

---

# Two-Stream Convolutional Networks for Action Recognition in Videos

---

**Karen Simonyan**                      **Andrew Zisserman**  
Visual Geometry Group, University of Oxford  
{karen, az}@robots.ox.ac.uk

## Abstract

We investigate architectures of discriminatively trained deep Convolutional Networks (ConvNets) for action recognition in video. The challenge is to capture the complementary information on appearance from still frames and motion between frames. We also aim to incorporate into the network design aspects of the best performing hand-crafted features.

Our contribution is three-fold. First, we propose a two-stream ConvNet architecture which incorporates spatial and temporal networks. Second, we demonstrate that a ConvNet trained on multi-frame dense optical flow is able to achieve very good performance in spite of limited training data. Finally, we show that multi-task learning, applied to two different action classification datasets, can be used to increase the amount of training data and improve the performance on both. Our architecture is trained and evaluated on the standard video actions benchmarks of UCF-101 and HMDB-51, where it matches the state of the art. It also exceeds by a large margin previous attempts to use deep nets for video classification.

## 1 Introduction

Recognition of human actions in videos is a challenging task which has received a significant amount of attention in the research community [10, 13, 16, 22]. Compared to still image classification, video analysis is more computationally demanding, even in the case of short clips, which still contain hundreds of frames. At the same time, the temporal component of videos provides an additional (and important) clue for recognition, as a number of actions can be reliably recognised based on the motion information. Additionally, video provides natural data augmentation (jittering) for single image (video frame) classification.

In this work, we aim at applying deep Convolutional Networks (ConvNets) [17], a state-of-the-art still image representation [3, 14], to video data. This task has been recently addressed in [13], but their network was performing significantly worse than the best hand-crafted shallow representations [22, 23]. In this work, we develop a different architecture, combining a ConvNet trained for still image classification with a ConvNet trained on the dense video optical flow. The two ConvNets form two separate recognition streams (spatial and temporal), which are then combined by late fusion. We note the relation of our architecture to the two-streams hypothesis [8], according to which the human visual cortex contains two pathways: the ventral stream (which performs object recognition) and the dorsal stream (which recognises motion). Decoupling the spatial and temporal nets also allows us to exploit the availability of large amounts of annotated image data by pre-training the spatial net on the ImageNet challenge dataset [1]. Our experiments on two challenging datasets (UCF-101 [20] and HMDB-51 [15]) show that the two video recognition streams are complementary, and our architecture significantly outperforms that of [13] and matches the state of the art [18, 22, 23], in spite of being trained on a relatively small (for a large-capacity ConvNet) dataset.

The rest of the paper is organised as follows. In Sect. 1.1 we review the related work on action recognition using both shallow and deep architectures. In Sect. 2 we introduce the two-stream architecture and specify the Spatial ConvNet. Sect. 3 introduces the Temporal ConvNet and in

particular how it generalizes the previous architectures reviewed in Sect. 1.1. A multi-task learning ConvNet architecture is developed in Sect. 4 in order to allow effortless combination of training data over multiple datasets. Implementation details are given in Sect. 5, and the performance evaluated in Sect. 6 and compared to the state of the art.

### 1.1 Related work

Video recognition research has been largely driven by the advances in image recognition methods, which were often adapted and extended to deal with video data. A large family of video action recognition methods is based on shallow high-dimensional encodings of local spatio-temporal features. For instance, the algorithm of [16] consists in detecting sparse spatio-temporal interest points, which are then described using local spatio-temporal features: Histogram of Oriented Gradients (HOG) [5] and Histogram of Optical Flow (HOF). The features are then encoded into the Bag Of Features (BoF) representation, which is pooled over several spatio-temporal grids (similarly to spatial pyramid pooling) and combined with an SVM classifier. In a later work [24], it was shown that dense sampling of local features outperforms sparse interest points.

Instead of computing local video features over spatio-temporal cuboids, state-of-the-art shallow video representations [22] make use of dense point trajectories. The approach, first introduced in [25], consists in adjusting local descriptor support regions, so that they follow dense trajectories, computed using optical flow. The best performance in the trajectory-based pipeline was achieved by the Motion Boundary Histogram (MBH) [6], which is a gradient-based feature, separately computed on the horizontal and vertical components of optical flow. A combination of several features was shown to further boost the accuracy. Recent improvements of trajectory-based shallow representations include compensation of global (camera) motion [9, 15, 22], and the use of the Fisher vector encoding [19] for trajectory features [22].

There has also been a number of attempts to develop a deep architecture for video recognition. In the majority of these works, the input to the network is a stack of consecutive video frames, so the model is expected to implicitly learn correspondences or spatio-temporal motion-dependent features in the first layers, which can be a difficult task. In [10], an HMAX architecture for video recognition was proposed with pre-defined spatio-temporal filters in the first layer. Later, it was combined [15] with a spatial HMAX model, thus forming spatial (ventral-like) and temporal (dorsal-like) recognition streams. Unlike our work, however, the streams were implemented as hand-crafted and rather shallow (3-layer) HMAX models. In [21], a convolutional RBM was used for unsupervised learning of spatio-temporal features, which were then plugged into a ConvNet for action classification. Discriminative end-to-end learning of video ConvNets has been addressed in [11] and, more recently, in [13], who compared several ConvNet architectures for action recognition. Training was carried out on a very large Sports-1M dataset, comprising 1.1M YouTube videos of sports activities. Interestingly, [13] found that a network, operating on individual video frames, performs similarly to the networks, whose input is a stack of frames. This might indicate that the learnt spatio-temporal features do not capture the motion well. The learnt representation, fine-tuned on the UCF-101 dataset, turned out to be 20% less accurate than hand-crafted state-of-the-art trajectory-based representation [22, 23].

