Interaction-Aware Spatio-Temporal Pyramid Attention Networks for Action Classification

Weiming Hu, Haowei Liu, Yang Du, Chunfeng Yuan, and Bing Li
(National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190)
{wmhu, haowei.liu, yang.du, cfyuan, bli}@nlpr.ia.ac.cn

Stephen Maybank
(Department of Computer Science and Information Systems, Birkbeck College, Malet Street, London WC1E 7HX)
sjmaybank@dcs.bbk.ac.uk

Abstract: For CNN-based visual action recognition, the accuracy may be increased if local key action regions are focused on. The task of self-attention is to focus on key features and ignore irrelevant information. So, self-attention is useful for action recognition. However, the current self-attention methods usually ignore correlations among local feature vectors at spatial positions in feature maps in CNNs. In this paper, we propose an effective interaction-aware self-attention model which can extract information about the interactions between feature vectors to learn attention maps. Since the different layers in a network capture feature maps at different scales, we introduce a spatial pyramid with the feature maps at different layers to attention modeling. The multi-scale information is utilized to obtain more accurate attention scores. These attention scores are used to weight the local feature vectors of the feature maps and then calculate the attention feature maps. Since the number of feature maps input to the spatial pyramid attention layer is unrestricted, we easily extend this attention layer to a spatio-temporal version. Our model can be embedded into any general CNN to form a video-level end-to-end attention network for action recognition. Besides using the RGB stream alone, several methods are investigated to combine the RGB and flow streams for the final prediction of the classes of human actions. Experimental results show that our method achieves state-of-the-art results on the datasets UCF101, HMDB51, Kinetics-400, and untrimmed Charades.
Index terms: Action recognition, Attention networks, Interaction-aware, Spatio-temporal pyramid.

1. Introduction

Recognition of human actions in videos occupies a significant position in computer vision [1, 2, 3, 4, 5, 6, 54, 55, 61, 76, 77, 78, 79, 80, 82]. It has a very wide arrange of applications, such as intelligent video surveillance, patient or the-aged monitoring, human-computer interaction, automatic drive, intelligent robots, virtual reality, smart home, intelligent security, athletes' auxiliary training, content-based video retrieval, and smart video compression. It has attracted a large amount of attention. The research on action recognition has developed from regular simple actions under controlled scenarios to complex actions in realistic scenes. The methods for action recognition have developed from the local feature-based to attribute-based and then to deep learning-based. The local feature-based methods extract points of interest and use local features to describe the 3D cubes of the points. The local features are encoded into vectors which are utilized to train a classifier for action recognition. The local feature-based methods are not dependent on human body detection. They perform well on simple datasets with actions under control. As the datasets become more complicated and larger, the accuracy of these methods decreases. The attribute-based methods utilize a series of semantic action attributes to describe human actions in videos. Action attribute graphs are constructed to map human actions to an attribute space in which action classification is carried out. The attribute-based methods decompose a human body into multiple components which are discriminant for actions. These methods conform to human perception. They are relatively robust to variations in illumination and occlusion. After the great progress of the CNN (convolutional neural networks) [7, 8, 9, 10] in image classification, deep learning-based methods has become popular for action recognition. As there are more labeled images to train the networks than labeled video data, many methods apply image-based classifiers to frames in videos in order to recognize actions. However, videos contain much information irrelevant for action recognition. The action recognition accuracy may be increased for the end-to-end deep learning-based methods if local key action regions or points are focused on.

The attention mechanism [11, 12] discards irrelevant information so as to focus on the key information in images. For action recognition, the attention mechanism combining with LSTM (long short-term memory) [13, 14] has been utilized to model the channel-level or frame-level local features extracted by CNN or RNN (recurrent neural networks). Besides the feature maps from CNN, the weights for convolutional features of different regions obtained from the previous frame are used as the input for the LSTM to predict the attention scores for the current frame. The computation of the weights may involve optical flow estimation [14]. This incurs a substantial computational cost. Although LSTM specializes in sequential modeling, it does not achieve high classification accuracy because of the similarity between consecutive frames. Self-attention [12] is one kind of the attention mechanism and also a special form of the non-local networks [15]. Self-attention uses attention scores to weight all feature vectors in order to identify salient feature vectors. Self-attention has lower computational cost compared with other attention mechanisms. The local feature vectors at neighboring spatial positions in feature maps of CNN have high correlations since their receptive fields highly overlap. However, the inherent interactions between feature vectors are ignored in self-attention because each attention score of a feature vector is calculated by the weighted sum (or other functions) of internal elements of the feature vector.

In order to include the correlations between feature vectors in self-attention, we propose an interaction-aware spatio-temporal pyramid attention layer [50]. It can be embedded into general CNNs to generate attention networks which are more discriminative for video action classification. The main components and contributions of our work are outlined as follows:

- **An interaction-aware self-attention layer inspired by PCA:** The attention mechanism enables extraction of key feature vectors with high attention scores, while PCA extracts key feature vectors with principal components.

By minimizing the trace of the covariance matrix, PCA utilizes the interaction information among feature vectors to remove the correlations between feature vectors and obtain basis projection vectors of principal components. We incorporate the idea of PCA into self-attention and propose to use interaction information between feature vectors to obtain their attention scores.

- **A spatial feature pyramid for obtaining more accurate attention maps using multi-scale information:** The feature pyramid [16, 17] stacks the feature maps from different layers and provides multi-resolution feature representation. We introduce the feature pyramid into our attention layer. The interaction-aware self-attention layer models the channel-level features in the pyramid and yield more accurate attention scores. The attention scores are used to aggregate the feature maps of the top layer of the pyramid to obtain more discriminative attentional feature maps.
- **A spatio-temporal version of the interaction-aware self-attention model allowing for any number of frames:** Spatio-temporal detection-based methods [18, 19] have good performance for action recognition. This indicates that attention is useful for detecting salient spatio-temporal regions in videos. We extend the spatial pyramid attention model to a spatio-temporal version to detect and utilize the key information in videos, with the determined architecture and parameters of the attention layer. Our model is independent of the temporal sequence order of the video frames and is compatible with any number of frames.

