Research & Development
–DRAFT–

Evaluation of current Approaches for Situation-Awareness in Autonomous Systems from Action Recognition in Video Data

---

Author:
Maximilian Schöbel

Advisors:
Prof. Dr. Erwin Prassler
Prof. Dr. Paul G. Plöger

January 12th, 2017

Contents

1 Introduction

META: Brief but concise review of the first two "W"s.

The operation of autonomous mobile systems in public, uncontrolled environments is despite active research still a difficult task.

Humans are perfectly able to act and move in unknown, crowded environments and even react successfully to new situations because they are aware of their surroundings.

An important part of Situation Awareness is the knowledge of what actions are currently performed by persons in the vicinity of an agent. This knowledge enables the agent to derive a suitable policy for its own future actions.

Actions of interest are single-person actions, person-person interactions, person-object interactions and group activities.

Enabling situation-awareness in autonomous systems is an important goal, which has an impact on other problems in autonomous systems.

Possible applications:
Pedestrian movement predicition in robotics,
Risk and danger evaluation through video surveillance in public environements,
surveillance of children or the elderly in assisted living environments,
patient monitoring in hospitals,
video retrieval (content-based video indexing),
human-computer interaction.

Requirement: Automated recognition of high-level actions.

Situation awareness is an abstract concept, which includes lots of independent manifestations and involves multiple sensory inputs.

This work focuses on approaches that process time-sequential video data, because video-cameras represent a cost-effective and widely used technology in many existing systems.

Motivation for using videos: Promising results in classfication tasks from images. Video adds another (temporal) dimension, which conveys a lot of information that can be accessed for classification as well. Video provides natural data augmentation cite:simnoyan two-stream paper (??)

1.1 Situation Awareness from video data

META: General Definition of Situation Awareness in the context of autonomous systems. Placement of Action Recognition among other vision-based methods, i.e. Action Prediction, Anomaly Detection, Event and Action Detection, Person/Pedestrian Detection, Gesture Recognition. Definitions of the above methods.

Simple case: Video contains the performance of a single human action which needs to be classified into one of several preknown classses.

General real-world case: System operates on a video stream and needs to perform continuous recognition of human actions, including detection of beginning and endings times of containing acions.

General Processing Pipeline for Action Recognition: Person Detection -> Tracking -> Action Detection -> Segmentation -> Action recognition.

Action Recognition: A part of Computer Vision research, it's goal is to automatically analyze human actions/actitvities from video-data.

Other sensory input than video possible

1.2 The Action Recognition Problem

Action Recognition is a classification-task.

Difference to face/gate recognition: Generalize over person characteristics.

Intra- and inter-class variances.

Background and recording settings.

Temporal variations.

Obtaining and labeling training data.

Main task of action recognition research: Overcome these challenges and built systems, that recognize actions robustly, even when performed by different persons in differently lighted environments at different speeds.

Main components: (i) A discriminative architecture that is able to recognise the general characteristics of different action classes while ignoring personal characteristics of different performers. (ii) Large datasets that provide this information by containing many different examples for each action class.

1.3 Survey Papers in Action Recognition (Related work)

Review of most important/recent review papers in Action Recognition with traditional and Deep Learning approaches.

1.3.1 A survey on vision-based human action recognition, Ronald Poppe (2010)

Definition of action: Uses the hierarchical classification of human motion in action primitives, actions and activities as given in Moeslund et al. (cite ??)

Action primitives are atomic movements at the limb-level.

Actions are possibly cyclic whole body movements and consist of multiple action primitives.

Activities consist of multiple actions whose subsequent execution make the movement interpretable.

Example: Action primitives: Left/right leg forward -> Action: Starting, Running, Jumping -> Activity: Jumping hurdles.

Scope: Gives a very good classification of conventional methods in human action recognition.

The discussion is split according to video representations and classification methods.

Challenges of the domain are described very well.

Deficits: No Deep Learning methods are discussed.

Datasets and benchmarks are only discussed briefly.

1.3.2 Human Activity Analysis: A Review – Aggarwal and Ryoo (2011)

Gives an approach-based taxonomy.

1.3.3 A survey on vision-based methods for action representation, segmentation and recognition – Weinland et al. (2011)

1.3.4 A survey of video datasets for human action and activity recognition – Chaquet et al. (2013)

1.3.5 A review of unsupervised feature learning and deep learning for time-series modeling – Längkvist et al. (2014)

1.3.6 Going Deeper into Action Recognition: A survey – Herath et al. (2016)

Definition of action:

2 Conventional Methods in Action Recognition

META: Condensed overview and description of conventional Methods in action Recognition using the taxonomy of Aggarwal and Ryoo's fine survey paper. More detailed description of methods using local-features, since these have become the standard approach in action recognition after Aggarwal and Ryoo's overview.

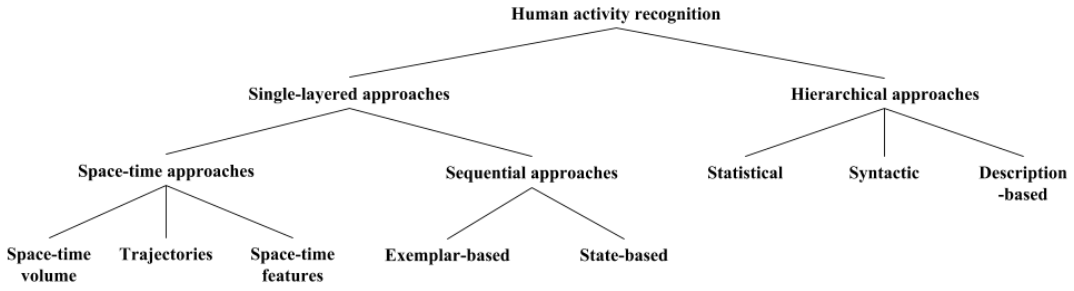


Figure 1: Approach-based taxonomy for conventional methods in human activity recognition as given by Aggarwal and Ryoo[1]

3 Main components in action recognition using local features: Feature Extraction, Representation Building, Classification.

Methods for feature extraction: Interest point detectors or dense sampling.