Our temporal stream ConvNet operates on multiple-frame dense optical flow, which is typically computed in an energy minimisation framework by solving for a displacement field (typically at multiple image scales). A common method [7], is to use an eight-parameter global motion model, with the energy obtained by approximating local image neighbourhoods with polynomials. Another popular method [2] formulates the energy based on constancy assumptions for intensity and its gradient, as well as smoothness of the displacement field. Recently, [26] proposed an image patch matching scheme, which is reminiscent of deep ConvNets, but does not incorporate learning.

## 2 Two-stream architecture for video recognition

Video can naturally be decomposed into spatial and temporal components. The spatial part, in the form of individual frame appearance, carries information about scenes and objects depicted in the video. The temporal part, in the form of motion across the frames, conveys the movement of the observer (the camera) and the objects. We devise our video recognition architecture accordingly, dividing it into two streams, as shown in Fig. 1. Each stream is implemented using a deep ConvNet, the scores of which are combined by late fusion. In our implementation, fusion is performed by

combining the softmax class posteriors from the two nets, so they can be trained and evaluated separately (the details of ConvNet training and testing are given in Sect. 5).

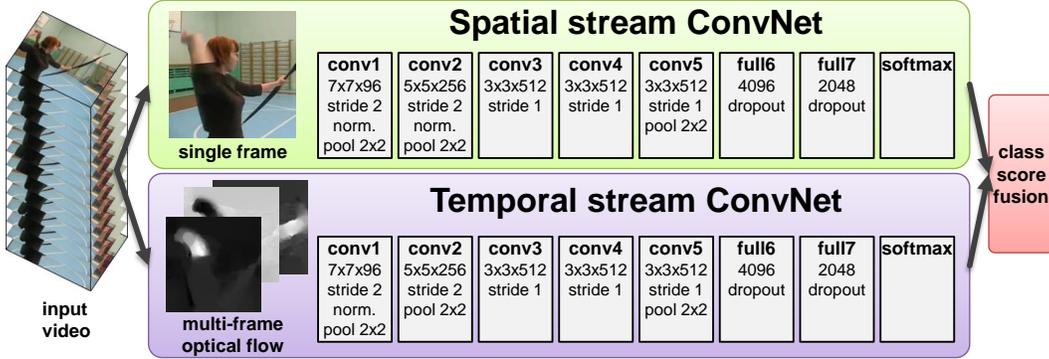


Figure 1: Two-stream architecture for video classification.

## 2.1 Spatial ConvNet

We begin with the description of a spatial stream ConvNet, which operates on individual video frames, effectively performing action recognition from still images. The static appearance by itself is a useful clue, since some actions are strongly associated with particular objects. Since a spatial ConvNet is essentially an image classification architecture, we can build upon the recent advances in large-scale image recognition methods [14], and pre-train the network on a large image classification dataset, such as the ImageNet challenge dataset.

**Architecture.** The ConvNet layer configuration is similar to that of [27], which is the work horse of our recognition framework: with limited changes, it is also used in the temporal ConvNet (Sect. 3). The input to the network is a  $224 \times 224$  RGB image (a crop from a single video frame). It is then passed through five convolutional and two fully-connected layers, interleaved with rectification (RELU) non-linearities, max-pooling, and response normalisation (Fig. 1). The final layer is a softmax classifier. Given a test video, a fixed number of frames are sampled with equal temporal spacing between them, and each class is then scored by averaging its soft-max probabilities over the sampled frames. Full details of training and testing are given in Sect. 5.

As will be shown in Sect. 6, action classification from still frames (the spatial recognition stream) is fairly competitive on its own. Next, we describe the temporal stream ConvNet. This exploits motion and leads to significantly improved accuracy.

## 3 Optical flow ConvNets

In this section, we describe a ConvNet model, which forms the temporal recognition stream of our architecture (Sect. 2). Unlike the ConvNet models, reviewed in Sect. 1.1, the input to our model is formed by stacking optical flow displacement fields between several consecutive frames. Such input explicitly describes the motion between video frames, which makes the recognition easier, as the network does not need to estimate motion implicitly. We consider several variations of the optical flow-based input, which we describe below.

### 3.1 ConvNet input configurations

**Optical flow stacking.** A dense optical flow can be seen as a set of displacement vector fields  $\mathbf{d}_t$  between the pairs of consecutive frames  $t$  and  $t + 1$ . By  $\mathbf{d}_t(u, v)$  we denote the displacement vector at the point  $(u, v)$  in frame  $t$ , which moves the point to the corresponding point in the following frame  $t + 1$ . The horizontal and vertical components of the vector field,  $d_t^x$  and  $d_t^y$ , can be seen as image channels (shown in Fig. 2), well suited to recognition using a convolutional network. To represent the motion across a sequence of frames, we stack the flow channels  $d_t^{x,y}$  of  $L$  consecutive frames to form a total of  $2L$  input channels. More formally, let  $w$  and  $h$  be the width and height of a video; a ConvNet input volume  $I_\tau \in \mathbb{R}^{w \times h \times 2L}$  for an arbitrary frame  $\tau$  is then constructed as

