Computer Vision Notes
Action Recognition (Early Approaches)

Faisal Z. Qureshi

Faculty of Science
Copyright information and license

© Faisal Z. Qureshi

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.
Acknowledgements

▶ Sullivan and Carlsson, “Recognizing and Tracking Human Actions”
Action Recognition Applications

- Surveillance footage
- User-interfaces
- Automatic video organization
Challenges

- Occlusions
- Scale
- Camera movement
- Presence of multiple actions
- Clutter (background)
- Variations in how actions are performed
Spatio Temporal Interest Points

- Construct scale-space representation $L(., \sigma^2, \tau^2)$ using Gaussian convolutional kernel.
- Compute second-moment matrix $\nabla L$ within Gaussian neighborhood of each point.
- Define feature positions using local maxima of $H$.

**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.
Spatio-Temporal Feature Descriptors

- Spatio-temporal neighborhoods of local features contain information about the motion and the spatial appearance of events in image sequences.
- Compute spatio-temporal jets (descriptors) to capture this information \( l = (L_x, L_y, L_t, L_{xx}, \cdots, L_{tttt}) \).
- Cluster descriptors \( l \) using K-means. This gives us a vocabulary of primitive events \( h_i \).
- Compute histogram \( H = (h_1, \cdots, h_n) \), where bin \( h_i \) count the number of features with label \( h_i \).
Figure 2. Action database (available on request): examples of sequences corresponding to different types of actions and scenarios.
Evalution

Representations
1. Local features described by spatio-temporal jets of order 4 (LF)
2. 128-bin histograms of local features (HistLF)
3. Marginalized histograms of normalized spatial temporal gradients (HistSTG)

Classifiers
1. Support Vector Machine (SVM)
2. Nearest Neighbor Classifier (NNC)

Observations
- LF with SVM gives the best performance.
- SVM gives better performance than NNC on HistLF and HistSTG, with HistLF performing slightly better than HistSTG.
- Supervised learning approach
Matching Local Features

The pairs correspond to features with jet descriptors $l_{jh}$ and $l_{jk}$ selected by maximizing the feature kernel over $j_k$ in

$$K(L_h, L_k) = \frac{1}{n_h} \sum_{j_h=1}^{n_h} \max_{j=1, \ldots, n_k} K_l(l_{jh}, l_{jk})$$
Dataset

- Backgrounds are mostly free of clutter
- Single actor
- 25 people, each
  - 6 actions (walking, jogging, running, boxing, hand waving, clapping)
  - 4 scenarios (outdoors, outdoors + scale, outdoors + different clothes, indoors)
Approach

- Learn different classes of actions present in a collection of unlabeled videos.
- Classify actions in previously unseen videos by applying the learned models.

Assumptions

- Videos may contain a small amount of camera motion.
- Videos may contain some amount of background clutter.
- **Training**: videos contain a single actor.
- **Testing**: videos may contain more than one actor.
Approach

- Extract local space-time regions using space-time interest point detector.
- Cluster these regions in a codebook.
- Learn probability distributions and discover latent topics using.
- Use the learned model to recognize and localize human action classes.
Space-time interest point detectors

- Separable linear filters (2D Gaussian + 1D Gabor).
- Extract a small video cube around each interest point.
Video Representation as a Bag of Visual Words

Space-time descriptors
- Histogram of brightness gradient at each feature point.
  - Gradients concatenated to form feature vector.
  - Use PCA for reducing dimensionality.
- K-means clustering of video word descriptors to construct the codebook.

Video representation
- Histogram of video words from the codebook.
Model (learning probabilities and latent topics)

Given an input video $d_j$ and video words $w_i$, we can write the joint probability as follows:

$$p(d_j, w_i) = p(w_i|d_j)p(d_j).$$

Furthermore, given a set of (latent) actions $z_k$, where $k = 1, \ldots, K$,

$$p(w_i|d_j) = \sum_{k=1}^{K} p(w_i|z_k)p(z_k|d_j).$$

Here $K$ is the number of action categories, $p(z_k|d_j)$ are action category weights and $p(w_i|z_k)$ are action category vectors.