Besides using single RGB stream CNN, multiple stream CNNs are utilized to combine the information about appearance and motion for action recognition. We embed our attention network layer into four baseline networks, VGGNet-16, BN-Inception, Inception-ResNet-V2, and ResNet-50. Its effectiveness is validated on the UCF101, HMDB51, Kinetics-400, and untrimmed Charades datasets. State-of-the-art results are obtained.

The rest of the paper is organized as follows: Section 2 reviews the related work. Section 3 proposes our interaction-aware spatio-temporal pyramid attention layer. Section 4 presents our action recognition method based on fusion of appearance and motion information. Section 5 reports the experimental results. Section 6 concludes the paper.

2. Related Work

We briefly review deep neural networks and attention methods for action recognition, in order to provide the context for our work.

2.1. Deep neural networks

CNN-based methods have made great progress in image recognition compared with methods based on hand-crafted features [20, 21, 22, 23, 24, 25]. Two-Stream ConvNet [26] uses image-based methods to represent videos from RGB and optical flow streams. The weighted average of the classification scores of the two streams is used for the action prediction. Some researchers [30, 31, 32] directly extend the 2D ConvNet to the 3D ConvNet which is trained end-to-end using entire videos. The extension requires abundant computations and pre-training on a larger dataset such as Kinetics [32]. One of the representative action recognition frameworks uses the recurrent neural network (RNN) [35] or its variants such as LSTM [36, 37, 38, 39] to model the temporal structure of the deep action features extracted from each frame. For instance, CNNs are used to extract low-level visual features and then LSTM is used to carry out high level modeling of the low-level features. Temporal segment networks [28] divide a video into multiple segments. For each segment, two-stream ConvNets are used separately to model the temporal sequence. Deep ConvNets [27] fuse features from different layers. Training on a large-scale dataset, such as Sports-1M, is required. The ST-ResNet [29] uses ResNet [7] as the base of the network structure to carry out the fusion of the two-streams for action recognition. Wang et al. [19] propose trajectory-pooled deep-convolutional descriptors for action recognition. The networks in [33, 34] explore video representations based on spatio-temporal convolutions. Feichtenhofer et al. [64] present an excellent very lightweight SlowFast network which includes a slow pathway and a fast pathway for video recognition. The SlowFast network is appropriate for both fast and slow actions. However, the slow-fast network is computationally expensive to deal with the dynamics and the temporal scale of actions at the input frame level [65, 66]. Zhang et al. [67] propose an efficient temporal reasoning graph for action recognition by simultaneously capturing the appearance features and temporal relations between video sequences at multiple time scales. The temporal reasoning graph extracts discriminative features for action recognition. The lack of fusion of low-level and high-level semantics may be detrimental for reducing the video semantic gap. Ji et al. [68] propose a representation that decomposes actions into spatio-temporal scene graphs. They only utilize homogeneous graphs as scene graphs. Heterogeneous graphs are not the constructed. Ghadiyaram et al. [69] use a large volume of web videos for pre-training models for action recognition. This weakly-supervised method substantially improves the state-of-the-art on some challenging public action recognition datasets. Li et al. [70] propose a weakly-supervised method based on multi-instance multi-label learning for multi-person action recognition in 360° videos. Multiple actions in a video are recognized and localized using only video-level action labels as supervision. Wang et al. [71] propose an excellent weakly supervised architecture to directly learn action recognition models from untrimmed videos without the requirement of temporal annotations of action instances. The merit of the weakly supervised methods is the avoidance of heavy reliance on a large-scale of labeled trimmed videos which are expensive and time-consuming to acquire. The weakly supervised methods are usually utilized to localize actions in videos. Zhuang et al. [72] propose a video instance embedding framework, which trains deep nonlinear embedding on video sequence inputs. The deep neural embedding is promising for unsupervised video learning for action recognition. However, the obtained recognition accuracy is not high. Some action recognition methods focus on detecting key elements related to object classes, global and local motion, as well as global context etc. Wang and Gupta [73] represent videos as spatio-temporal graphs whose nodes are detected object regions from different frames. Action reasoning is carried out via graph convolutional networks. Baradel et al. [74] reason about semantically meaningful spatio-temporal interactions using object detection networks. Huang et al. [75] propose dynamic graph modules for modeling object-object interactions for action recognition. The limitation of the detection-based methods is their dependence on object detection which requires prior knowledge. If object detection is inaccurate, the accuracy of action recognition may substantially decrease. Zhou et al. [81] propose an effective and interpretable network module, the temporal relation network (TRN), to learn and reason about temporal dependencies between video frames at multiple time scales. The network is carried out on activity recognition tasks. Most of the above methods mainly treat equally the information from different frames or spatio-temporal regions. The performance is limited because it is hard to differentiate key features.