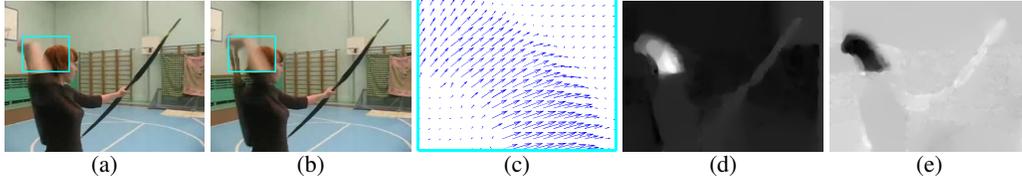


Figure 2: **Optical flow.** (a),(b): a pair of consecutive video frames with the area around a moving hand outlined with a cyan rectangle. (c): a close-up of dense optical flow in the outlined area; (d): horizontal component  $d^x$  of the displacement vector field (higher intensity corresponds to positive values, lower intensity to negative values). (e): vertical component  $d^y$ . Note how (d) and (e) highlight the moving hand and bow. The input to a ConvNet contains multiple flows (Sect. 3.1).

follows:

$$\begin{aligned} I_\tau(u, v, 2k-1) &= d_{\tau+k-1}^x(u, v), \\ I_\tau(u, v, 2k) &= d_{\tau+k-1}^y(u, v), \quad u = [1; w], v = [1; h], k = [1; L]. \end{aligned} \quad (1)$$

For an arbitrary point  $(u, v)$ , the channels  $I_\tau(u, v, c)$ ,  $c = [1; 2L]$  encode the motion at that point over a sequence of  $L$  frames (as illustrated in Fig. 3-left).

**Trajectory stacking.** An alternative motion representation, inspired by the trajectory-based descriptors [25], replaces the optical flow, sampled at the same locations across several frames, with the flow, sampled along the motion trajectories. In this case, the input volume  $I_\tau$ , corresponding to a frame  $\tau$ , takes the following form:

$$\begin{aligned} I_\tau(u, v, 2k-1) &= d_{\tau+k-1}^x(\mathbf{p}_k), \\ I_\tau(u, v, 2k) &= d_{\tau+k-1}^y(\mathbf{p}_k), \quad u = [1; w], v = [1; h], k = [1; L]. \end{aligned} \quad (2)$$

where  $\mathbf{p}_k$  is the  $k$ -th point along the trajectory, which starts at the location  $(u, v)$  in the frame  $\tau$  and is defined by the following recurrence relation:

$$\mathbf{p}_1 = (u, v); \quad \mathbf{p}_k = \mathbf{p}_{k-1} + \mathbf{d}_{\tau+k-2}(\mathbf{p}_{k-1}), \quad k > 1.$$

Compared to the input volume representation (1), where the channels  $I_\tau(u, v, c)$  store the displacement vectors at the locations  $(u, v)$ , the input volume (2) stores the vectors sampled at the locations  $\mathbf{p}_k$  along the trajectory (as illustrated in Fig. 3-right).

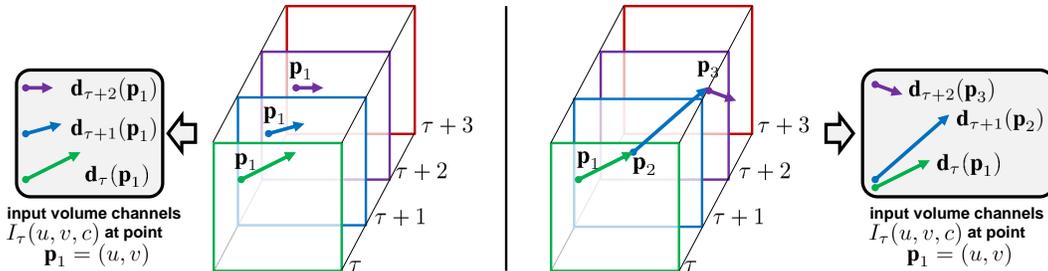


Figure 3: **ConvNet input derivation from the multi-frame optical flow.** *Left:* optical flow stacking (1) samples the displacement vectors  $\mathbf{d}$  at the same location in multiple frames. *Right:* trajectory stacking (2) samples the vectors along the trajectory. The frames and the corresponding displacement vectors are shown with the same colour.

**Bi-directional optical flow.** Optical flow representations (1) and (2) deal with the forward optical flow, i.e. the displacement field  $\mathbf{d}_t$  of the frame  $t$  specifies the location of its pixels in the following frame  $t+1$ . It is natural to consider an extension to a bi-directional optical flow, which can be obtained by computing an additional set of displacement fields in the opposite direction. We then construct an input volume  $I_\tau$  by stacking  $L/2$  forward flows between frames  $\tau$  and  $\tau+L/2$  and  $L/2$  backward flows between frames  $\tau-L/2$  and  $\tau$ . The input  $I_\tau$  thus has the same number of channels ( $2L$ ) as before. The flows can be represented using either of the two methods (1) and (2).

