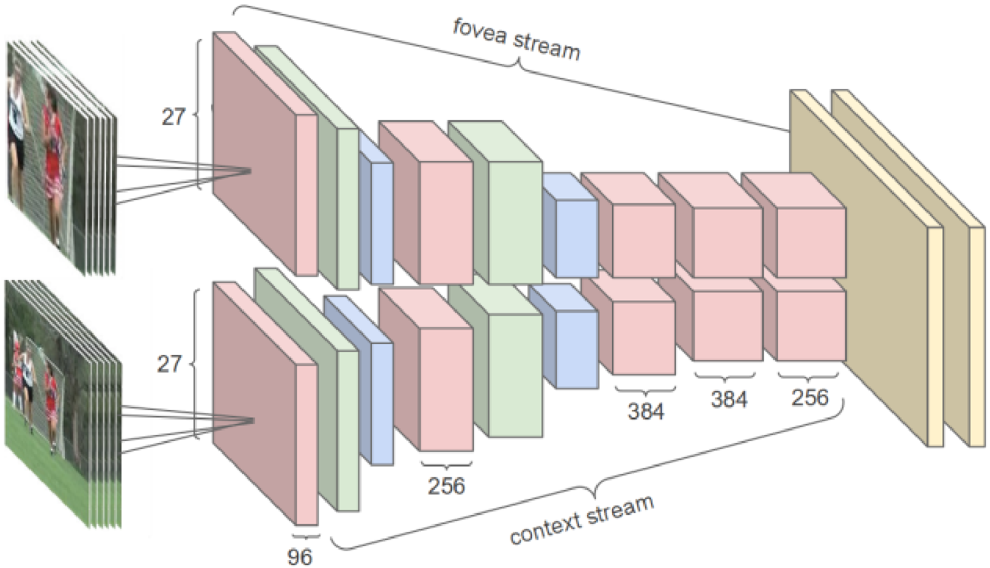
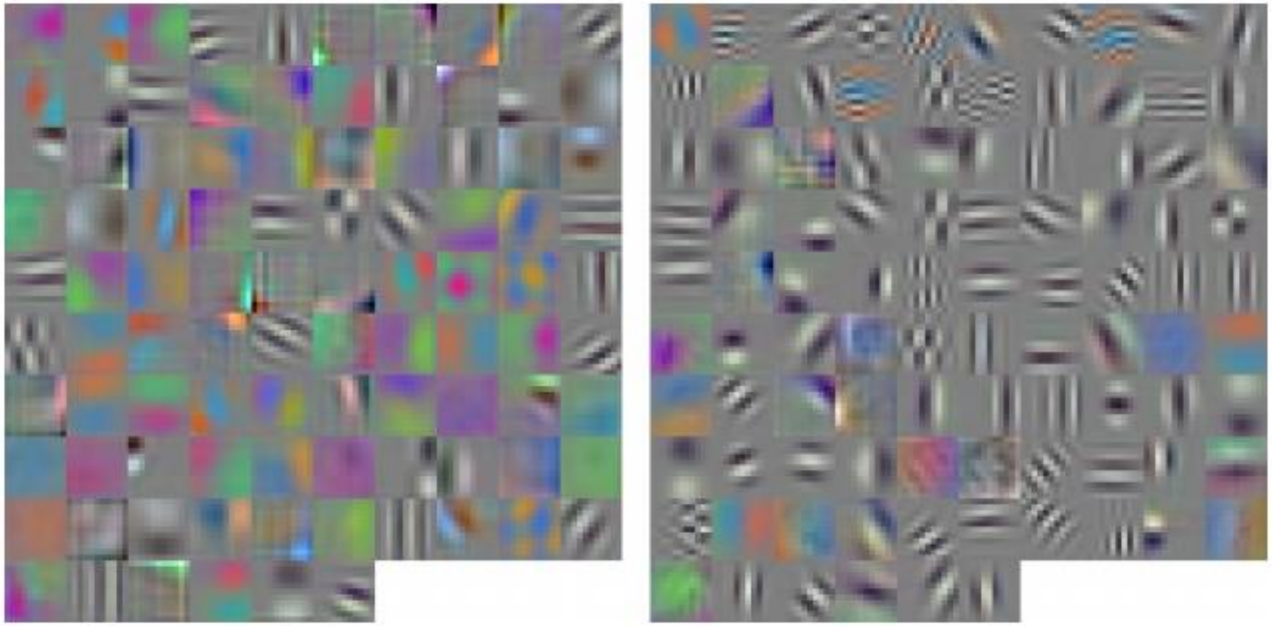


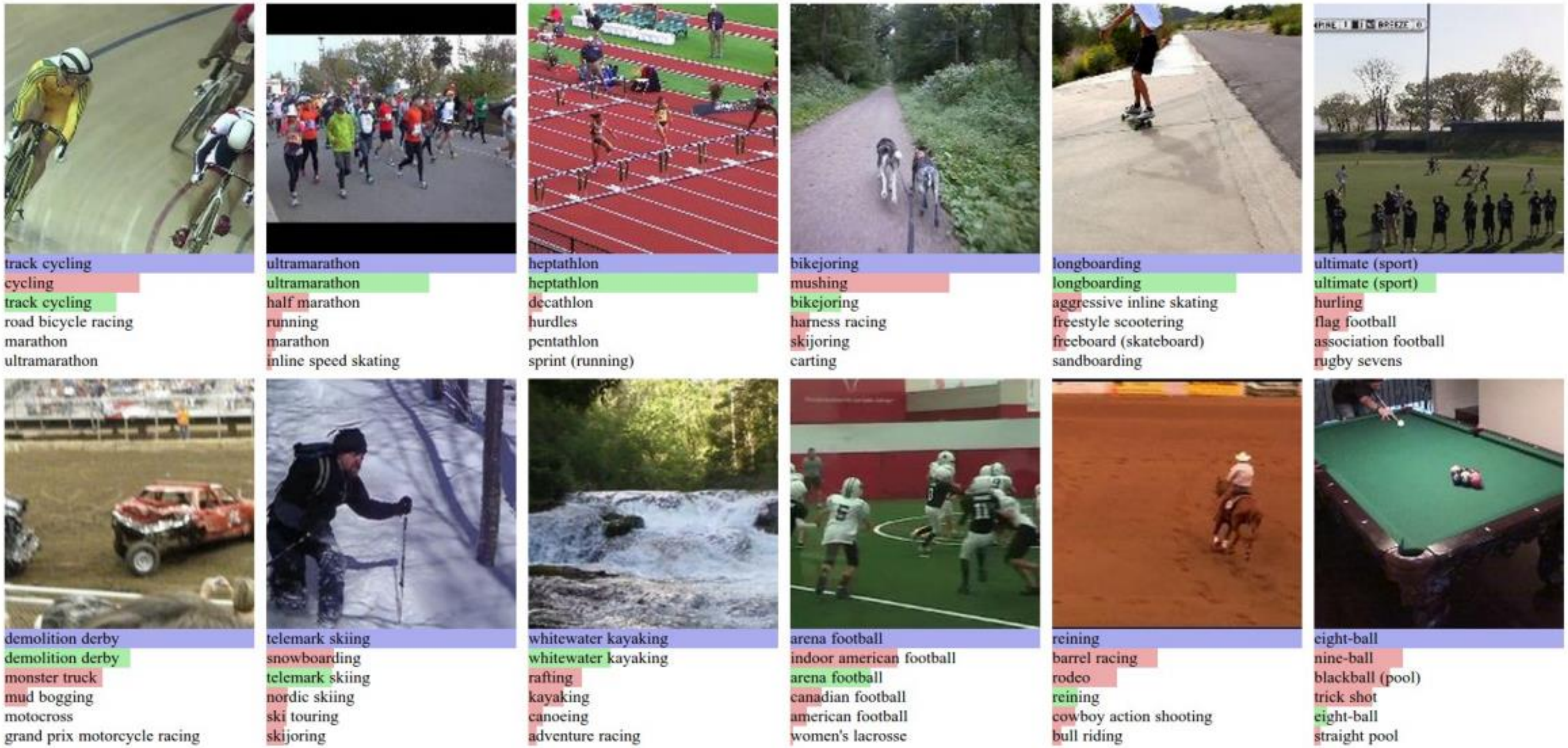

Figure 2: Multiresolution CNN architecture. Input frames are fed into two separate streams of processing: a *context stream* that models low-resolution image and a *fovea stream* that processes high-resolution center crop. Both streams consist of alternating convolution (red), normalization (green) and pooling (blue) layers. Both streams converge to two fully connected layers (yellow).