Space-time interest point detectors: Harris3D[2], Cuboids[3], Hessian Detector[4]

Descriptors for 3D volumes around previously detected space-time interest points: Histogram of Gradient HOG[5], Histogram of Optical Flow (HOF)[6], 3D Histogram of Gradient (HOG3D)[7], Extended SURF (ESURF)[4]

"standard approach to video classification" described in [8]

2.1 Overview

2.2 Local Features

2.2.1 Feature extractors

2.2.2 Aggregation Methods

Bag of visual Words paradigm

Fisher Vector

3 Deep Learning Methods in Action Recognition

META: Review of approaches that use Deep Learning.

3.1 Spatio-Temporal Networks

I.e. convolutional methods.

3.1.1 3D Convolutional Neural Networks for Human Action Recognition – Ji et al. (2010/2013)

NOTE:Paper was first published in 2010, the publication of 2013 is more popular however.

In this work [9] the authors porpose 3D convolution for action recognition from video, which processes spatial as well as temporal information in a convolutional layer.

In regular convolutional neural networks 2D convolutions are applied in the convolutional layers to extract features from the feature maps in the previous layer. More formally in the notation of the authors, the value of feature map j in the ith layer at spatial position (x, y) is given by:

$$v_{ij}^{xy} = \tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}\right)$$

w_{ijm}^{pq} denotes the value at position (p, q) of the kernel, that performs convolution on feature map m of the previous layer, resulting in feature map j in layer i. P_i and Q_i denote the dimensions of the kernel in x- and y-direction respectively. Tensor w therefore represents all kernels that produce the feature maps in layer i through convolution.

The two inner sums carry out the convolutional operation on feature map m of the previous layer, which is then combined with the results for the other feature maps by summation over m, added with a bias and fed into a non-linear function ($\tanh(\cdot)$) to result in the value of the current feature map.

Note: The convolutional operation used here is called cross-correlation, which differs from mathematical discrete convolutions in that the kernel is not flipped. This results in a non-commutative operation as described in chapter 9 of cite deep learning book ??

The authors propose an extension of 2D convolutions by using a three dimensional kernel. More formally, in the notation as above, the value of feature map j in the ith layer at position (x, y, z) is given by:

$$v_{ij}^{xyz} = \tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)}\right)$$

As above, w_{ijm}^{pqr} denotes the value of the now three dimensional kernel at position (p, q, r), which performs convolution on the mth feature map of the previous layer. R_i denotes the dimension of the kernel in temporal direction.

Based on the 3D convolution, the authors design a neural network architecture, that takes an input of 7 frames of size 60x40.

The network is evaluated as part of an action detection and recognition system on the TRECVID (TREC Video Retrieval Evaluation) data, which consists of surveillance videos recorded at London Gatwick Airport.

The details of the architecture are described below:

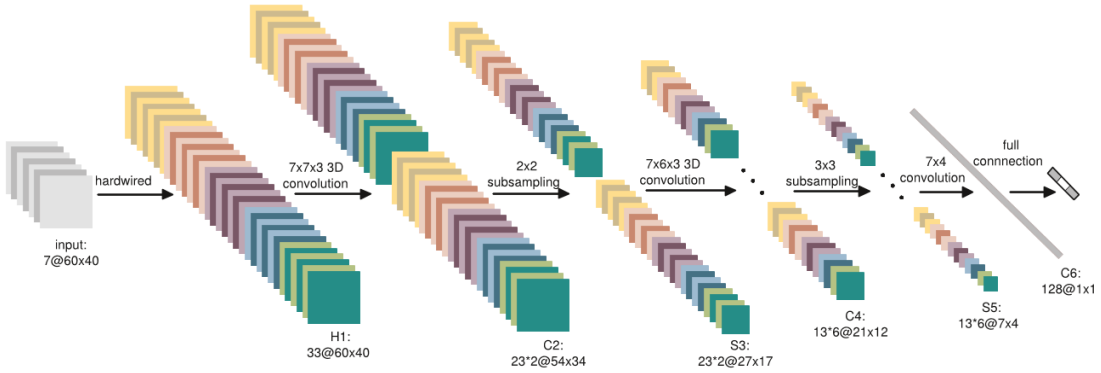


Figure 2: 3D CNN architecture developed for human action recognition on the TRECVID 2008 development dataset [9]

At first, hard wired kernels are applied to the input frames, which extract gray values, gradients along the horizontal and vertical direction in the frames and the optical flow between two consecutive frames. This results in 33 feature maps, organized in 5 different channels: gray, gradient-x, gradient-y, optflow-x and optflow-y.

In the first convolutional layer (C2) two 3D kernels with dimesions 7x7x3 (7x7 in the spatial dimension and 3 in the temporal dimension) are applied at each of the five channels separately. This results in 2x5 channels with a total of 46 feature maps. The first convolutional layer therefore necessitates $2\times5\times7\times7\times3$ (kernel-weights) $+2\times5$ (biases) $= 1480$ trainable parameters.

After subsampling layer S3, three different kernels with size 7x7x3 are applied to each of the 2x5 channels of the previous layer.

The last convolutional layer then performs, after 3x3 subsampling, 2D convolutions to obtain 128 feature maps of dimension 1x1. These feature maps can be seen as a 128 dimensional feature-vector representation of the input.

The resulting feature-vector is classified by a fully connected layer (into three different classes).

The architecture has 295.458 trainable parameters in total, which are initialized randomly and learned by online error back-propagation as in cite lecun 1998 ??. NOTE: Perhaps add comparison to recent image recognition networks here. ??

The authors tried other layouts but conclude that the one described above works best. The performance was evaluated on the TRECVID 2008 development dataset ?? and on the KTH dataset ??.

TRECVID:
The TRECVID 2008 development dataset contains 49-hour annotated surveillance videos from five different cameras on five days. The authors train their model to recognize three different action classes from the dataset: CellToEar, ObjectPut and Pointing. The other classes were used to generate negative training examples.

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

Figure 3: Number of samples per class from the TRECVID 2008 development dataset [9]

Since the dataset provides continuous videos with several persons in a real-world scence, the authors apply a human detector and a detection-driven tracker, to keep track of the heads in the scene. This information is used to extract a bounding box around a person, as soon as an action is performed. Six other equally sized bounding boxes are sampled at the same location from three frames before and after the intial frame. Between those samples, one frame is omitted which results in a temporal step size of 2.

The contents of these 7 bounding boxes are stacked and taken as input to the 3D CNN architecture for classification.

