# Real-time Action Recognition with Enhanced Motion Vector CNNs

Bowen Zhang[1,2] Limin Wang[1,3] Zhe Wang[1] Yu Qiao[1*] Hanli Wang[2]

[1]Shenzhen key lab of Comp. Vis. & Pat. Rec., Shenzhen Institutes of Advanced Technology, CAS, China

[2]Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai, China

[3]Computer Vision Lab, ETH Zurich, Switzerland

IEEE 2016 Conference on Computer Vision and Pattern Recognition

CVPR 2016

## Introduction

### Goal:

- Improve the speed of two stream ConvNets for video based action recognition.

### Existing works:

- Two-stream ConvNets [1]: Using stacked optical flows and RGB images as inputs to CNN. However, the calculation of optical flows is computationally expensive.
- Efficient feature extraction, encoding and classification for action recognition [3]: Extracting features around motion vector trajectories and using tree-based ANN to accelerate VLAD/Fisher Vector computation.

### Our observations:

- Calculating optical flow is time consuming while motion vector can be obtained in video decoding process without extra calculation. Please see Table 3 for the speed comparison.
- Optical flow and motion vector share some similar characteristics which allows us to transfer the fine knowledge learned in optical flow CNN (OF-CNN) to motion vector CNN (MV-CNN).

### Our idea: *Enhanced Motion Vector CNNs:*

- A real-time CNN based action recognition method with high performance is proposed.
- We firstly introduce motion vector as the input of CNN to avoid the heavy computational cost of optical flow.
- We propose techniques to transfer the knowledge of optical flow CNN to motion vector CNN, which significantly improves the recognition performance.

## Reference

1. K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS'14*, 2014.
2. T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV'14*, 2004.
3. V. Kantorov and I. Laptev. Efficient feature extraction, encoding, and classification for action recognition. In *CVPR'14*, 2014.

## Framework of real-time action recognition with EMV-CNN

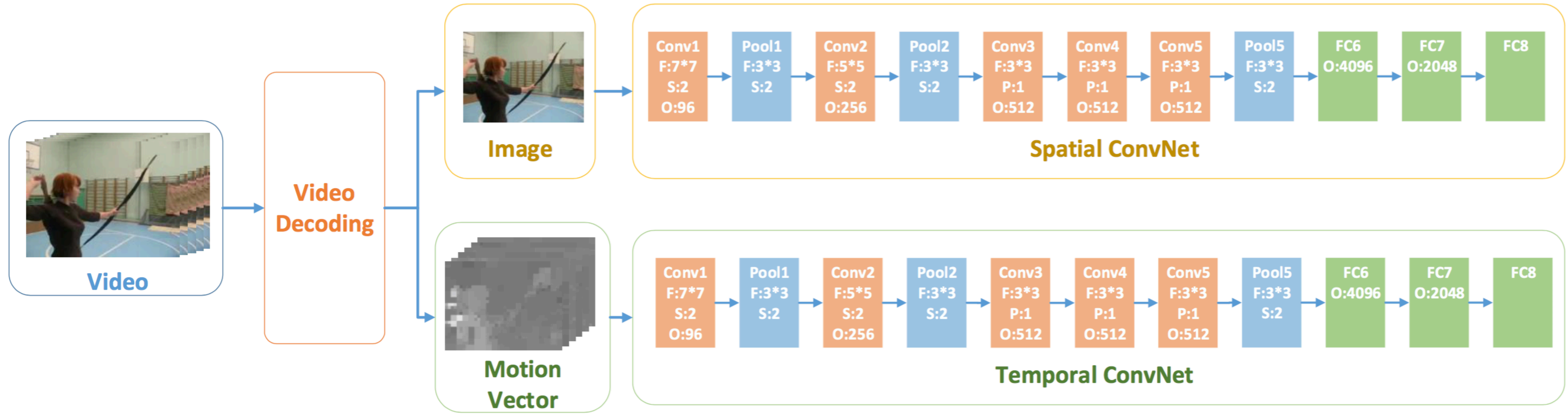


Figure 1: Framework of real-time action recognition with EMV-CNN

## Motion Vector

### Motion Vector:

- Motion vectors are designed for describing macro blocks movement from one frame to the next, and are widely used in video compression standards.
- Motion vectors only contain block-level motion information, which exhibit much coarser structure than optical flows.
- As precision motion information is not obligatory for motion vectors, motion vectors contain noisy information.