# Multi-resolution CNNs

- Left: context stream, Right: fovea stream
  - The fovea stream learns grayscale, high-frequency features while the context stream models lower frequencies and colors



# Predictions on Sports-1M test data



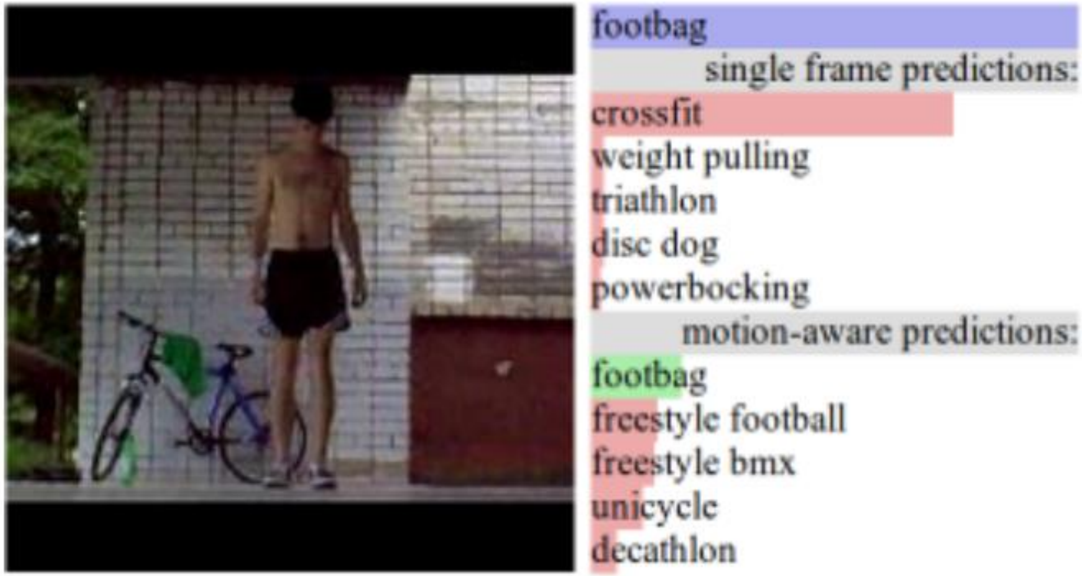
# Results

# Results

- Motion information didn't add all that much

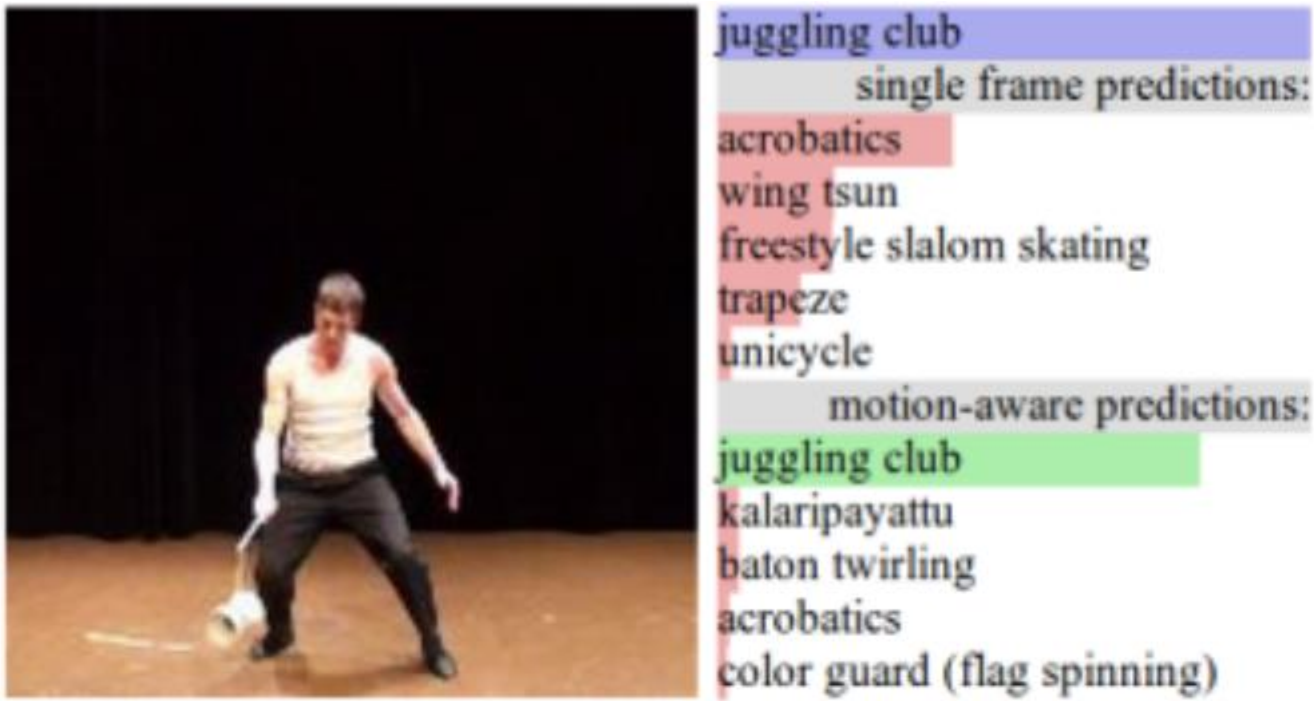
| Model | Clip Hit@1 | Video Hit@1 | Video Hit@5 |
|---|---|---|---|
| Feature Histograms + Neural Net | - | 55.3 | - |
| Single-Frame | 41.1 | $\boxed{59.3}$ | 77.7 |
| Single-Frame + Multires | **42.4** | **60.0** | **78.5** |
| Single-Frame Fovea Only | 30.0 | 49.9 | 72.8 |
| Single-Frame Context Only | 38.1 | 56.0 | 77.2 |
| Early Fusion | 38.9 | 57.7 | 76.8 |
| Late Fusion | 40.7 | 59.3 | 78.7 |
| Slow Fusion | **41.9** | $\boxed{\mathbf{60.9}}$ | **80.2** |
| CNN Average (Single+Early+Late+Slow) | 41.4 | 63.9 | 82.4 |