*Use probabilistic Latent Semantic Analysis (pLSA) or Latent Dirichlet Allocation (LDA) to learn the above model.*
Given a new video and the learnt model, we can classify it as belonging to one of the action categories using

\[ \arg \max_k P(z_k | d_{test}). \]

Recall

\[ p(w | d_{test}) = \sum_{k=1}^{K} P(z_k | d_{test})p(w | z_k). \]
CalTech Dataset

Words colored according to their most likely action category.
KTH Datasets

Words colored according to their most likely action category.
**Performance on KTH Dataset**

![Confusion Matrix](image)

Fig. 8 (a) Confusion matrix for the KTH dataset using 1500 codewords (performance average = 83.33%); *rows* are ground truth, and *columns* are model results; (b) Classification accuracy vs. codebook size for the KTH dataset. Experiments show that the results for the recognition task are consistently better when the pLSA model is adopted.
Weizmann Human Action Dataset
Fig. 13 (a) Confusion matrix for the Weizmann human action dataset (Blank et al. 2005); rows are ground truth, and columns are model results. The action models learnt with pLSA and using 1200 codewords show an average performance of 90%. (b) Classification accuracy obtained using pLSA and LDA models vs. codebook size. Our results show that pLSA performs slightly better than LDA in the video categorization task.
Dealing with multiple actions

1. Select topics with high $p(z_k|d_{test})$
2. Assign words to topics using $p(w|z_k)$
3. Cluster words from selected topics according to their spatial position
Dealing with multiple actions - Spatial Localization
Dealing with multiple actions - Localization in Time
Tracking Local Representations for Action Recognition


- Histogram of Oriented Gradients (HOG)
- Histogram of Optical Flow (HOF)
- Motion Boundary Histogram (MBH)
Paper 3 - Deep Learning and Action Recognition

Architectures (Before 2018)
ConvNET + LSTM (Fig. a)

- Compute deep features from image classification networks
- Benefits from ImageNet pre-training

Approach 1
- Pool deep features (as in bag of visual words) to perform action classification.
- **Drawbacks:** Ignore temporal structure. So, these approaches, for example, cannot distinguish between opening a door and closing a door.

Approach 2
- Feed deep features to LSTM to capture temporal structure.
- **Drawbacks:** LSTM using last layer features doesn’t capture low-level information (such as optical flow).
3D ConvNets (Fig. b)

- Computes spatio-temporal deep features.
- Creates hierarchical representations of spatio-temporal data.

**Drawbacks**
- A lot more parameters as compared to 2D ConvNets.
- Precludes ImageNet pre-training.
Two-Stream Networks (Fig. c, d and e)

- RGB stream + flow stream
- [1] models short temporal snapshots of videos by averaging the predictions from a single RGB frame and a stack of 10 externally computed optical.
- A recent extension [2] fuses the spatial and flow streams after the last network convolutional layer, showing some improvement on HMDB while requiring less test time augmentation (snapshot sampling).
Two-Stream Inflated 3D ConvNets

- Convert successful image (2D) classification models into 3D ConvNets.
- Given a 2D architecture (trained on, say, ImageNet), inflate all the filters and pooling kernels.
  - An $N \times N$ filter becomes $N \times N \times N$ filter.
- Bootstrap 3D filters from 2D filters using “boring videos”
  - An image can be made into a boring video by copying it repeatedly into a video sequence.
- Carefully control receptive field growth in space and time
Two-Stream Inflated 3D ConvNets

Figure 3. The Inflated Inception-V1 architecture (left) and its detailed inception submodule (right). The strides of convolution and pooling operators are 1 where not specified, and batch normalization layers, ReLu’s and the softmax at the end are not shown. The theoretical sizes of receptive field sizes for a few layers in the network are provided in the format “time,x,y” – the units are frames and pixels. The predictions are obtained convolutionally in time and averaged.
## Evaluations