**Mean flow subtraction.** It is generally beneficial to perform zero-centering of the network input, as it allows the model to better exploit the rectification non-linearities. In our case, the displacement

vector field components can take on both positive and negative values, and are naturally centered in the sense that across a large variety of motions, the movement in one direction is as probable as the movement in the opposite one. However, given a pair of frames, the optical flow between them can be dominated by a particular displacement, e.g. caused by the camera movement. The importance of camera motion compensation has been previously highlighted in [9, 22], where a global motion component was estimated and subtracted from the dense flow. In our case, we consider a simpler approach: from each displacement field  $\mathbf{d}$  we subtract its mean vector.

**Architecture.** Above we have described different ways of combining multiple optical flow displacement fields into a single volume  $I_\tau \in \mathbb{R}^{w \times h \times 2L}$ . Considering that a ConvNet requires a fixed-size input, we sample a  $224 \times 224 \times 2L$  sub-volume from  $I_\tau$  and pass it to the net as input. The hidden layers configuration remains largely the same as that used in the spatial net, and is illustrated in Fig. 1. Testing is similar to the spatial ConvNet, and is described in detail in Sect. 5.

### 3.2 Relation of the temporal ConvNet architecture to previous representations

In this section, we put our temporal ConvNet architecture in the context of prior art, drawing connections to the video representations, reviewed in Sect. 1.1. Methods based on feature encodings [16, 25] typically combine several spatio-temporal local features. Such features are computed from the optical flow and are thus generalised by our temporal ConvNet. Indeed, the HOF local descriptor [16] is based on the histograms of optical flow orientations, which can be obtained from the displacement field input (1) using a single convolutional layer (containing orientation-sensitive filters), followed by the rectification and pooling layers. The MBH feature [6] is different in that it computes the histograms of *flow gradient* orientations; this corresponds to adding an additional convolutional layer, making it a two-layer representation which is thus also captured in principle by our temporal ConvNet architecture. The kinematic features of [9] (divergence, curl and shear) are also computed from the optical flow gradient, and, again, can be captured by our convolutional model. Finally, the trajectory feature [25] is computed by stacking the displacement vectors along the trajectory, which corresponds to the trajectory stacking (2). The feature encoding part of the hand-crafted representations is also generalised by a convolutional network, considering that a combination of single convolutional and rectification layers can be seen as a variant of sparse coding (and a network contains several layers of such kind, leading to highly non-linear representations).

As far as the deep networks are concerned, a two-stream video classification architecture of [15] contains two HMAX models which are hand-crafted and less deep than our discriminatively trained ConvNets, which can be seen as a learnable generalisation of HMAX. The convolutional models of [11, 13] do not decouple spatial and temporal recognition streams, and rely on the motion-sensitive convolutional filters, learnt from the data. In our case, motion is explicitly represented using the optical flow displacement field, computed based on the assumptions of constancy of the intensity and smoothness of the flow. Incorporating such assumptions into a ConvNet framework might be able to boost the performance of end-to-end ConvNet-based methods, and is an interesting direction for future research.

### 3.3 Visualisation of learnt convolutional filters

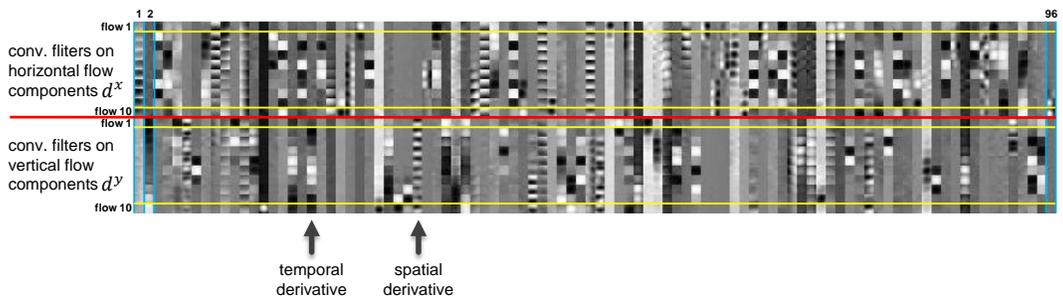


Figure 4: **First-layer convolutional filters learnt on 10 stacked optical flows.** The visualisation is split into 96 columns and 20 rows: each column corresponds to a filter, each row to an input channel.

In Fig. 4 we visualise the convolutional filters from the first layer of the temporal ConvNet, trained on the UCF-101 dataset (training details are given in Sect. 5). Each of the 96 first-layer filters has a spatial receptive field of  $7 \times 7$  pixels, and spans 20 input channels, corresponding to the horizontal

( $d^x$ ) and vertical ( $d^y$ ) components of 10 stacked optical flow displacement fields  $\mathbf{d}$ . As can be seen, some filters compute spatial derivatives of the optical flow, capturing how motion changes with image location. This can be seen as the generalisation of the derivative-based hand-crafted descriptors (e.g. MBH [6]). Other filters compute temporal derivatives, capturing changes in motion over time.

## 4 Multi-task learning

Unlike the spatial stream ConvNet, which can be pre-trained on a large still image classification dataset (such as ImageNet), the temporal ConvNet needs to be trained on video data – and the available datasets for video action classification are still rather small. In our experiments (Sect. 6), training is performed on the UCF-101 and HMDB-51 datasets, which have only: 9.5K and 3.7K videos respectively. To decrease over-fitting, one could consider combining the two datasets into one; this, however, is not straightforward due to the intersection between the sets of classes. One option (which we evaluate later) is to only add the images from the classes, which do not appear in the original dataset. This, however, requires manual search for such classes and limits the amount of additional training data.