Figure 4: Example scenes taken by different cameras with the results of the human detection and tracking [9]

In order to compare their 3D CNN model to other techniques, the authors also evaluate:

1. A frame based 2D CNN model
2. Extraction of dense SIFT features from the gray images of the 3D CNN input cube, which are then aggregated using the BoW-Paradigm and classified through a linear SVM
3. Extraction of dense SIFT features from the motion edge history images (MEHI) of the 3D CNN input cube, which are then aggregated as above.

The 3D CNN model outperfomed the other approaches on the TRECVID 2008 development dataset on all classes except Pointing, where the 2D frame-based CNN performed best. The authors note, that the positive example for Pointing are significantly larger than for the other classes and conclude, that their architecture performs best, when little positive examples are present.

KTH:
Additionally, the 3D CNN model was evaluated on the KTH dataset. It achieves an overall accuracy of 90.2%.

DISCUSSION: Authors cite the work of schindler and van gool from 2008 and state, that 5-7 frames are enough for action recognition. Schindler and van gool: 5-7 frames are enough to recognize SIMPLE actions.

3.1.2 Sequential Deep Learning for Human Action Recognition – Baccouche et al. (2011)

Baccouche et al. [10] identify two issues with the previous approaches for extending convolutional neural networks to the video domain, i.e. 3D convolutions as in [9] and [11]:

1. They rely on hand crafted inputs (hard wired preprocessing of the data to produce image gradients or optical flow in the first processing layer).
2. The models typically process less than 15 input frames, and therefore only classify short sub-clips, not the entire video.

To address these issues, the authors design a two-step deep architecture consisting of:

1. A convolutional spatio-temporal feature extractor network, based on 3D convolutions.
2. A recurrent neural network classifier, that incorporates LSTM cells [12], to classify the entire sequence of previously extracted spatio-temporal features.

The architecture is trained and evaluated on the KTH dataset.

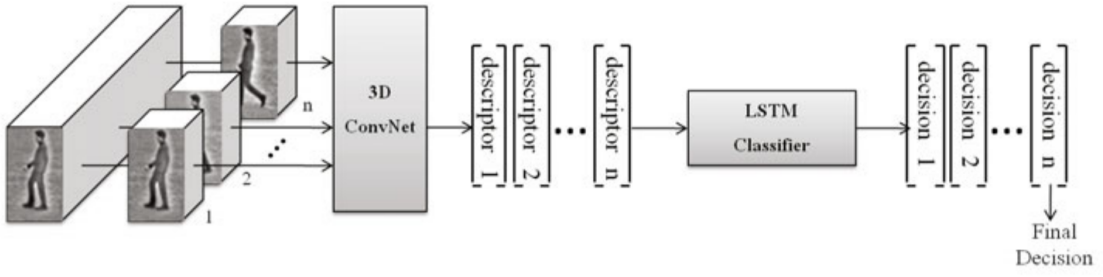


Figure 5: Overview of the two-steps architecture consisting of a 3D ConvNet and a RNN classifier. [10]

The ConvNet architecture of this approach is depicted in figure 5. It is comparable to the one of Ji et al. [9] in that it incorporates 3D convolutions, but works on raw input pixels instead of hand-crafted inputs.

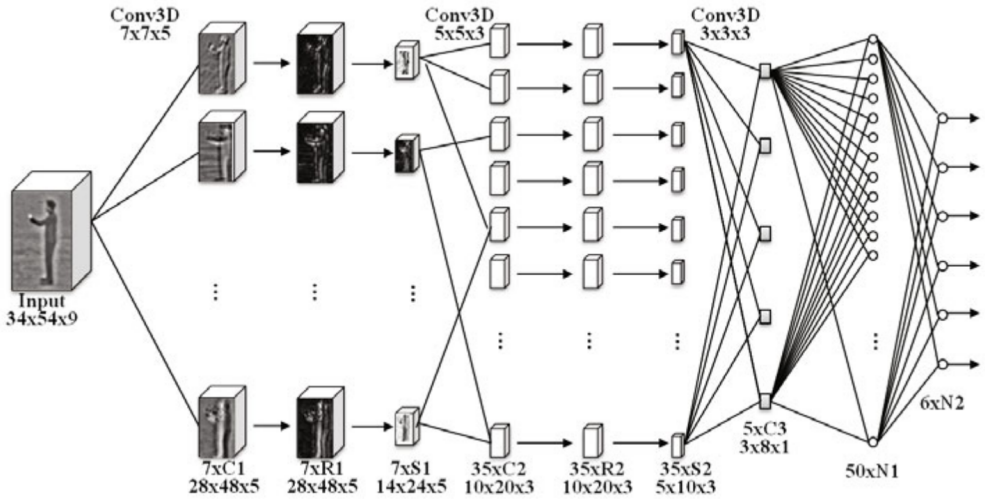


Figure 6: 3D ConvNet architecture for the extraction of spatio-temporal features [10]

Input to the network is formed by stacking 9 succesive frames of spatial size 34x54, which results in an input cube of dimension 34x54x9.

The configuration of it's layers is as follows:

1. First convolutional layers C1 computes 7 feature maps, by convolving a 3D 7x7x5 kernel for each feature map with the input cube.
2. Layer R1 and S1 perform rectification (building of the absolute value) and subsampling with a spatial factor of 2 respectively.
3. Second convolutional layer C2 computes 35 feature maps. Each of the 7 feature maps in layer S1 is connected to two different feature maps in C2 (resulting in 14 feature maps) and each pair of different feature maps of S1 is connected to one feature map in C2 (resulting in additional 21 feature maps, leaving a total of 35 feature maps).
4. Layer R1 and S2 are equivalent to R1 and S1.
5. Third (and last) convolutional layer C3 computes five features maps, which are fully connected to all feature maps in previous layer S2 by 3x3x3 kernels. These five feature maps have dimension 3x8x1, rendering the raw input encoded into a 120 dimensional feature vector.
6. For training the individual ConvNet model, the 120D feature vector is finally fed into two fully connected layers with 6 output neurons (one for each class of the KTH dataset).

The ConvNet model embeds a total of 17,169 trainable parameters, which is about 15 times less than the number of parameters trained by Ji et al. [9] (295,458 parameters).

The authors use the same training algorithm as Ji et al., which was proposed by cite lecun 1998 ??: online Backpropagation with momentum adapted to weight-sharing.