# Single-frame vs Motion-aware



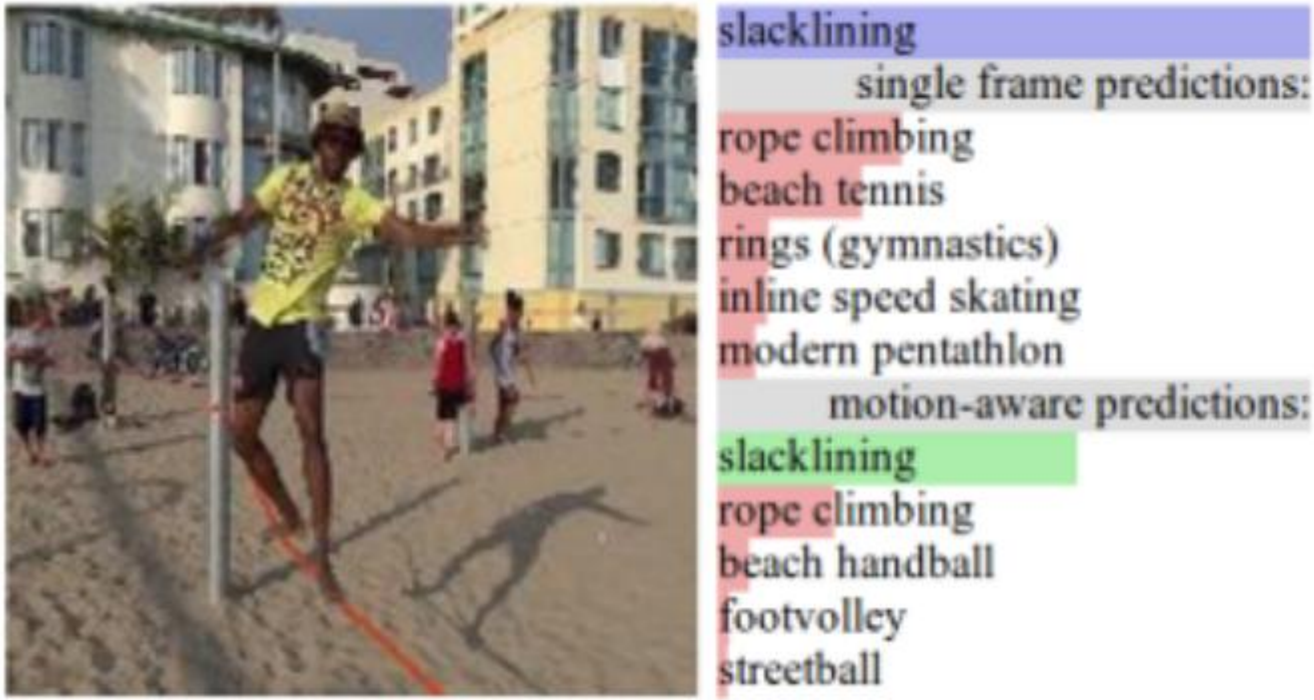

vs

# Single-frame vs Motion-aware



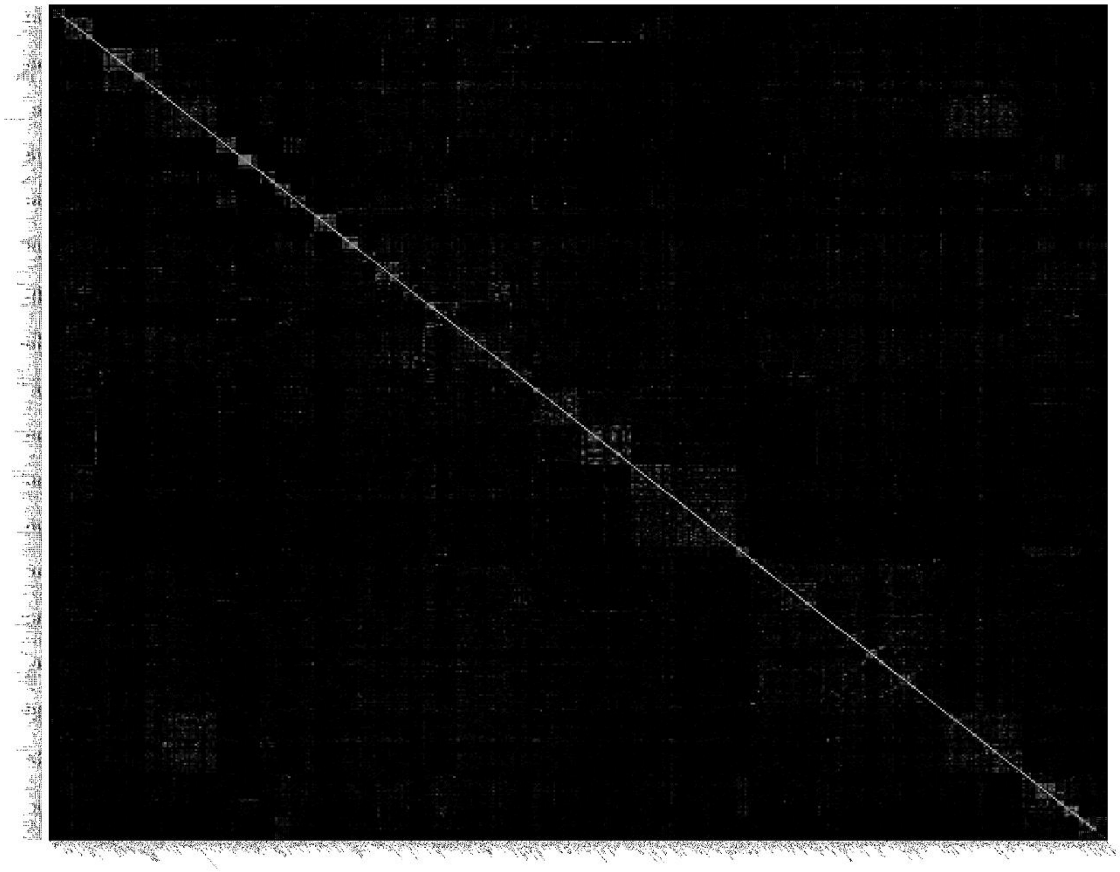

vs

# Single-frame vs Motion-aware

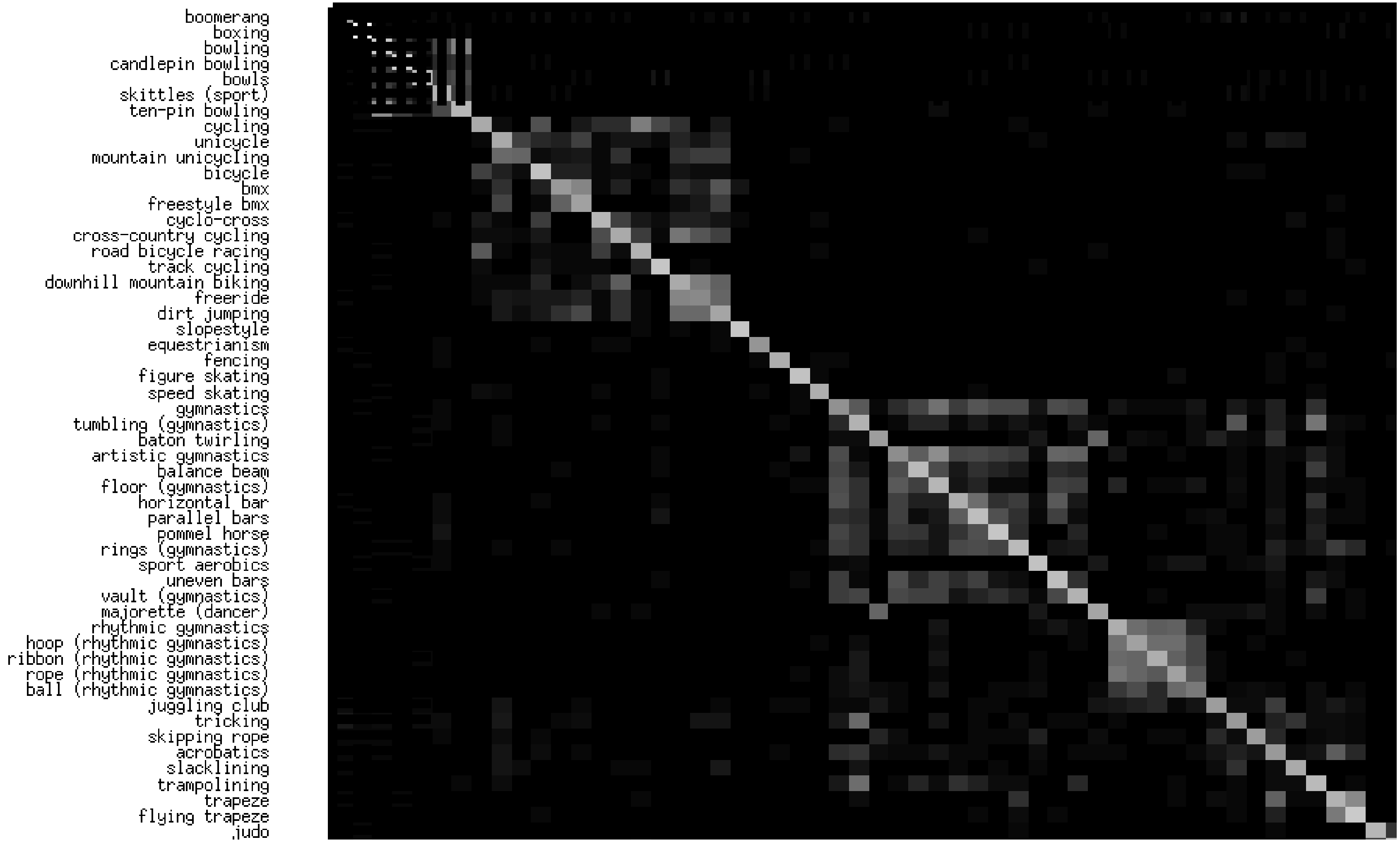


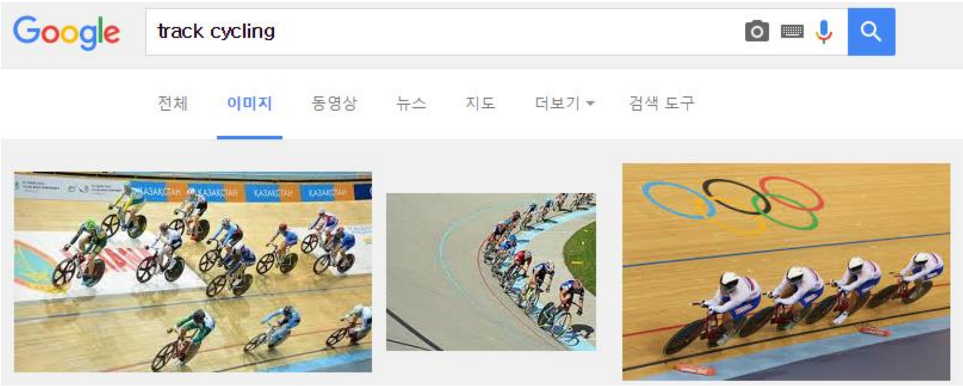

vs

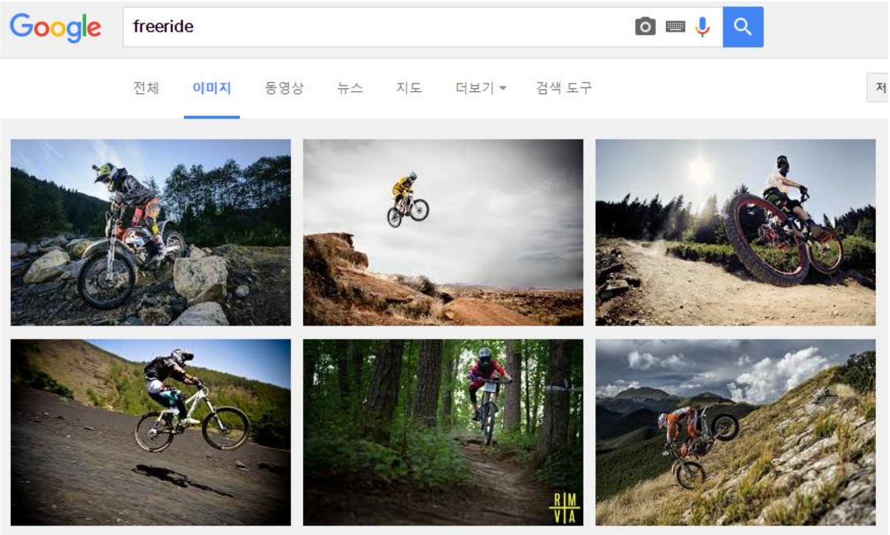
# Confusion matrix

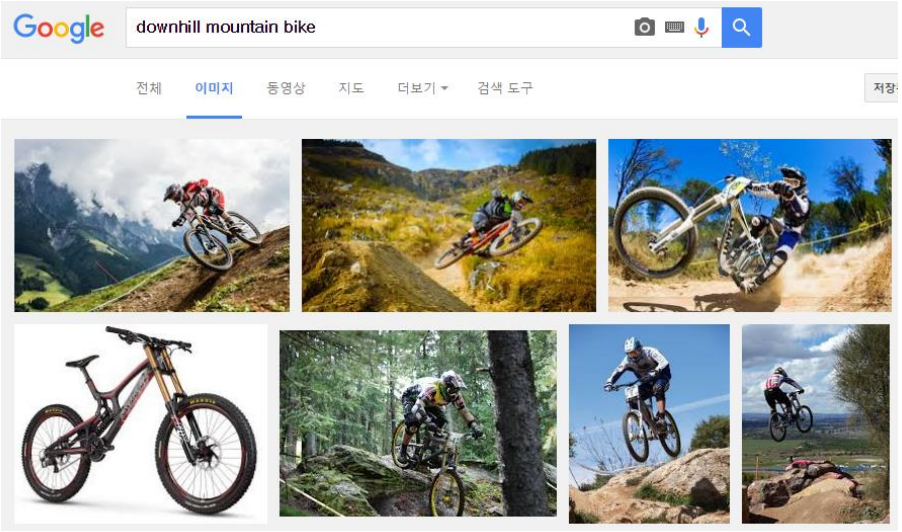


# Confusion matrix




track cycling