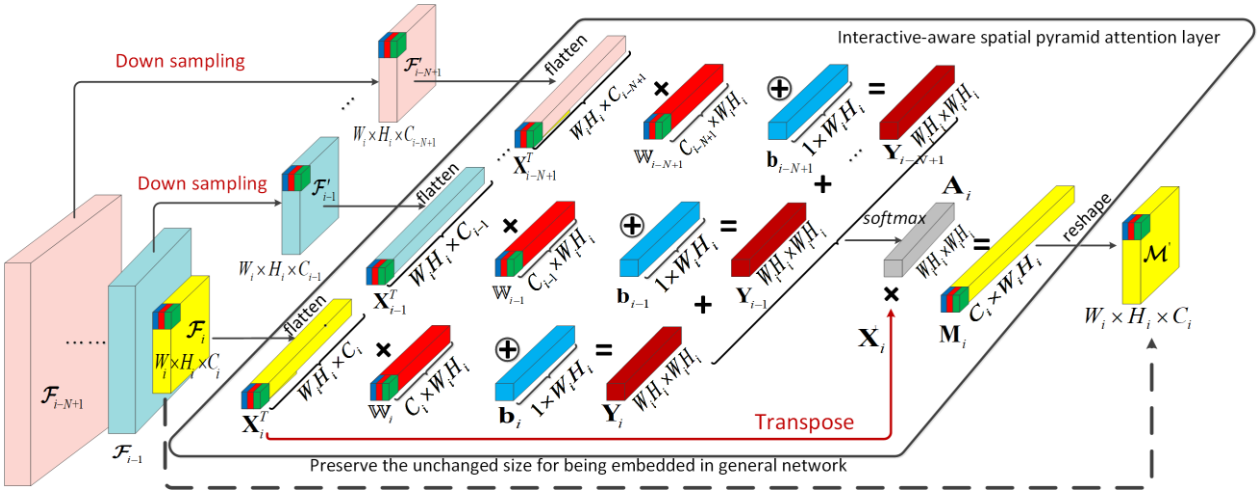


Fig. 1. Our interaction-aware spatial pyramid attention layer: The feature maps of different sizes in different layers are used to construct a multi-scale attention layer to obtain more accurate spatial attention.

2.2. Attention methods

Attention methods are divided into hard and soft methods. Hard attention methods make hard binary choices. Training hard methods is difficult. It is often necessary to add extra supervised information about attention regions to enhance the original model. Mnih et al. [41] and Ba et al. [42] make hard binary choices to select regions using attention RNNs for objection recognition. Gkioxari et al. [43] use an auxiliary box to encode context in addition to the human bounding box. The action-specific models and the feature maps are trained jointly. Mallya and Lazebnik [44] use the entire image as the context and use multiple instance learning to detect all humans in the image and then predict an action label for the image. Soft attention uses weighted averages instead of hard selection. Li et al. [14] propose an end-to-end sequence learning model called VideoLSTM for action recognition. Sharma et al. [13] propose a soft-attention LSTM on top of the RNNs to pay attention to salient parts of the video frames for action classification. However, these soft attention models require auxiliary information from other frames to guide the weighted averages for the current frame. Girdhar and Ramanan [46] propose an unconstrained self-attention pooling operation added at the last layer of a CNN to generate a global representation of the CNN. Long et al. [47] propose a method based on attention clusters to integrate local features of each frame by self-attention. Ma et al. [48] propose a model based on key image-level representations to summarize the entire video sequence using attention LSTM. Previous attention-based methods often focus on the frame-level deep network. This reduces the average performance. It is required to model the interactions between channel vectors of feature maps for constructing a self-attention mechanism and extending the attention model to video-level-based action classification.

3. Interaction-Aware Spatio-temporal Pyramid Attention

We propose an interaction-aware spatial pyramid attention layer inspired by PCA. This layer can be embedded into a general CNN to form an end-to-end attention network and generate discriminative attention feature maps. A new loss function is given for our network. The embedded layer is extended to a temporal version for aggregating temporal sequences for action classification.

3.1. Spatial pyramid attention layer

The CNN extracts feature maps by passing convolutional kernels over the image and treating every local region equally. An attention layer is added into the general CNN after one convolutional layer to emphasize the features of the key local regions and further improve the performance of the network. Let $\mathcal{F}_i \in \mathbb{R}^{W_i \times H_i \times C_i}$ denote a group of feature maps at the i-th layer of a network when a frame is input, where $W_i \times H_i$ is the spatial size and C_i is the number of the channels in the feature maps. We flatten $\mathcal{F}_i$ into a matrix $\mathbf{X}_i = [\mathbf{x}_1^i, \mathbf{x}_2^i, ..., \mathbf{x}_{W_i H_i}^i] \in \mathbb{R}^{C_i \times W_i H_i}$, where the column vector $\mathbf{x}_k^i \in \mathbb{R}^{C_i \times 1}$ (k=1, 2, …, $W_i H_i$) is the channel vector for the k-th spatial position of $\mathcal{F}_i$. The vector $\mathbf{x}_k^i$ represents the local feature vector of its receptive field in the input image. We introduce the feature pyramid into attention modeling. An interaction-aware spatial pyramid attention layer is embedded after the i-th layer and before the i+1th layer of the CNN. This pyramid attention layer generates $W_i H_i$ discriminative feature vectors $\mathbf{M}_i \in \mathbb{R}^{C_i \times W_i H_i}$ based on local feature vectors $\{\mathbf{x}_k^i\}_{k=1}^{w_i h_i}$. We then reshape $\mathbf{M}_i$ into a set of attentional feature maps $\mathcal{M}_i' \in \mathbb{R}^{W_i \times H_i \times C_i}$. In this way, the architecture of the CNN behind the i-th layer is preserved unchanged.

As shown in Fig. 1, we use feature maps of a pyramid of N layers including the i-th layer (the top layer) and the N-1 layers before the i-th layer in the CNN to construct a feature pyramid $\{\mathcal{F}_j \in \mathbb{R}^{W_j \times H_j \times C_j}\}_{j=i-N+1}^{i}$. We down-sample the feature maps in the N-1 layers before the top layer to fit the spatial size of the top layer:

$$\mathcal{F}_j' = \begin{cases} \mathcal{R}(\mathcal{F}_j), & j = i-N+1, ..., i-1, \\ \mathcal{F}_i, & j = i, \end{cases} \quad (1)$$

where $\mathcal{R}(\bullet)$ is a down-sampling function. **A** self-attention estimation is carried out on the channel vectors $\{\mathbf{x}^j\}$. In this way, different numbers of channels in multiple feature maps $\{\mathcal{F}_j'\}_{j=i-N+1}^{i}$ are adapted to. We flatten $\mathcal{F}_j'$ into matrix $\mathbf{X}_j \in \mathbb{R}^{C_j \times W_i H_i}$. Each spatial position m in the feature maps of the j-th layer has a trainable weigh vector $\mathbf{w}_m^j \in \mathbb{R}^{C_j \times 1}$ and a trainable bias value b_m^j. Let $\mathbb{W}_j \in \mathbb{R}^{C_j \times W_i H_i} = \{\mathbf{w}_m^j\}_{m=1}^{W_i H_i}$ denote the trainable weight matrix of the j-th layer. Let $\mathbf{b}_j \in \mathbb{R}^{1 \times W_i H_i} = \{b_m^j\}_{m=1}^{W_i H_i}$ denote the trainable bias row vector. Let $\mathbf{y}_m^j \in \mathbb{R}^{W_i H_i \times 1}$ denote the attention score vector of spatial position m in the j-th layer's feature maps. Let $\mathbf{Y}_j = [\mathbf{y}_1^j, \mathbf{y}_2^j ..., \mathbf{y}_{W_i H_i}^j] \in \mathbb{R}^{W_i H_i \times W_i H_i}$ denote the attention score