A more principled way of combining several datasets is based on multi-task learning [4]. Its aim is to learn a (video) representation, which is applicable not only to the task in question (such as HMDB-51 classification), but also to other tasks (e.g. UCF-101 classification). Additional tasks act as a regulariser, and allow for the exploitation of additional training data. In our case, a ConvNet architecture is modified so that it has *two* softmax classification layers on top of the last fully-connected layer: one softmax layer computes HMDB-51 classification scores, the other one – the UCF-101 scores. Each of the layers is equipped with its own loss function, which operates only on the videos, coming from the respective dataset. The overall training loss is computed as the sum of the individual tasks’ losses, and the network weight derivatives can be found by back-propagation.

## 5 Implementation details

**ConvNets configuration.** The layer configuration of our spatial and temporal ConvNets is schematically shown in Fig. 1. It is similar to that of [27], but we use less filters in the conv4 layer (512 vs. 1024) and lower-dimensional full7 layer (2048-D vs. 4096-D). All hidden weight layers use the rectification (RELU) activation function; max-pooling is performed over  $3 \times 3$  spatial windows with stride 2; local response normalisation across channels uses the same settings as [14]. The only difference between spatial and temporal ConvNet configurations is that we removed the second response normalisation layer from the latter to reduce memory consumption.

**Training.** The training procedure can be seen as an adaptation of that of [14] to video frames, and is generally the same for both spatial and temporal nets. The network weights are learnt using the mini-batch stochastic gradient descent with momentum (set to 0.9). At each iteration, a mini-batch of 256 samples is constructed by sampling 256 training videos (uniformly across the classes), from each of which a single frame is randomly selected. In spatial net training, a  $224 \times 224$  sub-image is randomly cropped from the selected frame; it then undergoes random horizontal flipping and RGB jittering. The videos are rescaled beforehand, so that the smallest side of the frame equals 256. We note that unlike [14], the sub-image is sampled from the whole frame, not just its  $256 \times 256$  center. In the temporal net training, we compute an optical flow volume  $I$  for the selected training frame as described in Sect. 3. From that volume, a fixed-size  $224 \times 224 \times 2L$  input is randomly cropped and flipped. The learning rate is initially set to  $10^{-2}$ , and then decreased according to a fixed schedule, which is kept the same for all training sets. Namely, when training a ConvNet from scratch, the rate is changed to  $10^{-3}$  after 50K iterations, then to  $10^{-4}$  after 70K iterations, and training is stopped after 80K iterations. In the fine-tuning scenario, the rate is changed to  $10^{-3}$  after 14K iterations, and training stopped after 20K iterations.

**Testing.** At test time, given a video, we sample a fixed number of frames (25 in our experiments) with equal temporal spacing between them. From each of the frames we then obtain 10 ConvNet inputs [14] by cropping and flipping four corners and the center of the frame. The class scores for the whole video are then obtained by averaging the scores across the sampled frames and crops therein.

**Pre-training on ImageNet ILSVRC-2012.** When pre-training the spatial ConvNet, we use the same training and test data augmentation as described above (cropping, flipping, RGB jittering). This yields the state-of-the-art 13.5% top-5 error on ILSVRC-2012 validation set, which compares

favourably to 16.1% reported in [27] for a similar network. We believe that the main reason for the improvement is sampling of ConvNet inputs from the whole image, rather than just its center.

**Multi-GPU training.** Our implementation is derived from the publicly available Caffe toolbox [12], but contains a number of significant modifications, including parallel training on multiple GPUs installed in a single system. We exploit the data parallelism, and split each SGD batch across several GPUs. Training a single temporal ConvNet takes 1 day on a system with 4 NVIDIA Titan cards, which constitutes a 3.2 times speed-up over single-GPU training.

**Optical flow** is computed using the off-the-shelf GPU implementation of [2] from the OpenCV toolbox. In spite of the fast computation time (0.06s for a pair of frames), it would still introduce a bottleneck if done on-the-fly, so we pre-computed the flow before training. To avoid storing the displacement fields as floats (which would require 1.5TB of storage for UCF-101), the horizontal and vertical components of the flow were rescaled to a  $[0, 255]$  range and compressed using JPEG. This reduced the total flow size to 27GB, which can be stored on a RAM disk for fast access.

## 6 Evaluation

**Datasets and evaluation protocol.** The evaluation is performed on UCF-101 [20] and HMDB-51 [15] action recognition benchmarks, which are among the largest available annotated video datasets<sup>1</sup>. UCF-101 contains 13K videos (180 frames/video on average), annotated into 101 action classes; HMDB-51 includes 6.8K videos of 51 actions. The evaluation protocol is the same for both datasets: the organisers provide three splits into training and test data, and the performance is measured by the mean classification accuracy across the splits. Each UCF-101 split contains 9.5K training videos; an HMDB-51 split contains 3.7K training videos. We begin by comparing different architectures on the first split of the UCF-101 dataset. For comparison with the state of the art, we follow the recommended evaluation protocol and report the average accuracy over three splits on both UCF-101 and HMDB-51.