전체
이미지
동영상
뉴스
지도
더보기
검색 도구


freeride
전체
이미지
동영상
뉴스
지도
더보기
검색 도구


downhill mountain bike
전체
이미지
동영상
지도
더보기
검색 도구

# Human Action Recognition

1. Hand crafted feature + Shallow classifier

2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

3. 3D CNN
   - Input is a small chunk of video

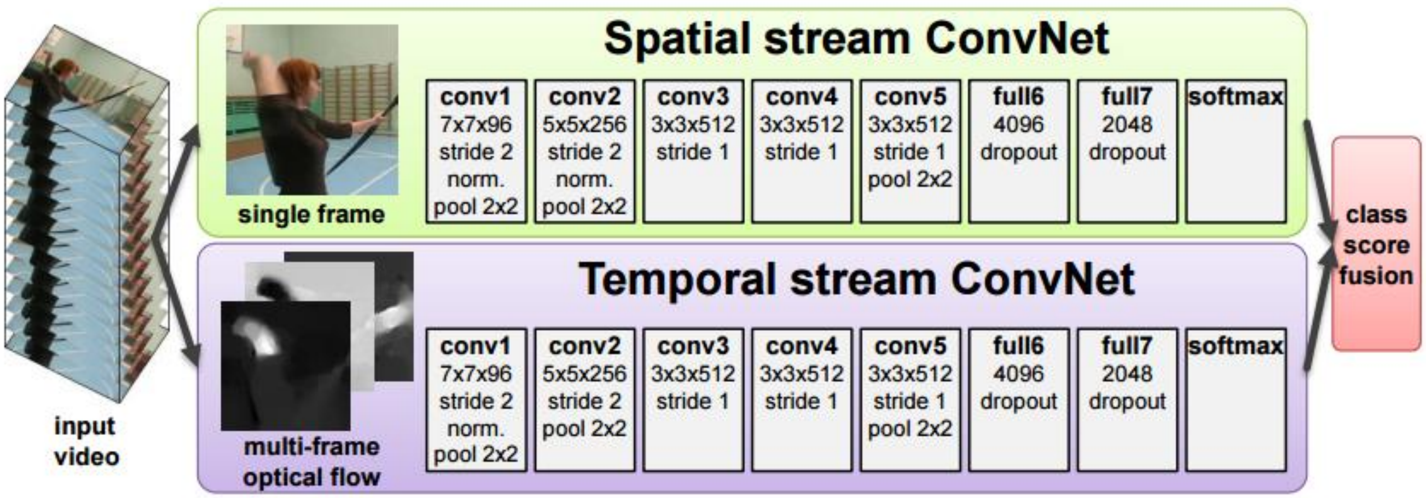
4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - **Two stream (single frame + multi-frame optical flow)**