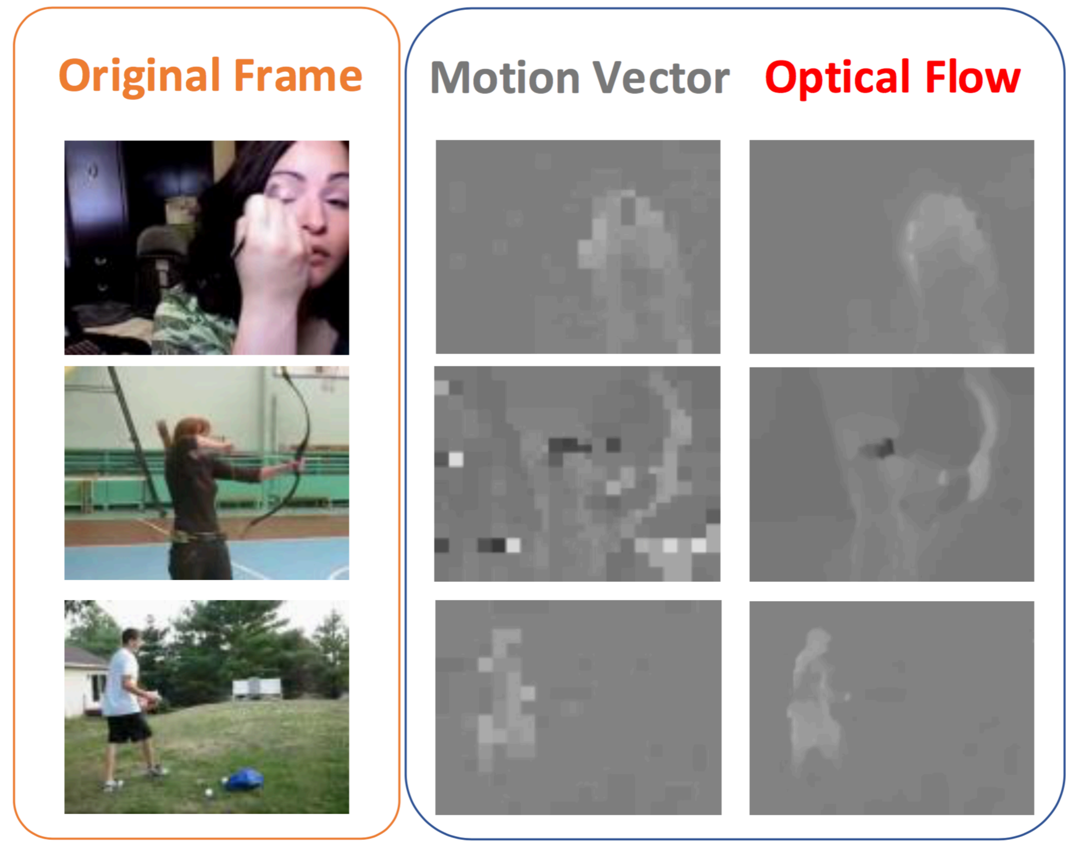


Figure 2: Examples of Motion Vector and Optical Flow

## Enhanced Motion Vector CNNs

### Motivation:

- Using motion vectors can improve the processing speed, but achieve inferior performance.
- Due to the coarse structure and inaccurate motion information of motion vectors, it is hard to obtain high performance with MV-CNN.
- Both optical flows and motion vectors contain motion information. The difference is that optical flows have fine grained structure, while motion vectors contain coarse ones.

### Enhanced Motion Vector CNNs:

- In order to enhance MV-CNN, we propose three methods to transfer knowledge from OF-CNN to MV-CNN: Teacher Initialization, Supervision Transfer and their combination.