In order to take the temporal evolution of movements in a video into account, the authors propose to train a recurrent neural network with one hidden layer of LSTM cells for classifying sequences of previously extracted CNN feature-vector representations.

Several configurations were tested and 50 LSTM cells in the hidden layer were found to be a good compromise between training time and performance.

The output of the third convolutional layer C3 of the ConvNet architecture is fed into the recurrent neural network as input at each time step.

The LSTM cells in the hidden layer are fully connected with the input and have recurrent connections among each other.

The output layer is connected to the outputs of the LSTM cells.

The RNN is trained as part of the two-step architecture using online backpropagation through time with momentum.

The architecture is evaluated on the KTH dataset. To build the inputs to the network, the following simple pre-processing steps are applied: down-sampling by a factor of 2

horizontally and vertically, extraction of the bounding boxes around persons, application of 3D Local Contrast Normalization on a 7x7x7 neigbourhoood.

The authors find the 3D ConvNet model to yield a recognition rate of 91.04%, when the classification is done by majority voting over several short sub-sequences of the test-video (without using the LSTM classifier). Which is comparable to other deep learning approaches at that time and almost the same result as obtained by Ji et al. [9] (90.2%), although the model requires about 15 times less paramerters.

When using the LSTM classifier network, performance increases of about 3%. The combined two-steps architecture consistently outperformed other deep models at that time.

Evaluation scheme: five best results averages over 30 trials.

3.1.3 Large-scale Video Classification with Convolutional Neural Networks – Karpathy et al. (2014)

The author's contribution in [8] is four-fold:

1. They gather the Sports-1M dataset containing a collection of 1 million automatically annotated sports videos from YouTube.
2. They study multiple approaches for extending convolutional neural networks to incorporate spatio-temporal information.
3. They propose a multiresolution convolutional architecture with the goal of speeding up the training at no cost in accuracy.
4. They retrain the top layers of a network on the UCF-101 dataset, which has previously been trained on the Sports-1M dataset and thereby achieve significant increase in performance towards training the network on UCF-101 alone (transfer learning).

Here, the architectures will be reviewed (contributions 2. and 3.), for a description and review of the Sports-1M dataset and the transfer learning paradigm see section 4 of this work.

To obtain fixed-sized inputs for their architectures, the authors interpret an entire video as a set of short, fixed-sized video clips.

The authors first implement a baseline CNN architecture to classify videos based on a single frame. The model is similar to the winning network of the 2012 ImageNet Classification Challenge cite ??, but receives slightly smaller inputs (here 170x170x3 against 224x224x3 in the ImageNet model).

Based on the single frame architecture, several extensions for processing temporal information in the network are being investigated. These types of fusion methods are depicted in figure 7.

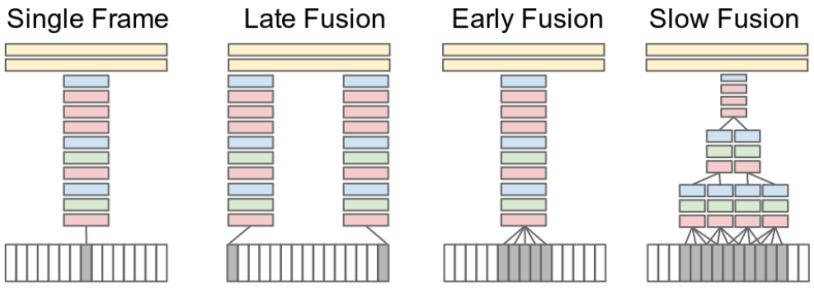


Figure 7: Different methods for fusing temporal with spatial information in convolutional neural network architectures. Red, green and blue denotes convolutional, normalization and pooling-layers. Grey colored inputs are single video frames. [8]

Early Fusion:
Temporal information is incorporated in the network on the pixel level by extending the convolutional kernel in the first layer to be of dimension $11 \times 11 \times 3 \times T$, where T is the temporal extend (here used: $T = 10$).

Late Fusion:
Two separate single-frame networks with shared parameters and without their individual classification layers are used on input frames with a temporal distance of 15 frames. Two shared fully connected layers then merge the individual network's information and classify the input. The fully connected layers are able to compute motion information by comparing the feature representations of the two single-frame networks.

Slow Fusion:
Temporal information is processed throughout the network by extending the kernels of each convolutional layer in time, as done in the first layer of the Early Fusion approach. Thereby higher layers progressively processes more spatio-temporal information of the input frames. These kinds of convolutions have also been applied in [9] and baccouche ??. The first convolutional layer uses a temporal extend of $T = 4$, while the second convolutional layer uses a temporal extend of $T = 2$, which allows the third layer to access the information of all 10 input frames.

Since the training time of a network heavily influences the amount of experiments that can be conducted on different hyperparameter settings, it is of great interest to reduce the training time for neural networks (for CNNs: in the order of weeks) while maintaining the accuracy.

The authors propose a multiresolution architecture as shown in figure 8.

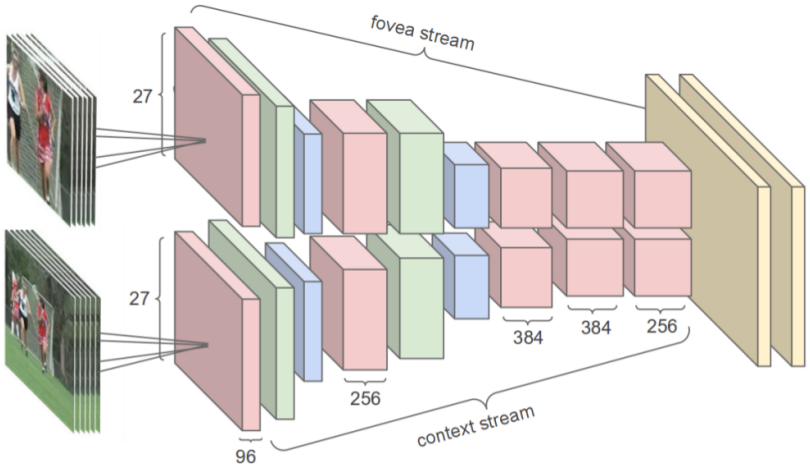


Figure 8: Multiresolution CNN architecture [8]

The overall multi-resolution network processes videos of size 178x178 pixel as input.

The context stream receives a downsampled version of the input frames, which has half the size of the original input (89x89 pixel). The fovea stream works on just the 89x89 pixel sized center of the original input frames.