5. ConvNet + RNN
   - 3D CNN + RNN
   - CNN + RNN

# TWO-STREAM CONVOLUTIONAL NETWORKS FOR ACTION RECOGNITION IN VIDEOS

Simonyan and Zisserman 2014

# Two-stream architecture



# Human Action Recognition

1. Hand crafted feature + Shallow classifier

2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

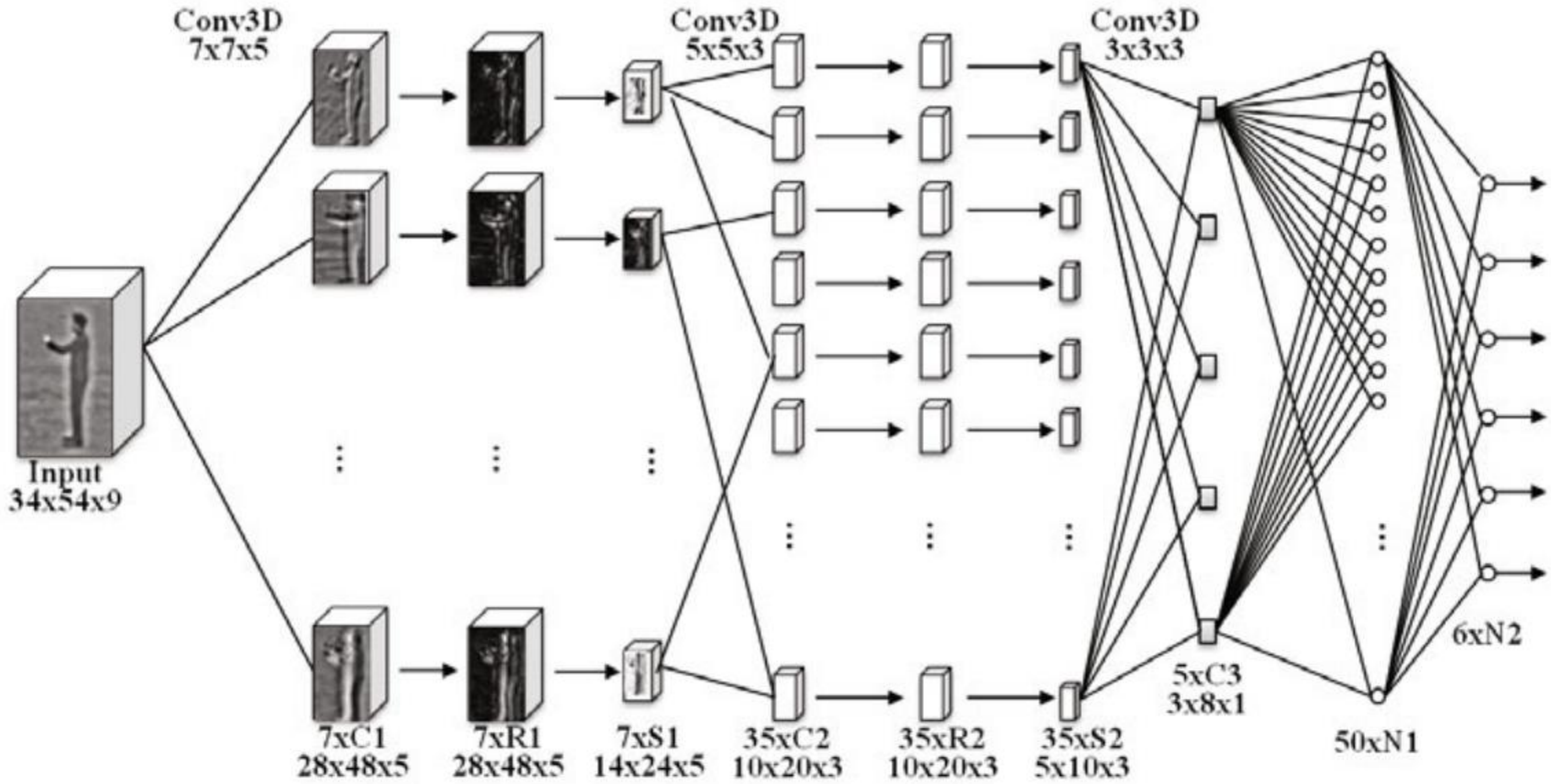
3. 3D CNN
   - Input is a small chunk of video