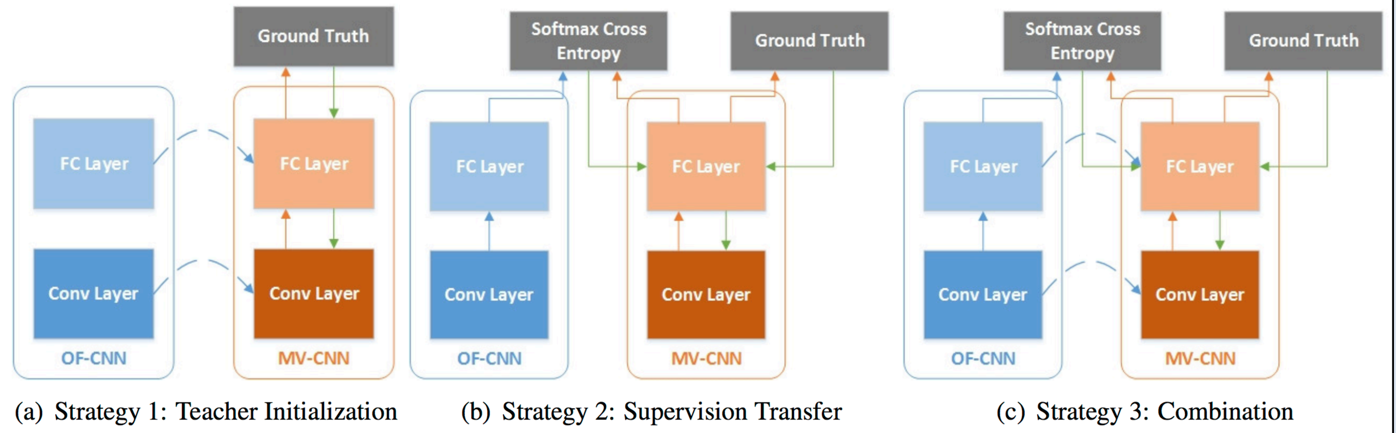


Figure 3: Knowledge transfer strategy from OF-CNN to MV-CNN

## Experiment Results

### EMV-CNN vs MV-CNN

| Temporal CNN | Accuracy |
|---|---|
| OF-CNN [1] | 81.2% |
| MV-CNN trained from scratch | 74.4% |
| EMV-CNN with ST | 77.5% |
| EMV-CNN with TI | 78.2% |
| EMV-CNN with ST+TI | **79.3%** |

Table 1: Performance of different knowledge transfer strategies on UCF-101 Split 1

| CNN | MAP |
|---|---|
| RGB CNN | 57.7% |
| OF-CNN | 55.3% |
| RGB CNN+OF-CNN | 66.1% |
| MV-CNN | 29.8% |
| EMV-CNN | 41.6% |
| RGB CNN+MV-CNN | 58.7% |
| RGB CNN+EMV-CNN | **61.5%** |

Table 2: Performance on THUMOS-14

### Speed comparison

| Dataset | Spatial Resolution | Brox's Flow [2] (GPU) (fps) | MV (CPU) (fps) |
|---|---|---|---|
| UCF101 | $320 \times 240$ | 16.7 | 735.3 |
| THUMOS14 | $320 \times 180$ | 17.5 | 781.3 |

Table 3: Speed of Brox's Flow and MV on UCF101 and THUMOS14

### Comparison with state-of-the-art result

| | MAP | FPS |
|---|---|---|
| Objects (GPU) | 44.7% | - |
| iDT+CNN (CPU+GPU) | 62.0% | $< 2.38$ |
| Motion (iDT+FV) (CPU) | 63.1% | 2.38 |
| Objects+Motion (CPU+GPU) | 71.6% | $< 2.38$ |
| EMV+RGB-CNN | **61.5%** | **403.2** |

Table 4: Performance on THUMOS-14

| | Accuracy | FPS |
|---|---|---|
| MV+FV (CPU) (re-implement) [3] | 78.5% | 132.8 |
| C3D (1 net) (GPU) | 82.3% | 313.9 |
| C3D (3 net) (GPU) | 85.2% | - |
| iDT+FV (CPU) | 85.9% | 2.1 |
| Two-stream CNNs (GPU) | 88.0% | 14.3 |
| EMV+RGB-CNN | **86.4%** | **390.7** |

Table 5: Performance on UCF101 (3 splits)