matrix for all the spatial positions in the *j*-th layer. Each attention score in $\mathbf{Y}_j$ is obtained by individually computing the sum of the weighted elements of the corresponding channel vector in $\mathbf{X}_j$:

$$\mathbf{Y}_j = \mathbf{X}_j^T \mathbb{W}_j \oplus \mathbf{b}_j, \quad (2)$$

where the symbol $\oplus$ denotes that each row of $\mathbf{X}_j^T \mathbb{W}_j \in \mathbb{R}^{W_i H_i \times W_i H_i}$ adds $\mathbf{b}_j$. The *k*-th score in the column attention vector $\mathbf{y}_m^j$ for the *m*-th spatial position is individually calculated by $\mathbf{x}_k^{jT} \mathbf{w}_m^j + b_m^j$. This means that the element in the score vector $\mathbf{y}_m^j$ for $\mathbf{x}_k^j$ is calculated by just weighting the elements in $\mathbf{x}_k^j$, i.e., independently of all the channel vectors $\{\mathbf{x}_d^j\}_{d \neq k}$ in the *j*-th layer except for $\mathbf{x}_k^j$.

The *N* attention score matrices $\{\mathbf{Y}_j\}_{j=i-N+1}^{i}$ of the *N* layers in the spatial pyramid are fused into a matrix which is then normalized into an attention score matrix $\mathbf{A}_i \in \mathbb{R}^{W_i H_i \times W_i H_i}$ for the entire spatial pyramid attention layer. All the *m*-th column vectors $\{\mathbf{y}_m^j \in \mathbb{R}^{W_i H_i \times 1}\}_{j=i-N+1}^{i}$ in the attention matrices $\{\mathbf{Y}_j\}_{j=i-N+1}^{i}$ for all the layers in the pyramid are used to calculate a vector $\mathbf{a}_m^i \in \mathbb{R}^{W_i H_i \times 1}$ in the following way:

$$\mathbf{a}_m^i = \text{soft}\max(\mathcal{U}(\mathbf{y}_m^{i-N+1}, \ldots, \mathbf{y}_m^{i-1}, \mathbf{y}_m^i)), \quad (3)$$

where $\mathcal{U}$ is a function for fusing column vectors into one vector. We investigate the following three fusion functions: element-wise maximum, element-wise sum, and element-wise multiplication. In (3), $\mathbf{a}_m^i$ is L_1-normalized due to the softmax. We further carry out L_2-normalization on $\mathbf{a}_m^i$: $L_2 - \text{normalization}(\mathbf{a}_m^i) \rightarrow \mathbf{a}_m^i$. Then, $\mathbf{a}_m^i$ is L_2-normalized [49] to preserve $\mathbf{a}_m^{i\,T} \mathbf{a}_m^i = 1$ and a normalized attention score matrix $\mathbf{A}_i = [\mathbf{a}_1^i, \mathbf{a}_2^i, \ldots, \mathbf{a}_{w_i h_i}^i]$ is obtained. It is noted that the normalization in (3) relates feature vectors, where each feature attention competes with others. This kind of comparative interaction simultaneously increases or decreases the components of each attention score vector. It does not directly model the inherent interactions among feature vectors for attention modeling.

The matrix $\mathbf{A}_i \in \mathbb{R}^{W_i H_i \times W_i H_i}$ is used to aggregate the flattened feature maps $\mathbf{X}_i \in \mathbb{R}^{C_i \times W_i H_i}$ of the *i*-th layer to obtain a more discriminative representation $\mathbf{M}_i \in \mathbb{R}^{C_i \times W_i H_i}$ for the *i*-th layer as follows:

$$\mathbf{M}_i = \mathbf{X}_i \mathbf{A}_i. \quad (4)$$

3.2. PCA-inspired interaction-aware modeling

The attention mechanism extracts key feature vectors from the set of all feature vectors by using the attention scores to weight the feature vectors. Each weight score for a local feature vector is obtained by the weighted sum of the internal elements in the local feature vector. The inherent interactions between feature vectors are not directly modeled. PCA extracts key feature vectors by using a set of basis vectors on which feature vectors are projected. The interactions between feature vectors are modeled in PCA. We give a new insight into the self-attention process using a PCA to model the interactions between feature vectors for the self-attention.

We flatten $\mathcal{F}_i$ into $\mathbf{X}_i^T \in \mathbb{R}^{W_i H_i \times C_i}$ which consists of the stacked feature map vectors of the channels: $\mathbf{X}_i^T = [\mathbf{v}_1^i, \mathbf{v}_2^i, \ldots, \mathbf{v}_{C_i}^i]$, where $\mathbf{v}_o^i \in \mathbb{R}^{W_i H_i \times 1}$ is a flattened global feature map of the *o*-th channel ($1 \le o \le C_i$). PCA is used to generate the key feature vectors $\mathbf{M}_i' \in \mathbb{R}^{W_i H_i \times C_i}$ with principal dimensions from the stacked columns $\{\mathbf{v}_o^i\}_{o=1}^{C_i}$. The matrix $\mathbf{M}_i'$ consist of discriminative features for $\mathbf{X}_i^T$. Correspondingly, the matrix $\mathbf{M}_i = (\mathbf{M}_i')^T \in \mathbb{R}^{C_i \times W_i H_i}$ consist of discriminative features for $\mathbf{X}_i$. Let $\mathbf{S} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{W_i H_i}]$ be a set of orthogonal basis vectors where $\mathbf{e}_n \in \mathbb{R}^{W_i H_i \times 1}$ (n=1,2,…, $W_i H_i$). The vectors $\{\mathbf{v}_o^i\}_{o=1}^{C_i}$ are projected onto $\mathbf{S}$ in order to extract principal components. PCA yields $\mathbf{M}_i = \mathbf{X}_i \mathbf{S} \in \mathbb{R}^{C_i \times W_i H_i}$. This is the same as the form of $\mathbf{M}_i$ in (4).