4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - Two stream

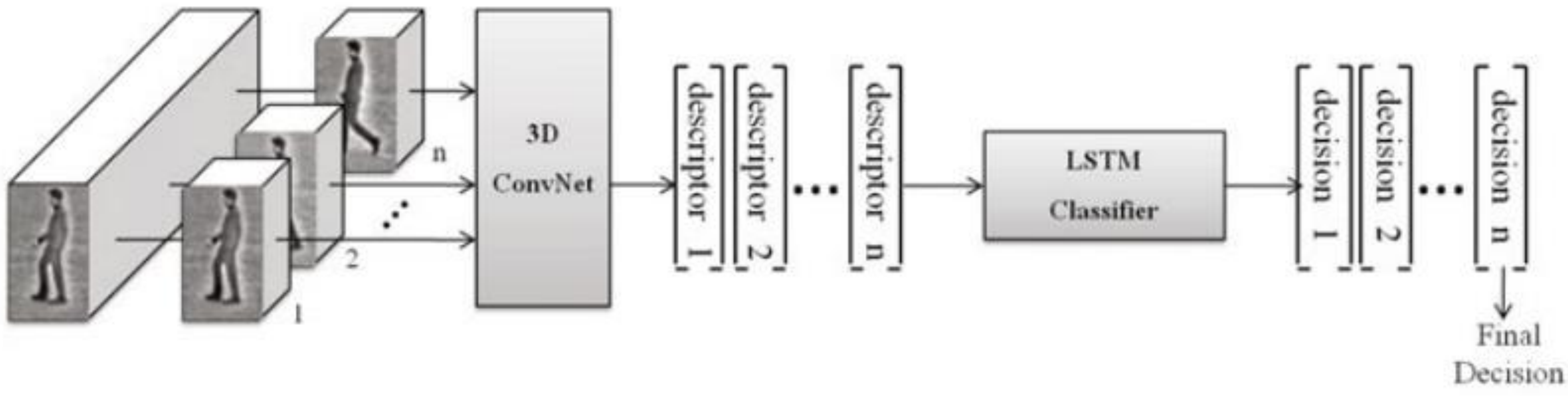
5. ConvNet + RNN
   - **3D CNN + RNN**
   - CNN + RNN

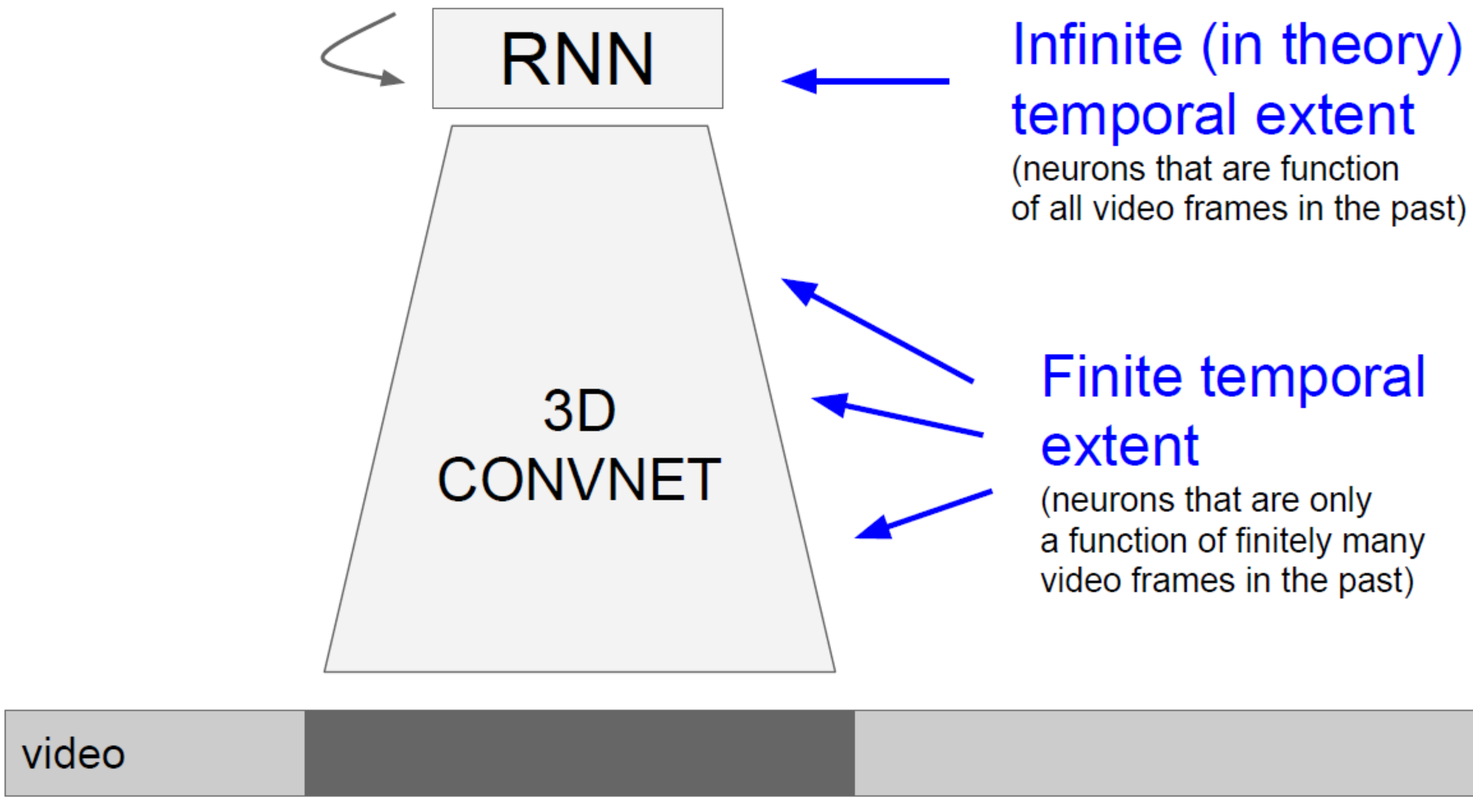
# SEQUENTIAL DEEP LEARNING FOR HUMAN ACTION RECOGNITION, BACCOUCHE ET AL., 2011

# 3D-ConvNet architecture



# Two-steps neural recognition scheme




RNN
3D
CONVNET
video
Infinite (in theory)
temporal extent
(neurons that are function
of all video frames in the past)
Finite temporal
extent
(neurons that are only
a function of finitely many
video frames in the past)

# Human Action Recognition

1. Hand crafted feature + Shallow classifier

2. Human localization + (Hand crafted features) + 3D CNN
   - Input is a small chunk of video

3. 3D CNN
   - Input is a small chunk of video

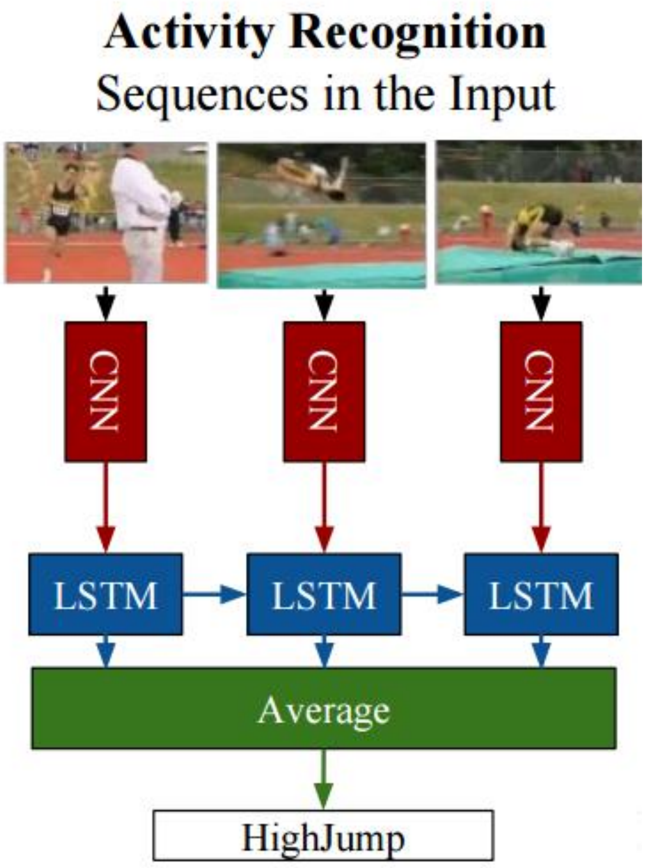
4. Other combinations?
   - Single frame/late fusion/slow fusion (3D CNN)
   - Two stream