<table>
<thead>
<tr>
<th>Architecture</th>
<th>UCF-101</th>
<th>HMDB-51</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Original</td>
<td>Fixed</td>
</tr>
<tr>
<td>(a) LSTM</td>
<td>81.0 / 54.2</td>
<td>88.1 / 82.6</td>
</tr>
<tr>
<td>(b) 3D-ConvNet</td>
<td>– / 51.6</td>
<td>– / 76.0</td>
</tr>
<tr>
<td>(c) Two-Stream</td>
<td>91.2 / 83.6</td>
<td>93.9 / 93.3</td>
</tr>
<tr>
<td>(d) 3D-Fused</td>
<td>89.3 / 69.5</td>
<td>94.3 / 89.8</td>
</tr>
<tr>
<td>(e) Two-Stream I3D</td>
<td>93.4 / 88.8</td>
<td>97.7 / 97.4</td>
</tr>
</tbody>
</table>

Table 4. Performance on the UCF-101 and HMDB-51 test sets (split 1 of both) for architectures starting with / without ImageNet pretrained weights. **Original**: train on UCF-101 or HMDB-51; **Fixed**: features from Kinetics, with the last layer trained on UCF-101 or HMDB-51; **Full-FT**: Kinetics pre-training with end-to-end fine-tuning on UCF-101 or HMDB-51.
### Evaluations

<table>
<thead>
<tr>
<th>Model</th>
<th>UCF-101</th>
<th>HMDB-51</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-Stream [27]</td>
<td>88.0</td>
<td>59.4</td>
</tr>
<tr>
<td>IDT [33]</td>
<td>86.4</td>
<td>61.7</td>
</tr>
<tr>
<td>Dynamic Image Networks + IDT [2]</td>
<td>89.1</td>
<td>65.2</td>
</tr>
<tr>
<td>TDD + IDT [34]</td>
<td>91.5</td>
<td>65.9</td>
</tr>
<tr>
<td>Two-Stream Fusion + IDT [8]</td>
<td>93.5</td>
<td>69.2</td>
</tr>
<tr>
<td>Temporal Segment Networks [35]</td>
<td>94.2</td>
<td>69.4</td>
</tr>
<tr>
<td>ST-ResNet + IDT [7]</td>
<td>94.6</td>
<td>70.3</td>
</tr>
<tr>
<td>Deep Networks [15], Sports 1M pre-training</td>
<td>65.2</td>
<td>-</td>
</tr>
<tr>
<td>C3D one network [31], Sports 1M pre-training</td>
<td>82.3</td>
<td>-</td>
</tr>
<tr>
<td>C3D ensemble [31], Sports 1M pre-training</td>
<td>85.2</td>
<td>-</td>
</tr>
<tr>
<td>C3D ensemble + IDT [31], Sports 1M pre-training</td>
<td>90.1</td>
<td>-</td>
</tr>
<tr>
<td>RGB-I3D, Imagenet+Kinetics pre-training</td>
<td>95.6</td>
<td>74.8</td>
</tr>
<tr>
<td>Flow-I3D, Imagenet+Kinetics pre-training</td>
<td>96.7</td>
<td>77.1</td>
</tr>
<tr>
<td>Two-Stream I3D, Imagenet+Kinetics pre-training</td>
<td>98.0</td>
<td>80.7</td>
</tr>
<tr>
<td>RGB-I3D, Kinetics pre-training</td>
<td>95.1</td>
<td>74.3</td>
</tr>
<tr>
<td>Flow-I3D, Kinetics pre-training</td>
<td>96.5</td>
<td>77.3</td>
</tr>
<tr>
<td>Two-Stream I3D, Kinetics pre-training</td>
<td>97.8</td>
<td>80.9</td>
</tr>
</tbody>
</table>

Table 5. Comparison with state-of-the-art on the UCF-101 and HMDB-51 datasets, averaged over three splits. First set of rows contains results of models trained without labeled external data.
Datasets

- UCF101
- HMDB-51
- Kinectics
From Actions to Activity to Behavior

The Heider-Simmel Illusion