Both streams are implemented as the same CNN architecture.

The outputs of both streams are fed into the first of two fully connected layers, where their information is merged and a class prediction is calculated.

The multiresolution architecture recquires half the input dimensionality compared to processing the raw 178x178 pixel input video in one stream.

Thereby the authors achieved a reduction in training time by a factor of 2-4.

The authors use downpour stochastic gradient descent cite ?? for training the networks on the Sports-1M dataset using a computing cluster. 70% of the dataset were used as training data, 10% as the validation set and 20% as the test set.

At testing time, 20 short clips are sampled from the current test-video and each clip is presented to the network individually. Each clip is passed through the network 4 times (with different crops and flips) and the result is averaged to produce a robust class prediction. The video-level predictions are computed from the clip-level predictions simply by averaging.

In addition to comparing different CNN architectures, the authors also implement a feature-histogram baseline approach by extracting multiple kinds of hand-crafted features from each video and aggregating them according to the bag-of-words paradigm. The resulting feature vector is classified using multilayer neural network, with rectified linear activation units.

Model	Clip Hit@1	Video Hit@1	Video Hit@5
Feature Histograms + Neural Net	-	55.3	-
Single-Frame	41.1	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**	**60.9**	**80.2**
CNN Average (Single+Early+Late+Slow)	41.4	63.9	82.4

Table 1: Results of different architectures on the Sports-1M dataset. Hit@k denotes the fraction of test samples, that had at least one of their class labels included in the top k predictions. [8]

The performance of the studied architectures is shown in table 1:

The authors find their deep models to consistently outperform the hand-crafted feature based baseline.

Compared to each other, the deep models performed similarly well, despite their different convolutional architectures. The slow fusion model however performed best, by a small margin.

According to the authors interpretation, the "variation among different CNN architectures turns out to be surprisingly insignificant"[8].

The single-frame model performs noticably well on it's own. The authors suspect, that the motion aware networks suffer from camera movements such as translations or zoom.

"we find that a single-frame model already displays very strong performance, suggesting that local motion cues may not be critically important, even for a dynamic dataset such as Sports. An alternative theory is that more careful treatment of camera motion may be necessary (for example by extracting features in the local coordinate system of a tracked point, as seen in [25])."

3.1.4 Learning Spatiotemporal Features with 3D Convolutional Networks – Tran et al. (2015)

3.1.5 Long-term Temporal Convolutions for Action Recognition – Varol et al. (2016)

Varol, Laptev, and Schmid [13] address, similar to Baccouche et al. [10], the common drawback of recent CNN extensions to action recognition, that class labels are learned

for very short video subsequences only, i.e. a temporal extend of only 1-16 input frames can be processed by the architectures at a time.[9][8][14]

Instead of classifying convolutionally extracted spatio-temporal features with a recurrent neural network, which is able to take their temporal evolution into account (as done by Baccouche et al. [10]), the authors of this publication study the effects of increasing the number of input frames of a 3D convolutional architecture on the performance in action recognition.

The authors name their approach long-term temporal convolutions. Their contribution in this work is two-fold:

1. Systematical evaluation of the influence of the temporal extend, i.e. the number of input frames $T = \{20, 40, 60, 80, 100\}$, to the performance of a 3D CNN architecture for video action recognition.
2. Confirming the importance of high-quality optical flow inputs, in order to learn accurate video features with a 3D CNN architecture for human action recognition.

Processing an increased temporal extend has to be compensated with a decreased spatial resolution, in order to not exceed computational limitations. The studied architectures therefore process spatial resolutions of 58x58 or 71x71 pixels.

The authors' 3D CNN architecture, for which different temporal extends are studied, is shown in figure 9.

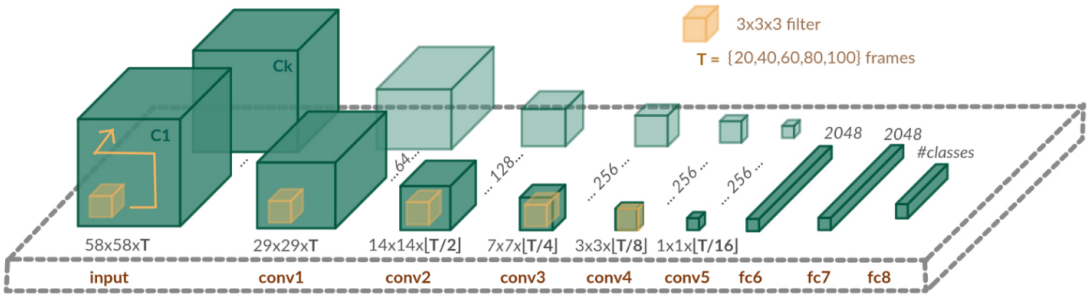


Figure 9: 3D CNN architecture for evaluating the influence of temporal extend on action recognition performance. [13]

Architectural details:

- 5 space-time, i.e. 3D, convolutional layers with 64, 128, 256, 256 and 256 feature maps.
- 3x3x3 space-time convolutional kernels are used in every convolutional layer.
- After each convolutional layer: One layer of rectified linear units (ReLU) and one layer of max-pooling with filter size 2x2x2 except in the first layer where it is 2x2x1.

- 3 fully connected layers at the end with sizes 2048, 2048 and number of classes.

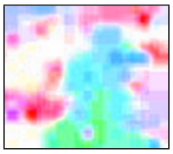
As a baseline approach the authors implement their architecture with inputs of size 112x112x16, beacause it can directly be compared to the work of [14]. Initially, the authors study the performance between the 16 frame baseline and another 60 frame network, which takes inputs of size 58x58x60 (spatial size has to be decreased to remain computationally tractable).

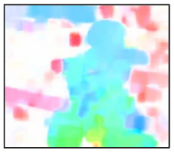
To systematically investigate the influence of temporal extend on the action recognition performance, the authors implement their architecture with different numbers of input frames $T = \{20, 40, 60, 80, 100\}$ and different spatial resolutions $58 \times 58, 71 \times 71$.

Additionally the influence of using optical flow inputs on the performance, specifically the influence of different sources of optical flow, is evaluated. These namely are:

1. MPEG flow, which directly can be obtained from the video encoding. It is a fast alternative to regular optical flow estimators, but has low spatial resolution and is not available for all video frames.
2. Farneback optical flow estimator cite ??, which is fast, but calculates noisy flow fields.
3. Brox optical flow estimator cite ??, which generates highest quality optical flow fields but is slower than the other two methods.