**Spatial ConvNets.** First, we measure the performance of the single-frame ConvNet, described in Sect. 2.1. Three scenarios are considered: (i) training from scratch on UCF-101, (ii) pre-training on ILSVRC-2012 followed by fine-tuning on UCF-101, (iii) keeping the pre-trained network fixed and only training the last (classification) layer. For each of the settings, we experiment with setting the dropout regularisation ratio to 0.5 or to 0.9. From the results, presented in Table 1a, it is clear that training the ConvNet solely on the UCF-101 dataset leads to over-fitting (even with high dropout), and is inferior to pre-training on a large ILSVRC-2012 dataset. Interestingly, fine-tuning the whole network gives only marginal improvement over training the last layer only. In the latter setting, higher dropout over-regularises learning and leads to worse accuracy. In the following experiments we opted for training the last layer on top of a pre-trained ConvNet.

Table 1: **Individual ConvNets accuracy on UCF-101 (split 1).**

(a) Spatial ConvNet.			(b) Temporal ConvNet.		
Training setting	Dropout ratio		Input configuration	Mean subtraction	
	0.5	0.9		off	on
From scratch	42.5%	52.3%	Single-frame optical flow ( $L = 1$ )	-	73.9%
Pre-trained + fine-tuning	70.8%	<b>72.8%</b>	Optical flow stacking (1) ( $L = 5$ )	-	80.4%
Pre-trained + last layer	<b>72.7%</b>	59.9%	Optical flow stacking (1) ( $L = 10$ )	79.9%	<b>81.0%</b>
			Trajectory stacking (2)( $L = 10$ )	79.6%	80.2%
			Optical flow stacking (1)( $L = 10$ ), bi-dir.	-	<b>81.2%</b>

**Temporal ConvNets.** Having evaluated spatial ConvNet variants, we now turn to the temporal ConvNet architectures, and assess the effect of the input configurations, described in Sect. 3.1. In particular, we measure the effect of: using multiple ( $L = \{5, 10\}$ ) stacked optical flows; trajectory stacking; mean displacement subtraction; using the bi-directional optical flow. The architectures are trained on the UCF-101 dataset from scratch, so we used an aggressive dropout ratio of 0.9 to help improve generalisation. The results are shown in Table 1b. First, we can conclude that stacking multiple ( $L > 1$ ) displacement fields in the input is highly beneficial, as it provides the network with long-term motion information, which is more discriminative than the flow between a pair of frames

<sup>1</sup>Very recently, [13] released the Sports-1M dataset of 1.1M automatically annotated YouTube sports videos. Processing the dataset of such scale is very challenging, and we plan to address it in future work.

( $L = 1$  setting). Increasing the number of input flows from 5 to 10 leads to a smaller improvement, so we kept  $L$  fixed to 10 in the following experiments. Second, we find that mean subtraction is helpful, as it reduces the effect of global motion between the frames. We use it in the following experiments as default. The difference between different stacking techniques is marginal; it turns out that optical flow stacking performs better than trajectory stacking, and using the bi-directional optical flow is only slightly better than a uni-directional forward flow. Finally, we note that temporal ConvNets significantly outperform the spatial ConvNets (Table 1a), which confirms the importance of motion information for action recognition.

**Multi-task learning of temporal ConvNets.** Training temporal ConvNets on UCF-101 is challenging due to the small size of the data training set. An even bigger challenge is to train the ConvNet on HMDB-51, where each training split is 2.6 times smaller than that of UCF-101. Here we evaluate different options for increasing the effective training set size of HMDB-51: (i) fine-tuning a temporal network pre-trained on UCF-101; (ii) adding 78 classes from UCF-101, which are manually selected so that there is no intersection between these classes and the native HMDB-51 classes; (iii) using the multi-task formulation (Sect. 4) to learn a video representation, shared between the UCF-101 and HMDB-51 classification tasks. The results are reported in Table 2. As expected, it is beneficial to

Table 2: **Temporal ConvNet accuracy on HMDB-51 (split 1 with additional training data).**

Training setting	Accuracy
Training on HMDB-51 without additional data	46.6%
Fine-tuning a ConvNet, pre-trained on UCF-101	49.0%
Training on HMDB-51 with classes added from UCF-101	52.8%
Multi-task learning on HMDB-51 and UCF-101	<b>55.4%</b>

utilise full (all splits combined) UCF-101 data for training (either explicitly by borrowing images, or implicitly by pre-training). Multi-task learning performs the best, as it allows the training procedure to exploit all available training data.

We have also experimented with multi-task learning on the UCF-101 dataset, by training a network to classify both the full HMDB-51 data (all splits combined) and the UCF-101 data (a single split). On the first split of UCF-101, the accuracy was measured to be **81.5%**, which improves on 81.0% achieved using the same settings, but without the additional HMDB classification task (Table 1b).

**Two-stream ConvNets.** Having evaluated spatial and temporal ConvNets, we now turn to the evaluation of the complete two-stream model, which combines the two recognition streams. One way of combining the networks would be to train a joint softmax classification layer on top of the full 7 layers of the two nets. This, however, is not feasible in our case due to severe over-fitting. We therefore resort to combining the softmax scores by simple (weighted) averaging, for which we consider two options: equal weights for both nets ( $1/2, 1/2$ ) and higher weight for the temporal stream ( $2/3$ ). From Table 3 we conclude that: (i) *temporal and spatial recognition streams are complementary, as their fusion significantly improves on both* (5% over temporal and 14% over spatial nets); (ii) using bi-directional flow is not beneficial in the case of ConvNet fusion; (iii) a higher weight for the temporal stream can improve the results, which hints at the possibility of further improvement by training a joint classification layer if sufficient training data is available; (iv) temporal ConvNet, trained using multi-task learning, performs the best both alone and when fused with a spatial net.