5. ConvNet + RNN
   - 3D CNN + RNN
   - **CNN + RNN**

# LONG–TERM RECURRENT CONVOLUTIONAL NETWORKS FOR VISUAL RECOGNITION AND DESCRIPTION

Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, Trevor Darrell

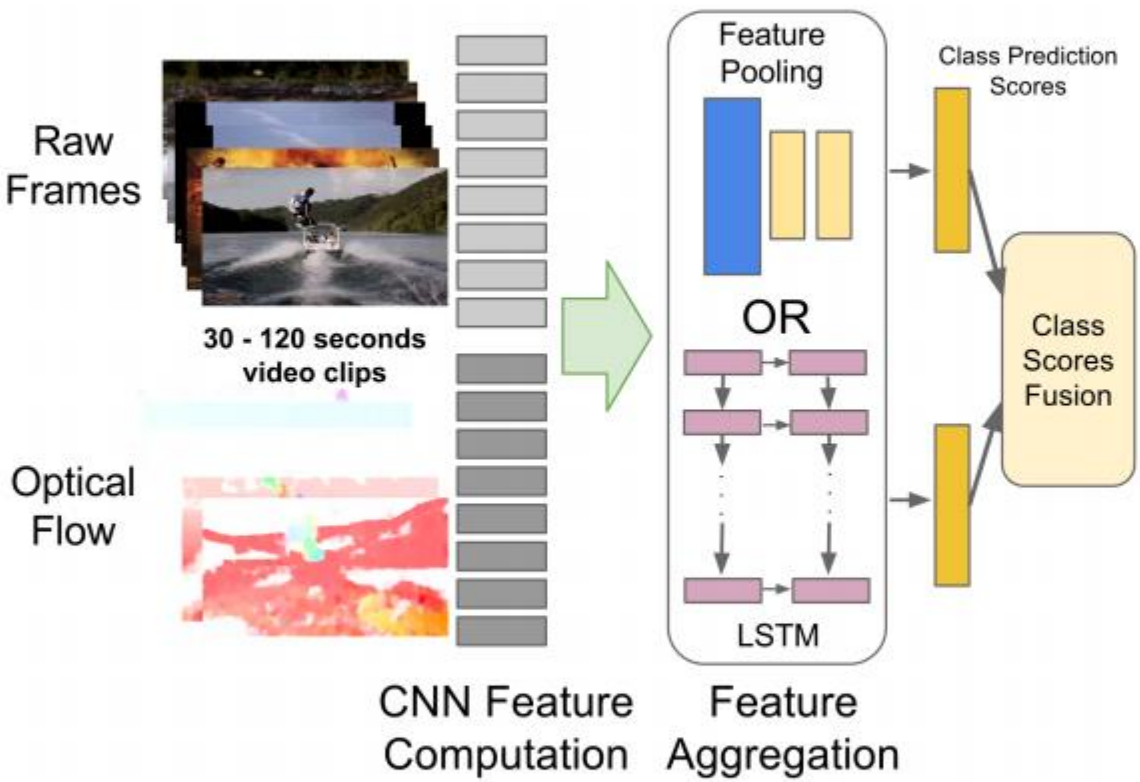
# Long-time Spatio-Temporal ConvNets



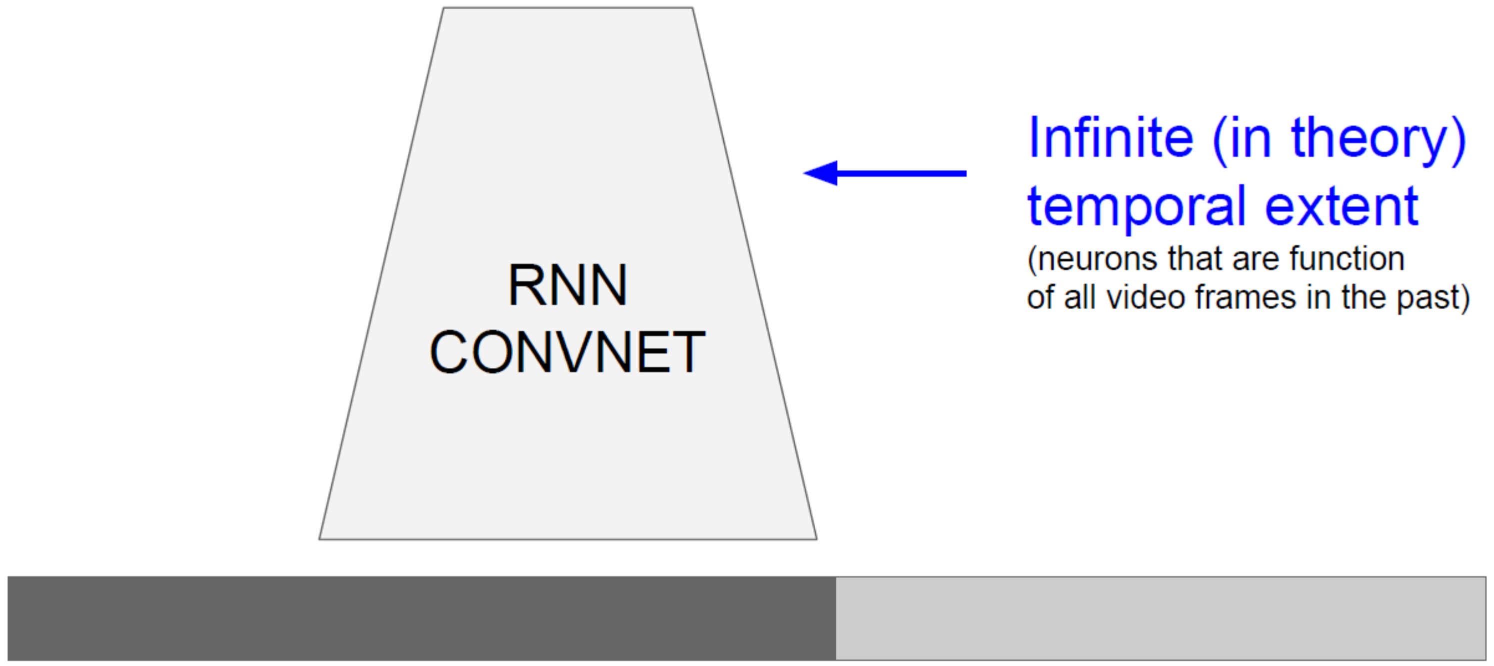
# BEYOND SHORT SNIPPETS: DEEP NETWORKS FOR VIDEO CLASSIFICATION

Joe Yue-Hei Ng et. al

# Long-time Spatio-Temporal ConvNets



# RNN-ConvNet

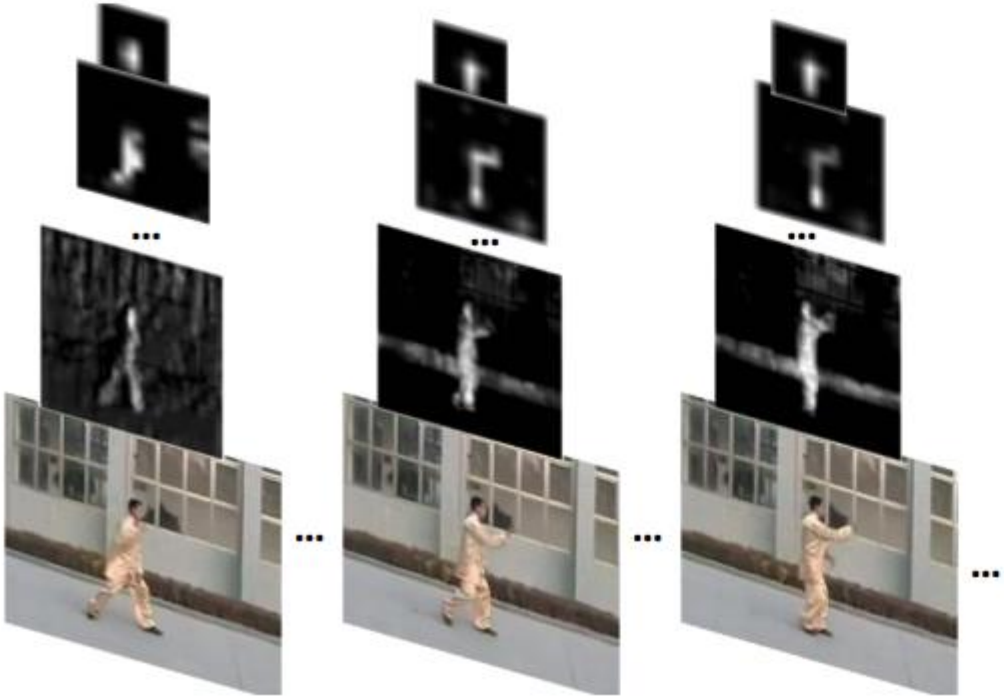

RNN
CONVNET
Infinite (in theory)
temporal extent
(neurons that are function
of all video frames in the past)

# DELVING DEEPER INTO CONVOLUTIONAL NETWORKS FOR LEARNING VIDEO REPRESENTATIONS

Ballas et al., 2016

# Limitations in simple RNN-ConvNet

- Visualization of convolutional maps on successive frames in video. As we go up in the CNN hierarchy, we observe that the convolutional maps are more stable over time, and thus discard variation



# Stack-GRU-RCN

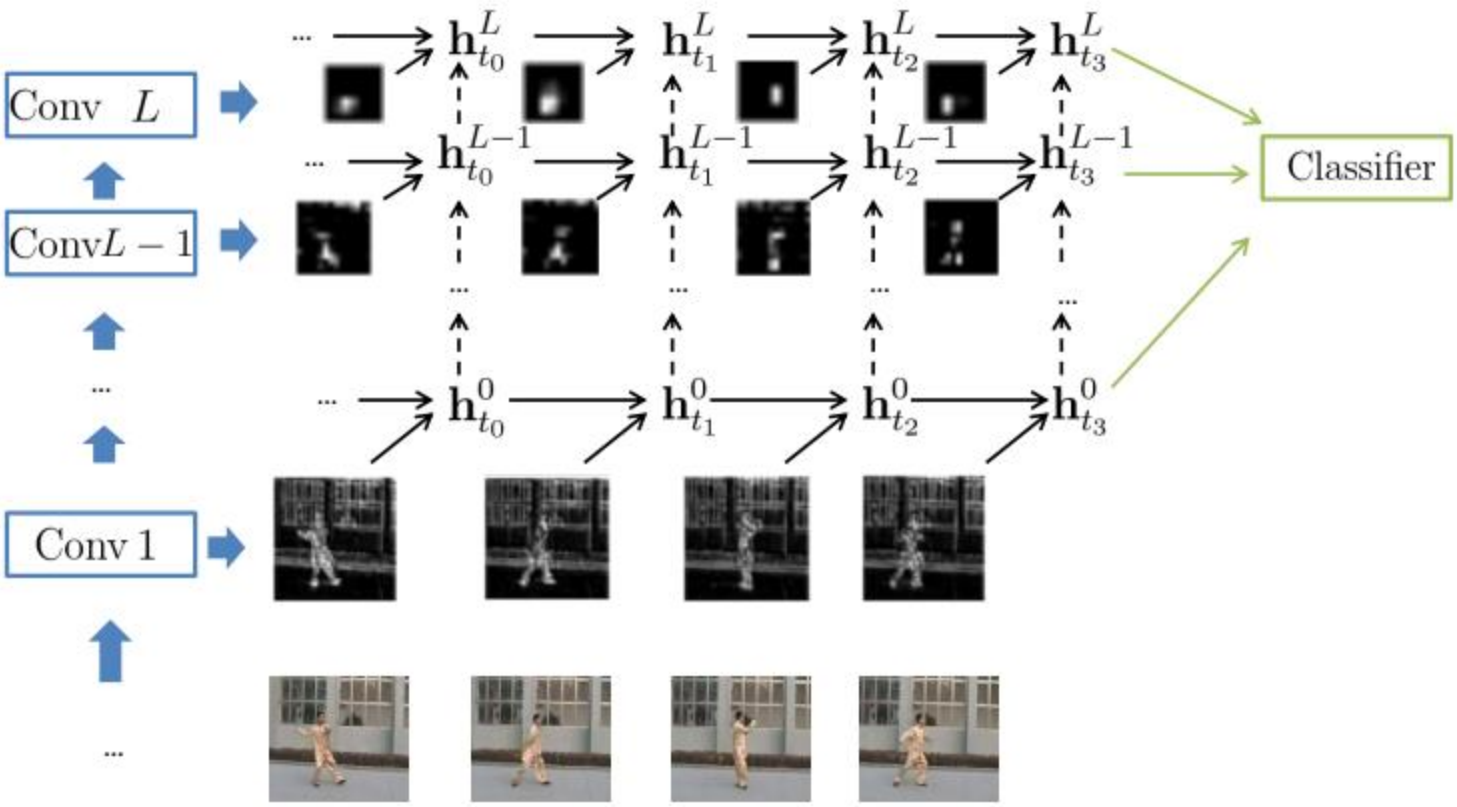