RGB


MPEG flow


Farneback


Brox

Input	Clip	Video
RGB	57.0	59.9
MPEG flow	58.5	63.8
Farneback	66.3	71.3
Brox	74.8	79.6

Figure 10: You really need to add a caption here!

The networks are trained on the UCF-101 and HMDB-51 dataset using stochastic gradient descent with negative log-likelihood criterion.

During training the authors randomly sample video subsequences with the desired spatial and temporal dimensions from the input videos, which in general have a higher resolution and are longer than needed. The authors name this form of pre-processing random clipping.

Another method used during training is called multiscale cropping. Smaller input volumes as needed are cropped from the training videos and then rescaled to fit to the input dimensions of the network. The size of the crop is determined by rendomly selected factors for frame width and height from $1.0, 0.875, 0.75, 0.66$. Finally the input is flipped with a probability of 50%.

Test time procedure.

3.1.6 Summary and Comparison

Most of the current CNN methods use architectures with 2D convolutions, enabling shift-invariant representations in the image plane. Meanwhile, the invariance to translations in time is also important for action recognition since the beginning and the end of actions is unknown in general. Laptev 2016 Note: No need for action detection???

3.2 Multiple Stream Networks

The most successfull architecture at action recognition. They are equally powerful as the improved dense trajectories approach. cite TDD ??

These approaches use the decomposability of videos into a spatial component (analysing different frames) and a temporal component (analysing the change between frames) for action recognition.

The first architecture of this kind was proposed in 2014 from Simonyan and Zisserman.

3.2.1 Two-Stream Convolutional Networks for Action Recognition in Videos - Simonyan and Zisserman (2014)

The authors propose a novel architecture for action recognition with two separate recognition streams (spatial- and temporal-stream) which are combined by late fusion.

The authors evaluate two different fusion methods: building the average of both network's outputs and training a linear multi-class SVM on stacked L_2-normalised softmax scores.

This approach is motivated by the two-streams hypothesis cite (??), according to which the human visual cortex contains two paths: the ventral stream for object recognition and the dorsal stream for recognising motion.

Both streams are implemented as deep CNNs, with the rectification activation function for all hidden units.

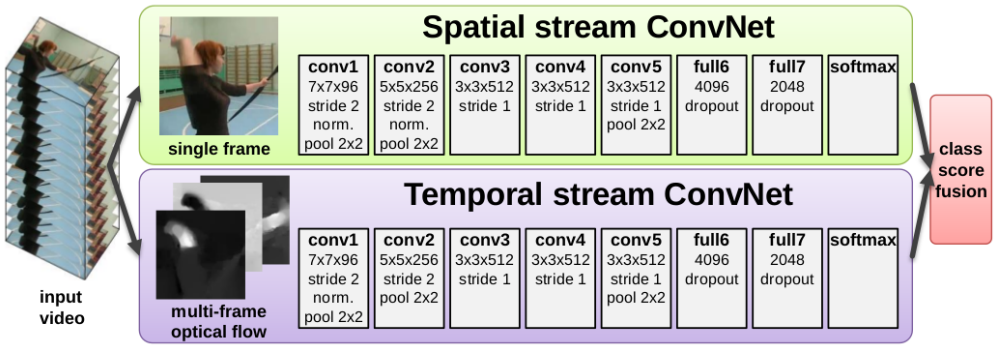


Figure 11: Two-stream architecture for video classification, depicting the spatial- and temporal-stream with implementation details as deep Convolutional Neural Network [15]

Spatial stream: Takes single video frames as input. Performs action recognition from still images and is fairly competitive on its own. Basically an image recognition architecture. Advantage: Can be pre-trained using large amount of image data, here from the ImageNet challenge dataset cite (??).

Spatial part of a video, i.e. the individual static frames, convey information about the objects and persons in the scene.

Temporal stream: is trained to recognize actions from motions given in the form of dense optical flow.

The temporal part of a video, i.e. the change between frames, conveys information about the movement of the observer (camera) and the movement of objects in the scene.

The second normalisation layer was removed from the the temporal stream network in order to reduce memory usage.

The authors propose two methods for constructing the input to the temporal network by stacking optical flow displacement fields along several consecutive frames of the input video.

Optical Flow Stacking:
A dense optical flow field $\mathbf{d}_t(u, v)$ of two consecutive frames at times t and $t + 1$ can be thought of as a two dimensional vector-field, which maps the displacement of each pixel along the transition from frame t to $t+1$. In this case $u, v \in \mathbb{N}$, $1 \leq u \leq w$ and $1 \leq v \leq h$ where w and h are the width and height of the video frames.

The horizontal and vertical components $d_t^x(u, v)$, $d_t^y(u, v)$ can be interpreted as image channels.

This method constructs the input volume $I_\tau \in \mathbb{R}^{w \times h \times 2L}$ of the temporal stream network by stacking the horizontal and vertical components of the dense optical flow field along L consecutive frames, beginning at time τ. Formally, with $1 \leq k \leq L$:

$$I_\tau(u, v, 2k - 1) = d^x_{\tau+k-1}(u, v)$$
$$I_\tau(u, v, 2k) = d^y_{\tau+k-1}(u, v)$$

Trajectory Stacking:
Instead of sampling at fixed locations in each frame, this methods samples the dense optical flow field along the motion trajectories of the initial points in frame τ.