There exists a subtle correspondence between PCA and the attention mechanism. The following points are noted:

- The attention score vector $\mathbf{a} \in \mathbb{R}^{W_i H_i \times 1}$ corresponds to the basis vector $\mathbf{e} \in \mathbb{R}^{W_i H_i \times 1}$ even though the ways for computing **a** and **e** are different.
- An attention process extracts key feature vectors from $\mathbf{X}_i$ by treating $\mathbf{X}_i$ as stacked channel-level feature vectors. PCA extracts principal components from $\mathbf{X}_i$ by treating $\mathbf{X}_i$ as stacked features divided by channels.
- the data central processing "$\oplus \mathbf{b}_j$" in (2) corresponds to the subtraction of the mean in PCA.

The self-attention process doesn't consider the interactions between feature vectors since it weights each feature vector independently to obtain its attention scores. As described in the analysis of (2), the attention scores for a feature vector **x** are obtained by weighting **x** itself. However, PCA usually obtains an orthonormal basis **S** by eigenvalue decomposition of the covariance matrix. In this way it utilizes the non-local interactions among feature vectors. Inspired by PCA, we incorporate the non-local interaction information among channel features $\{\mathbf{x}\}$ into the self-attention processing, in order to generate an interaction-aware spatial pyramid attention layer.

We refer to an equivalent form [51] of PCA for calculating **S**. The project of the *o*-th column vector $\mathbf{v}_o$ on the *n*-th project basis vector $\mathbf{e}_n$ is $\pi_n^o = \mathbf{e}_n^T \mathbf{v}_o$. To maximize the variance of the projected vectors, the first project basis vector $\mathbf{e}_1$ satisfies

$$\begin{aligned} \mathbf{e}_1 &= \arg\max_{\mathbf{e}} \sum_o (\pi_1^o)^2 = \arg\max_{\mathbf{e}} \sum_o (\mathbf{e}^T \mathbf{v}_o)^2 \\ &= \arg\max_{\mathbf{e}} \left\| \mathbf{e}^T \mathbf{X}_i^T \right\|^2 = \arg\max_{\mathbf{e}} \mathbf{e}^T \mathbf{X}_i^T \mathbf{X}_i \mathbf{e}, \end{aligned} \quad (5)$$

where $\|\mathbf{e}\| = 1$. Considering all the basis vectors in **S**, the matrix trace is used to equivalently represent the optimization of **S**:

$$\mathbf{S} \leftarrow \arg\min_{\mathbf{S}} -tr(\mathbf{S}^T \mathbf{X}_i^T \mathbf{X}_i \mathbf{S}), \quad (6)$$

where $\mathbf{S}\mathbf{S}^T = \mathbf{I}$ ($\mathbf{I}$ is an identity matrix). The non-local interactions among feature vectors are modeled in (6). Since there is close correspondence between **S** and $\mathbf{A}_i$, we replace **S** in (6) with $\mathbf{A}_i$ to utilize the non-local interaction information among channel features for the self-attention processing:

$$\mathbf{A}_i \leftarrow \arg\min_{\mathbf{A}_i} -tr(\mathbf{A}_i^T \mathbf{X}_i^T \mathbf{X}_i \mathbf{A}_i), \quad (7)$$

where $\mathbf{A}_i^T \mathbf{A}_i = \mathbf{I}$. We add (7) into the loss function.

3.3. Loss function

The loss function of our network includes the PCA inspired interaction-aware loss, the pyramid attention loss, and the classification loss. The individual losses and the final loss are defined below.

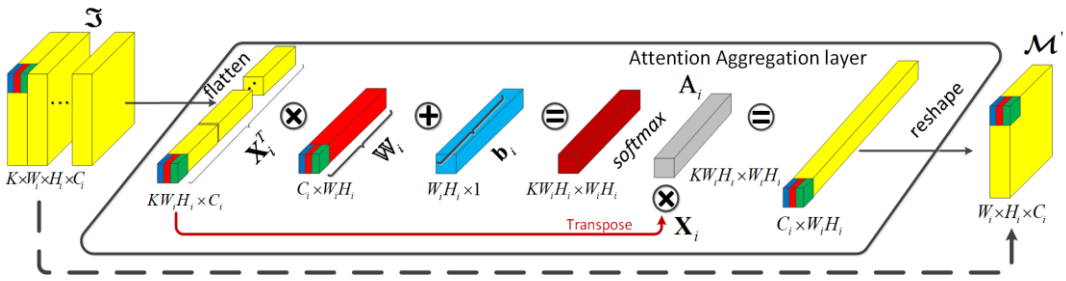


Fig. 2. The process of aggregating K groups of feature maps $\mathfrak{I}_i \in \mathbb{R}^{K\times W_i\times H_i\times C_i}$ into one group of feature maps $\mathcal{M}_i' \in \mathbb{R}^{W_i\times H_i\times C_i}$.

The specific form of our interaction-aware loss inspired by PCA includes a differentiable form into which (7) is changed and the constraint $\mathbf{A}_i^T\mathbf{A}_i = \mathbf{I}$. Let $\mathbf{1} \in \mathbb{R}^{W_iH_i\times W_iH_i}$ be a matrix of all ones. Let $\mathbf{I} \in \mathbb{R}^{W_iH_i\times W_iH_i}$ be the identity matrix. The interaction-aware loss is defined as:

$$\mathcal{L}_{interactive} = -\mathcal{G}((\mathbf{A}_i^T\mathbf{X}_i^T\mathbf{X}_i\mathbf{A}_i)\circ\mathbf{I}) + \mathcal{H}((\mathbf{A}_i^T\mathbf{A}_i)\circ(\mathbf{1}-\mathbf{I})), \quad (8)$$

where $\circ$ is element-wise multiplication, $\mathcal{G}$ is the operation of sum of elements, and $\mathcal{H}$ is the operation of quadratic sum of elements. Because the column vectors in $\mathbf{A}_i$ are L_2-normalized, the diagonal elements of $\mathbf{A}_i^T\mathbf{A}_i$ are all equal to one. This ensures that minimizing the second term in the right-hand of the equality sign in (8) yields the constraint $\mathbf{A}_i^T\mathbf{A}_i = \mathbf{I}$.