Figure 2: High-level visualization of our model. Our approach leverages convolutional maps from different layers of a pretrained-convnet. Each map is given as input to a convolutional GRU-RNN (hence GRU-RCN) at different time-step. Bottom-up connections may be optionally added between RCN layers to form Stack-GRU-RCN.

RCN (Recurrent Convolution Networks)

# Summary

- You think you need a Spatio-Temporal Fancy Video ConvNet
- STOP. Do you really?
- Okay fine: do you want to model:
  - local motion? (use 3D CONV), or
  - global motion? (use LSTM).
- Try out using Optical Flow in a second stream (can work better sometimes)
- Try out GRU-RCN

# BACKUPS

# 3D CONVOLUTIONAL NEURAL NETWORKS FOR HUMAN ACTION RECOGNITION, JI ET AL., 2010

# 3D convolution

| | Gray | | Gradient(x2) | | Optical flow(x2) | | | Image size |
|---|---|---|---|---|---|---|---|---|
| | time | channel | time | channel | time | channel | | |
| H1 | 7 | 1 | 7 | 1 | 6 | 1 | 33 | 60x40 |
| C2 | 5 | 2 | 5 | 2 | 4 | 2 | 23x2 | 54x33 |
| S3 | 5 | 2 | 5 | 2 | 4 | 2 | 23x2 | 27x17 |
| C4 | 3 | 6 | 3 | 6 | 2 | 6 | 13x6 | 21x12 |
| S5 | 3 | 6 | 3 | 6 | 2 | 6 | 13x6 | 7x4 |
| C6 | | | | | | | 128 | 1x1 |
| ouput | | | | | | | | |