Let $\mathbf{p}_k$ denote the motion trajectory of initial point (u, v). With $1 < k \leq L$ and $\mathbf{p}_1 = (u, v)$ the trajectory is recursively defined by:

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \mathbf{d}_{\tau+k-2}(\mathbf{p}_{k-1})$$

The input volume can then be constructed by sampling the horizontal and vertical optical flow components along these trajectories.

$$I_\tau(u, v, 2k - 1) = d^x_{\tau+k-1}(\mathbf{p}_k)$$
$$I_\tau(u, v, 2k) = d^y_{\tau+k-1}(\mathbf{p}_k)$$

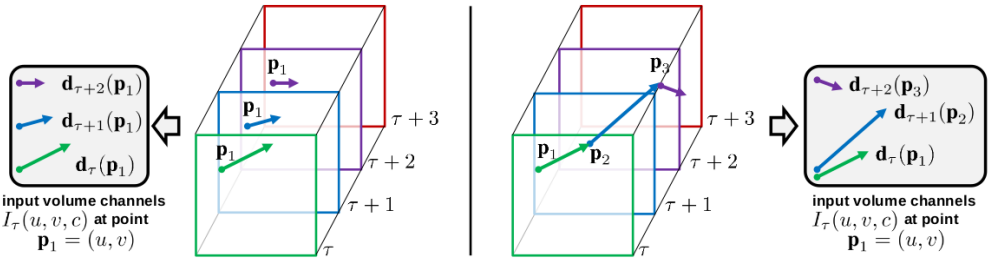


Figure 12: Construction of input volumes from multi-frame optical flow. Left: Optical Flow Stacking. Right: Trajectory Stacking [15]

Since the Convolutional Networks require fixed sized inputs, the authors sample a $224 \times 224 \times 2L$ subvolume from $I_\tau \in \mathbb{R}^{w \times h \times 2L}$ and use it as the temporal networks input.

By using optical flow, the authors explicitly incorporate a representation of motion in their action recognition architecture.

The authors further describe two optional techniques that are evaluated for constructing the inputs with either one of optical flow stacking methods.

1. **Bi-directional Optical Flow**: The input volume I_τ is created by using regular forward optical flow from frame τ to $\tau + L/2$ and additionally calculated optical flow from frame τ to $\tau - L/2$ in the backwards direction. Stacking the horizontal and vertical components of these optical flow fields results in an input volume of length $L/2$ around frame τ in both directions.
2. **Mean flow subtraction**: The displacement vectors between two frames can dominantly be caused by global movement of the camera. Compared to regular camera-motion compensation, the authors use a simpler approach and just subtract the mean vector from each displacement field $\mathbf{d}_t$.

An advantage of separating the spatial and the temporal stream is the possibility of pre-training the spatial net with large pre-existing image datasets (here ImageNet ILSVRC-2012 ??).

For training the temporal stream network with video-data, the authors address the general problem of video action recognition, that the existing datasets are too small in comparison to their image-dataset counterparts.

The authors use, according to the multi-task learning paradigm [16], the UCF-101 and the HMDB-51 dataset simultaneously for training the temporal stream network (see chapter ??).

By training a network on several tasks (here UCF-101 classification and HMDB-51 classification), the network learns more general video representations, since the second task acts as regulariser and more data can be utilized.

The authors implement this by using two softmax classification layers, one for each dataset. Each softmax layer has its own loss function and the sum of the individual losses is taken as the overall training loss for computing the weight updates by backpropagation.

Training for both networks is conducted with mini-batch gradient descent with 256 randomly selected videos at each iteration. From each of those videos, a single frame is randomly chosen, a 224×224 sub-image is randomly cropped, randomly horizontal flipped, RGB jittered and then used as training input for the spatial stream network.

An optical flow volume is constructed for this selected frame as described above. For optical flow computation the authors use a fast implementation (0.06s per pair of frames) from the OpenCV toolbox. Despite it's speed, on-the-fly computation of optical-flow would be a bottleneck and is therefore pre-computed and stored for the complete datasets.

The creators of UCF-101 and HMDB-51 provide three splits of their datasets into training- and testing-data. The standard evaluation procedure is to report the average accuracy over those three splits, which the authors follow in this work as well.

The authors build their final design of the two-stream architecture by evaluating different setups for the spatial and temporal stream network on their own using UCF-101 (split 1).

Besides using two different dropout rate (0.5 and 0.9), the performance of the spatial-stream network is evaluated for:

1. Training the complete network from scratch on UCF-101.
2. Pre-training the network on ILSVRC-2012 and fine-tuning it on UCF-101.
3. Pre-training the network on ILSVRC-2012 and fine-tuning of only the last (classification) layer.

For evaluating the temporal-stream network a fixed dropout rate of 0.9 is chosen, because the network needs to be trained from scratch. The performance is then measured for:

1. Using one or several (stacked) optical flow displacement fields $L \in 1, 5, 10$.
2. Regular optical flow stacking
3. Trajectory stacking
4. Using bi-directional optical flow
5. Using mean subtraction

The findings are presented in table 2.

Spatial-stream network: The authors decided on pre-trained the network on ILSVRC and fine-tuning only the last layer.

Temporal-stream network: Mean subtraction and stacking multiple optical flows is beneficial, so $L = 10$ is used as the default setting. Regular optical flow stacking performs

better than trajectory stacking and bi-directional optical flow only yields slight improvement against forward optical flow. Therefore regular forwars optical flow stacking is chosen for the temporal-stream network.

The authors highlight that the temporal-stream network significantly outperforms the spatial-stream network, which confirms the importance of motion information for action recognition from video.

(a) **Spatial ConvNet.**

Training setting	Dropout ratio	
	0.5	0.9
From scratch	42.5%	52.3%
Pre-trained + fine-tuning	70.8%	**72.8%**
Pre-trained + last layer	**72.7%**	59.9%

(b) **Temporal ConvNet.**

Input configuration	Mean subtraction	
	off	on
Single-frame optical flow ($L = 1$)	-	73.9%
Optical flow stacking ($L = 5$)	-	80.4%
Optical flow stacking ($L = 10$)	79.9%	**81.0%**
Trajectory stacking ($L = 10$)	79.6%	80.2%
Optical flow stacking ($L = 10$), bi-dir.	-	**81.2%**

Table 2: Performance of the individual convolutional networks on UCF-101 (split 1) [15]

After having found the optimal configurations for the individual temporal-stream and spatial-stream networks, the authors evaluate different fusion methods (averaging and SVM) and find that fusion by SVM performs best. The fused network's performance significantly improves over the individual network's performance, which implies, that the spatial and temporal recognition stream are complementary.

Method	UCF-101	HMDB-51
Spatial stream ConvNet	73.0%	40.5%
Temporal stream ConvNet	83.7%	54.6%
Two-stream model (fusion by averaging)	86.9%	58.0%
Two-stream model (fusion by SVM)	**88.0%**	**59.4%**

Table 3: Mean accuracy over three splits on UCF-101 and HMDB-51 [15]

The results in table 3 show, that fusion by SVM works best.