Table 3: **Two-stream ConvNet accuracy on UCF-101 (split 1).**

Spatial ConvNet	Temporal ConvNet	$1/2, 1/2$ fusion	$1/3, 2/3$ fusion
Pre-trained + last layer	bi-directional	85.6%	85.8%
Pre-trained + last layer	uni-directional	85.9%	86.2%
Pre-trained + last layer	uni-directional, multi-task	<b>86.2%</b>	<b>86.4%</b>

**Comparison with the state of the art.** We conclude the experimental evaluation with the comparison against the state of the art on three splits of UCF-101 and HMDB-51. For that we used a spatial net, pre-trained on ILSVRC, with the last layer trained on UCF or HMDB. The temporal net was trained on UCF and HMDB using multi-task learning, and the input was computed using uni-directional optical flow stacking with mean subtraction. The softmax scores of the two nets were combined using weighted averaging. As can be seen from Table 4, both our spatial and temporal

nets alone outperform the deep architectures of [13, 15] by a large margin. The combination of the two nets further improves the results (in line with the single-split experiments above), matching the state-of-the-art hand-crafted models [18, 22].

Table 4: **Mean accuracy (over three splits) on UCF-101 and HMDB-51.**

Method	UCF-101	HMDB-51
Improved dense trajectories [22, 23]	85.9%	57.2%
Improved dense trajectories, higher-dimensional encodings [18]	<b>87.9%</b>	<b>61.1%</b>
Spatio-temporal HMAX network [10, 15]	-	22.8%
“Slow fusion” spatio-temporal ConvNet [13]	65.4%	-
Spatial stream ConvNet	72.6%	39.0%
Temporal stream ConvNet	83.6%	54.7%
Two-stream model ( $1/2$ spatial weight, $1/2$ temporal weight)	<b>86.9%</b>	<b>56.3%</b>
Two-stream model ( $1/3$ spatial weight, $2/3$ temporal weight)	<b>87.6%</b>	<b>57.9%</b>

**Confusion matrix for UCF-101 classification.** In Fig. 5 we show the confusion matrix for UCF-101 classification using our two-stream model. We note that the following 23 classes out of 101 are recognised with 100% accuracy: *BabyCrawling, BalanceBeam, BasketballDunk, BenchPress, Biking, Billiards, BreastStroke, CleanAndJerk, CuttingInKitchen, Diving, Drumming, Fencing, HorseRace, HulaHoop, JumpingJack, Knitting, PlayingPiano, PlayingTabla, PommelHorse, SalsaSpin, Skijet, SumoWrestling, TableTennisShot*.

The worst class accuracy corresponds to *Hammering* class, which is confused with *HeadMassage* and *BrushingTeeth* classes. We found that this is due to two reasons. First, the spatial ConvNet confuses *Hammering* with *HeadMassage*, which can be caused by the significant presence of human faces in both classes. Second, the temporal ConvNet confuses *Hammering* with *BrushingTeeth*, as both actions contain similar motion patterns (hand moving up and down).

## 7 Conclusions and directions for improvement

We proposed a deep video classification model with the state-of-the-art performance, which incorporates spatial and temporal recognition streams, based on ConvNets. Currently it appears that training a temporal ConvNet on optical flow (as here) is significantly better than training on raw stacked frames [13]. The latter is probably too challenging, and might require architectural changes (for example, a combination with the deep matching approach of [26]). Despite using optical flow as input, our temporal model does not require significant hand-crafting, since the flow is computed using a method based on the generic assumptions of constancy and smoothness.

As we have shown, extra training data is beneficial for our temporal ConvNet, so we are planning to train it on large video datasets, such as the recently released collection of [13]. This, however, poses a significant challenge on its own due to the gigantic amount of training data (multiple TBs).

There still remain some essential ingredients of the state-of-the-art shallow representation [22], which are missed in our current architecture. The most prominent one is local feature pooling over spatio-temporal tubes, centered at the trajectories. Even though the input (2) captures the optical flow along the trajectories, the spatial pooling in our network does not take the trajectories into account. Another potential area of improvement is explicit handling of camera motion, which in our case is compensated by mean displacement subtraction.

## Acknowledgements

This work was supported by ERC grant VisRec no. 228180. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research.