The pyramid attention loss is a regularization of the attention scores in order to enhance the attention maps. The set $\{\mathbf{a}_m^i\}$ represents the attention scores calculated from all the scales of the spatio-temporal pyramid and $\{\mathbf{y}_m^j\}$ represents the attention scores obtained by the j-th scale of the pyramid. The softmax is used to normalize the vector $\mathbf{y}_m^j$: $\mathrm{softmax}(\mathbf{y}_m^j)$. To utilize the information at different scales of the pyramid and ensure that the characteristics at each scale is focused on as far as possible, we maximize the distance $\delta_m^j \in [0,1]$ between $\mathbf{a}_m^i$ and $\mathrm{softmax}(\mathbf{y}_m^j)$:

$$\delta_m^j = \| \mathbf{a}_m^i - \mathrm{softmax}(\mathbf{y}_m^j) \|, \quad m = 1, ..., W_iH_i. \quad (9)$$

Maximizing δ_m^j is equivalent to minimizing $1-\delta^2$. Subsequently, we define the specific form of our pyramid attention loss as:

$$\mathcal{L}_{attn} = \sqrt{\sum_j \sum_m (1-(\delta_m^j)^2)}. \quad (10)$$

The classification loss is defined using the cross-entropy loss. Let l_c be a one-hot label of the current sample , i.e., l_c equals 1 if the ground truth label of the sample is c, otherwise l_c equals 0 . Let $\hat{l}_c$ be the probability that the input is predicted to class c. The classification loss is defined as the cross entropy:

$$\mathcal{L}_{class} = -\sum_{c=1}^{\mathcal{C}} l_c \log \hat{l}_c, \quad (11)$$

where C is the number of classes.

The final loss is defined a weighted sum of the classification loss, the interaction-aware loss, the pyramid attention loss, and a regularization of the network parameters. Let $\mathbf{w}_\theta$ be the trainable parameters of the neural network. The final loss is defined as:

$$\mathcal{L}_{final} = -\mathcal{L}_{class} + \beta\mathcal{L}_{interactive} + \gamma\mathcal{L}_{attn} + \lambda\sum_\theta \mathbf{w}_\theta^2, \quad (12)$$

where β, γ, and λ are weight decay coefficients. The ℓ_2-norm of the network parameters is incorporated into the loss function in order to regularize the network for better generalization.

3.4. Temporal aggregation

We extend the above interaction-aware spatial pyramid attention layer for a single image to multiple frames. The extended layer models temporal sequences and detects key spatio-temporal information.

We sample K frames from a video, and then input them into the network. At the i-th layer, K groups of feature maps $\mathfrak{I}_i \in \mathbb{R}^{K\times W_i\times H_i\times C_i}$ are extracted. These feature maps $\mathfrak{I}_i$ are flattened into a matrix $\mathbf{X}_i \in \mathbb{R}^{C_i\times KW_iH_i}$. To preserve unchanged the parameters and architecture of the network after the i-th layer, we aggregate $\mathfrak{I}_i$ into a group of attention feature maps $\mathcal{M}_i' \in \mathbb{R}^{W_i\times H_i\times C_i}$ which have the same size as $\mathcal{F}_i$. A feature pyramid is constructed using $\{\mathfrak{I}_j\}_{j=i-N+1}^{i}$. We still set $\mathbb{W}_j \in \mathbb{R}^{C_j\times W_iH_i}$ and $\mathbf{b}_j \in \mathbb{R}^{1\times W_iH_i}$, i.e., $\mathbb{W}_j$ and $\mathbf{b}_j$ for multiple frames have the same forms as those for a single frame, as in (2). This is because $\mathbb{W}_j$ is used to weight the channels of the feature maps. The size of the output attention maps is fixed to $W_i \times H_i$, which is also the size of the i-th layer in the original CNN. By replacing $\mathcal{F}_i$ with $\mathfrak{I}_i$ and replacing $\mathbf{X}_j \in \mathbb{R}^{C_j\times W_iH_i}$ with $\mathbf{X}_j \in \mathbb{R}^{C_j\times KW_iH_i}$, (2) becomes

$$\mathbf{Y}_j = \{\mathbf{y}_m^j \in \mathbb{R}^{KW_iH_i\times 1}\}_{m=1}^{W_iH_i} = \mathbf{X}_j^T\mathbb{W}_j \oplus \mathbf{b}_j \in \mathbb{R}^{KW_iH_i\times W_iH_i}, \quad (13)$$

where $\mathbf{y}_m^j$ is the attention score vector for the m-th position at the j-th layer. The normalized attention score matrix becomes $\mathbf{A}_i \in \mathbb{R}^{KW_iH_i\times W_iH_i}$ where its m-th column vector $\mathbf{a}_m$ is the normalized attention score vector for the m-position in the feature maps. Then, we obtain the aggregated attention feature matrix by $\mathbf{M}_i = \mathbf{X}_i\mathbf{A}_i \in \mathbb{R}^{C_i\times W_iH_i}$ which is then reshaped to attention feature maps $\mathcal{M}_i' \in \mathbb{R}^{W_i\times H_i\times C_i}$. Fig. 2 shows the pipeline of aggregating K groups of feature maps into one group of feature maps.