3.2.2 Action recognition with trajectory-pooled deep-convolutional descriptors – Wang et al. (2014)

3.2.3 Fusing Multi-Stream Deep Networks for Video Classification – Wu et al. (2015)

3.2.4 Towards good practices for very deep two-stream convnets – Wang et al. (2015)

3.3 Recurrent Models

3.4 Generative Models

Restricted Boltzmann Machine

3.5 Temporal Coherency Networks

4 Datasets and Benchmarks in Action Recognition

"From a practical standpoint, there are currently no video classification benchmarks that match the scale and variety of existing image datasets because videos are significantly more difficult to collect, annotate and store." cite large-scale image classification

"In particular, commonly used datasets (KTH, Weizmann, UCF Sports, IXMAS, Hollywood 2, UCF-50) only contain up to few thousand clips and up to few dozen classes. Even the largest available datasets such as CCV (9,317 videos and 20 classes) and the recently introduced UCF-101[22] (13,320 videos and 101 classes) are still dwarfed by available image datasets in the number of instances and their variety [7]." cite large-scale image classification

"In [4], Gao et al. presented a comprehensive study on the influence of the evaluation protocol on the final results. It was shown that the use of different experimental configurations can lead to performance differences up to 9%." "Action recognition methods are usually directly compared although they use different testing protocols or/and datasets (KTH1 or KTH2), which distorts the conclusions." cite sequential deep learning for human action recognition.

4.1 Review of Datasets for Human Action Classification

Review of the most important currently existing datasets, focus on newest ones (since 2013)

Reference dataset survey paper.

4.2 Alternative Benchmarks for Action Recognition Algorithms

4.3 Data Augmentation

refer to Imagenet classification with deep convolutional neural networks.

4.4 Inter-Dataset Approaches

4.4.1 Multi-task learning

4.4.2 Transfer learning

See Karparthy 'Large-scale video classification with convolutional neural networks' 2014

A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carls- son. CNN features off-the-shelf: an astounding baseline for recognition

4.4.3 Unsupervised pre-training

5 Evaluation

What do we need, what do we have, what is best suited so far?

References

[1] Jake K. Aggarwal and Michael S. Ryoo. "Human Activity Analysis: A Review". In: *ACM Computing Surveys (CSUR)* 43.3 (2011). 01121, p. 16. URL: http://dl.acm.org/citation.cfm?id=1922653 (visited on 05/23/2016).

[2] Ivan Laptev. "On Space-Time Interest Points". In: *International Journal of Computer Vision* 64 (2-3 2005). 02614, pp. 107–123. URL: http://link.springer.com/article/10.1007/s11263-005-1838-7 (visited on 06/01/2016).

[3] Piotr Dollár et al. "Behavior Recognition via Sparse Spatio-Temporal Features". In: *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*. 02076. IEEE, 2005, pp. 65–72. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1570899 (visited on 05/16/2016).

[4] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. "An Efficient Dense and Scale-Invariant Spatio-Temporal Interest Point Detector". In: *European Conference on Computer Vision*. 00647. Springer, 2008, pp. 650–663. URL: http://link.springer.com/chapter/10.1007/978-3-540-88688-4_48 (visited on 10/18/2016).

[5] Navneet Dalal and Bill Triggs. "Histograms of Oriented Gradients for Human Detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. 15268. IEEE, 2005, pp. 886–893. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1467360 (visited on 07/18/2016).

[6] Ivan Laptev et al. "Learning Realistic Human Actions from Movies". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. 02233. IEEE, 2008, pp. 1–8. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4587756 (visited on 05/25/2016).

[7] Alexander Klaser, Marcin Marsza\lek, and Cordelia Schmid. "A Spatio-Temporal Descriptor Based on 3d-Gradients". In: *BMVC 2008-19th British Machine Vision Conference*. 00929. British Machine Vision Association, 2008, pp. 275–1. URL: https://hal.inria.fr/inria-00514853/ (visited on 10/18/2016).

[8] Andrej Karpathy et al. "Large-Scale Video Classification with Convolutional Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 00537. 2014, pp. 1725–1732. URL: http://www.cv-foundation.org/openaccess/content_cvpr_2014/html/Karpathy_Large-scale_Video_Classification_2014_CVPR_paper.html (visited on 05/03/2016).

[9] Shuiwang Ji et al. "3D Convolutional Neural Networks for Human Action Recognition". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.1 (2013). 00485, pp. 221–231. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6165309 (visited on 04/27/2016).

[10] Moez Baccouche et al. “Sequential Deep Learning for Human Action Recognition”. In: *International Workshop on Human Behavior Understanding*. 00105. Springer, 2011, pp. 29–39. URL: http://link.springer.com/chapter/10.1007/978-3-642-25446-8_4 (visited on 11/02/2016).

[11] Ho-Joon Kim, Joseph S. Lee, and Hyun-Seung Yang. “Human Action Recognition Using a Modified Convolutional Neural Network”. In: *International Symposium on Neural Networks*. 00018. Springer, 2007, pp. 715–723. URL: http://link.springer.com/chapter/10.1007/978-3-540-72393-6_85 (visited on 11/02/2016).

[12] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural computation* 9.8 (1997). 02787, pp. 1735–1780. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6795963 (visited on 11/03/2016).

[13] Gül Varol, Ivan Laptev, and Cordelia Schmid. “Long-Term Temporal Convolutions for Action Recognition”. In: *arXiv preprint arXiv:1604.04494* (2016). 00001. URL: https://hal.inria.fr/hal-01241518/ (visited on 06/13/2016).

[14] Du Tran et al. “Learning Spatiotemporal Features With 3D Convolutional Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 00081. 2015, pp. 4489–4497. URL: http://www.cv-foundation.org/openaccess/content_iccv_2015/html/Tran_Learning_Spatiotemporal_Features_ICCV_2015_paper.html (visited on 06/21/2016).

[15] Karen Simonyan and Andrew Zisserman. “Two-Stream Convolutional Networks for Action Recognition in Videos”. In: *Advances in Neural Information Processing Systems*. 00353. 2014, pp. 568–576. URL: http://papers.nips.cc/paper/5353-two-stream-convolutional-networks-for-action-recognition-in-videos (visited on 05/06/2016).

[16] Ronan Collobert and Jason Weston. “A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning”. In: *Proceedings of the 25th International Conference on Machine Learning*. 01219. ACM, 2008, pp. 160–167. URL: http://dl.acm.org/citation.cfm?id=1390177 (visited on 10/21/2016).