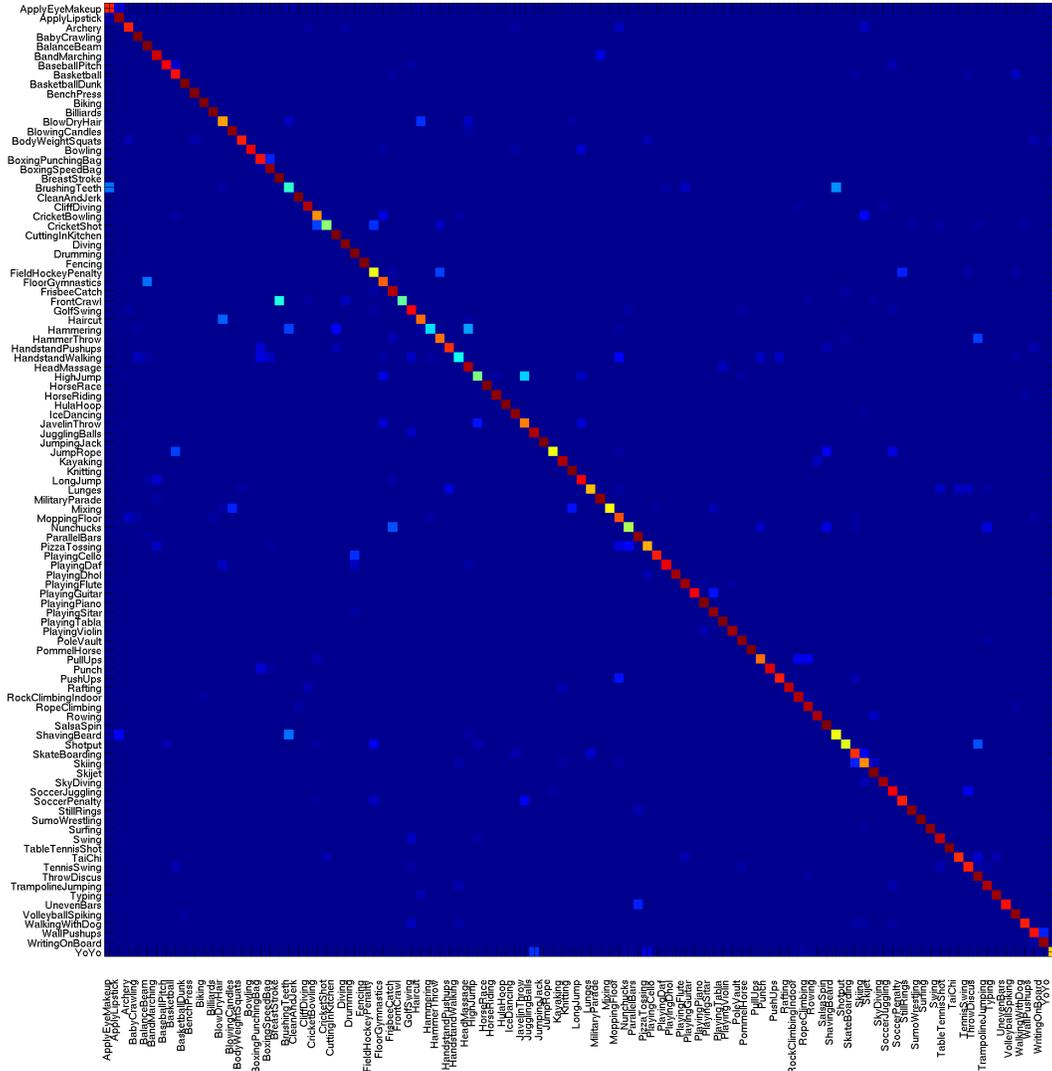


Figure 5: Confusion matrix of a two-stream model on the first split of UCF-101. The corresponding classification accuracy is 86.2% (the last row of Table 3).

## References

- [1] A. Berg, J. Deng, and L. Fei-Fei. Large scale visual recognition challenge (ILSVRC), 2010. URL <http://www.image-net.org/challenges/LSVRC/2010/>.
- [2] T. Brox, A. Bruhn, N. Papenber, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proc. ECCV*, pages 25–36, 2004.
- [3] D. C. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Proc. CVPR*, pages 3642–3649, 2012.
- [4] R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proc. ICML*, pages 160–167, 2008.
- [5] N. Dalal and B. Triggs. Histogram of Oriented Gradients for Human Detection. In *Proc. CVPR*, volume 2, pages 886–893, 2005.
- [6] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *Proc. ECCV*, pages 428–441, 2006.
- [7] G. Farneback. Two-frame motion estimation based on polynomial expansion. In *SCIA*, pages 363–370, 2003.
- [8] M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. *Trends in Neurosciences*, 15(1):20–25, 1992.

- [9] M. Jain, H. Jegou, and P. Bouthemy. Better exploiting motion for better action recognition. In *Proc. CVPR*, pages 2555–2562, 2013.
- [10] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *Proc. ICCV*, pages 1–8, 2007.
- [11] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. *IEEE PAMI*, 35(1):221–231, 2013.
- [12] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. <http://caffe.berkeleyvision.org/>, 2013.
- [13] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proc. CVPR*, 2014.
- [14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.
- [15] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A large video database for human motion recognition. In *Proc. ICCV*, pages 2556–2563, 2011.
- [16] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proc. CVPR*, 2008.
- [17] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.
- [18] X. Peng, L. Wang, X. Wang, and Y. Qiao. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *CoRR*, abs/1405.4506, 2014.
- [19] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *Proc. ECCV*, 2010.
- [20] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.
- [21] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *Proc. ECCV*, pages 140–153, 2010.
- [22] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proc. ICCV*, pages 3551–3558, 2013.
- [23] H. Wang and C. Schmid. LEAR-INRIA submission for the THUMOS workshop. In *ICCV Workshop on Action Recognition with a Large Number of Classes*, 2013.
- [24] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *Proc. BMVC.*, pages 1–11, 2009.
- [25] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Proc. CVPR*, pages 3169–3176, 2011.
- [26] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *Proc. ICCV*, pages 1385–1392, 2013.
- [27] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.