We correspondingly modify the loss function. Considering K input frames, the pyramid attention loss becomes

$$\mathcal{L}_{attn} = \sqrt{\sum_j \sum_m \sum_t (1-(\delta_{m,t}^j)^2)}, \quad t = 1, 2, ..., K, m = 1, ..., W_iH_i, \quad (14)$$

where

$$\delta_{m,t}^j = \| \mathbf{a}_{m,t}^i - \mathrm{softmax}(\mathbf{y}^j)_{m,t} \|. \quad (15)$$

The cross-entropy loss becomes

$$\mathcal{L}_{class} = -\frac{1}{K}\sum_{t=1}^{K}\sum_{c=1}^{\mathcal{C}} l_{t,c} \log \hat{l}_{t,c}. \quad (16)$$

Then, we can use sequences with different numbers of frames to train and test the networks. With the defined final loss function, the training samples are used to train the weights $\{\mathbb{W}_j\}$ and the bias vectors $\{\mathbf{b}_j\}$.

4. Action Recognition

We incorporate the proposed interaction-aware spatio-temporal pyramid attention layer into general CNNs to form end-to-end attention networks for action classification. RGB images and optical flow gray images [56] are input to train, respectively, two CNNs with the same structure. The RGB stream extracts spatial appearance information from videos. The optical flow stream extracts temporal motion information from videos. The framework of the proposed attention network-based framework for one stream is shown in Fig. 3. By inserting an interaction-aware spatio-temporal pyramid attention layer into the CNN, K group of feature maps from K frames are aggregated into one group of feature maps. Multi-scale feature maps are utilized to accurately focus on the salient regions. Information fusion of the RGB and flow streams is carried out for the final prediction of the classes of human actions. The following fusion modes are investigated (See the appendix for details):

- late fusion of the scores of the final layers of the RGB and flow streams: combining the classification results of the RGB and flow streams to yield the final classification result [92];
- attention layer-based fusion: using the spatio-temporal interaction-aware attention pyramid layer to combine the information in the middle layers in the RGB and flow streams;
- combination with motion compensation: combining the two-stream neural networks with optical flow compensation which is introduced to handle optical flow produced by camera motion [28];
- combination with motion reinforcement: incorporating motion reinforcement to give attention scores to the RBG branch for action recognition in order to enhance interactions and associations between the RGB branch and the optical flow branch [29, 93];
- combination with iDT (improved dense trajectories): combining our action recognition method with the traditional iDT method [18] to further improve the recognition accuracy [33].

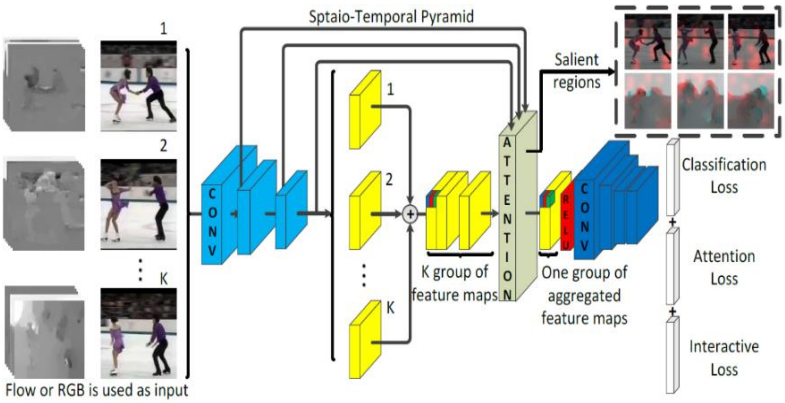


Fig. 3. Our interaction-aware spatio-temporal attention-based framework for one stream for action recognition.

5. Experiments

Our methods were implemented in TensorFlow with TITAN Xp×2 GPUs. We investigated VGGNet-16 [9], BN-Inception [52], Inception-ResNet-V2 [10], and ResNet-50 [7]. We evaluated our models on the following four challenging action classification benchmarks: the UCF101 dataset [6], the HMDB51 dataset [5], the Kinetics-400 dataset [94], and the untrimmed Charades dataset [39]. For the UCF101 and HMDB51 datasets, the original evaluation scheme was followed using three different training and test splits. Split 1 was used for ablation analysis and the final performance was assessed by the average classification accuracy over the three splits. For the Charades dataset, the evaluation pipeline of [39] was followed. For the Kinetics-400 dataset, the training set was used to train our model and the validation set was used to estimate performance of action recognition methods.

Max pooling was used as the down-sampling $\mathcal{R}(\bullet)$ in (1). This is because our experiments show that max pooling is marginally superior to average pooling and down-sampling convolution. In addition, max pooling doesn't have any new parameters. In the RGB stream, the dropout was set to 0.5 which yields good performance. Since RGB images are the inputs, the models trained on the ImageNet [53] were used to initialize the parameters of the networks. In the flow stream, a dropout of 0.7 was used to avoid over-fitting on small flow datasets. The optical flow stream was initialized using the RGB model [28]. The parameters of the first convolutional layer in the RGB pre-training model were modified to adapt the number of input frames in the optical flow model, i.e., an averaging operation was carried out on the RGB channels and the averaged channels were duplicated to obtain the same number of inputs as the optical flow stream. Data augmentation was carried out by random cropping and flipping for both the RGB frames and the flow frames. The mini-batch stochastic gradient descent (SGD) method with momentum of 0.9 was used to optimize the network model. For the weight decay coefficients in (12), λ was set to $4e^{-5}$ and both β and γ were set to $1e^{-4}$. The batch size for network training was set to 64. To fairly compare with the competing methods, the learning rates were set differently for different backbones:

- **VGGNet-16, BN-Inception, and Inception-ResNet-V2:** For the spatial network, the learning rate was initialized as 0.001 and decreased by a factor 1/10 every 4000 iterations. The entire training procedure stopped at 12000 iterations. For the temporal network, the learning rate was initialized as 0.005 and decreased by factor 1/10 after 12000 and 24000 iterations. The maximum number of iterations was set to 30000.
- **ResNet-50:** The learning rate was initialized as 1e-2 and decreased by 1/10 at epoch 40 and 80. A total of 100 epochs of training were used.

In order to fairly compare with the different baselines, the following three sampling strategies in the baselines were utilized:

- The first strategy follows the segment setting in [28, 40] for training the neural networks. That is, a video was divided into 3 equal length segments, and a frame was randomly chosen from each segment. Then, a sequence of three frames (K=3) was formed as the input of the networks. In addition, we also evaluated the performance when K=1 for training. During the test process, we investigated the effects of K on the performance of the RGB and flow streams. We used K=25 frames for test and compared the results with other standard state-of-the-art methods with the same setting. The performance for larger number of frames (>25) was also investigated.
- The second strategy, for training, uniformly divides each video into 8 segments and one frame was sampled randomly from each segment, in the same way as in [45]. For testing, 25 frames were uniformly sampled from each video.
- The third strategy, in the same way as in [32], for training, randomly chooses 64 consecutive frames from each video and 8 frames were uniformly sampled from these 64 frames and used as the input to the neural networks. For testing, each video was uniformly divided into 10 clips. From the first 64 consecutive frames of each clip,

8 frames were uniformly sampled. The final prediction result for a video was determined by the average of the prediction scores for all the 10 clips.

The following evaluations were made:

- evaluation of the proposed interaction-aware attention layer,
- evaluation of the temporal aggregation in the attention layer,
- evaluation of the number of the parameters in the interaction-aware attention layer,
- analysis of the differences between confusion matrices,
- evaluation of the strategies for fusing the RGB stream and the optical flow stream,
- visualization of salient human action regions following the attention feature maps,
- comparison with the-state-of-the-art methods.

5.1. Evaluation of the interaction-aware attention layer

Our interaction-aware spatio-temporal pyramid attention layer was investigated in the following six ways:

- the layer position of feature maps used for aggregation with attention,
- the number of layers in the pyramid,
- different fusion functions $\mathcal{U}$ in (3) for a pyramid,
- effects of the components in the attention model,
- action class analysis for attention-based models,
- the embedding of the proposed interaction-aware attention layer in different popular deep networks, including VGGNet-16 [9], BN-Inception [52], and Inception-ResNet-V2 [10].

5.1.1. Layer position used for attention aggregation

To determine which layer of feature maps is suitable for aggregation with attention weights, the following four different layers were investigated:

- the last layers of Inception-ResNet-A, B, and C without activation (the sizes of these three layers are 35×35×320, 17×17×1088, and 8×8×2080, respectively),
- the last fully-connected layer which is denoted as $\mathbf{X}^{1536\times1}$ ($\mathbb{W}\in\mathbb{R}^{1536\times1}$, $\mathbf{b}\in\mathbb{R}^{1536\times1}$).

The pyramid is 1-scale, i.e., it has only one layer for evaluation. The results are shown in Table 1. It is seen that using the last convolutional layer of Block C as the top layer yields a better performance than using the last fully connected layer and the last layers of Inception-ResNet-A and B as the top layer. The reason is that the fully-connected layer loses much information about the spatial locations. The large spatial sizes of the feature maps in the last layers of Inception-ResNet-A and B cause that the features in these maps are less representative.

Table 1. Evaluation of positions of the interaction-aware attention layer on the UCF101 split 1 dataset: K=3 for training and K=25 for testing

Block (Inception-ResNet-V2)	RGB stream	Flow stream
Block A (35×35×320)	85.8%	83.5%
Block B (17×17×1088)	86.1%	83.7%
Block C (8×8×2080)	**86.3%**	**84.0%**
Fully Connected (1536)	85.5%	83.4%

5.1.2. The number of layers in the pyramid

The performances of our spatio-temporal pyramid attention layer were compared between 1 scale (only using the top layer of the pyramid), 2 scales (using 2 layers), 3 scales, and 4 scales. Table 2 shows the results for which the last convolutional layer of Inception-ResNet-Block C was used as the top layer of the pyramid, together with the results without the attention layer (0 scales). The following points are noted:

- Compared with the basic model without the attention layer, the 1-scale attention layer improves the accuracy by 1.1% and 0.9% on the RGB stream and the flow stream respectively. This indicates that the attention mechanism increases the classification performance of the networks by paying attention to more discriminative image regions and inhibiting the influence of background disturbances to some extent.
- The performance is further promoted when the number of scales is increased to 3. The 3 scales improve the accuracy by 2.1% and 2.4% on the RGB and flow streams respectively. This indicates that the spatio-temporal pyramid attention is effective because more information from multi-scale feature maps of different receptive fields is used for the aggregation of local features.
- When a fourth scale was added by using Conv2d 4a 3x3 (71×71×92) in Inception-ResNet-V2, the performance drops. The reason is that when feature maps with larger sizes are used, the receptive fields of the feature maps become narrow. Too narrow receptive fields increase the effect of noise on the local feature vectors. Hence, the architecture of 3 layers (3 scales) is appropriate for constructing the spatio-temporal pyramid.

Table 2. Evaluation of different numbers of scales with Inception-ResNet-V2 on the UCF101 split 1 dataset: K=3 for training and K=25 for testing

Scale	RGB stream	Flow stream
0 scales	85.2%	83.1%
1 scale	86.3%	84.0%
2 scales	86.8%	84.8%
3 scales	**87.3%**	**85.5%**
4 scales	86.5%	85.0%

5.1.3. Different fusion functions $\mathcal{U}$ for a pyramid

We evaluated different types of the fusion function $\mathcal{U}$ in (3) for feature maps in the pyramid on the RGB stream. Table 3 compares the results of different fusion strategies when the last convolutional layer of Inception-ResNet-Block C was used as the top layer of the pyramid with 3 scales. It is seen that element-wise multiplication performs better than element-wise maximum and element-wise sum. Therefore, the element-wise multiplication was selected as the default fusion function. A similar conclusion was obtained for the TLE networks in [40], showing that element-wise multiplication is more appropriate for fusion encoding of feature maps extracted from different frames.

Table 3. Performance of different fusion functions with 3 scales on the UCF101 split 1 dataset: K=3 for training and K=25 for testing

Fusion function ($\mathcal{U}$)	Accuracy #RGB
Element-wise maximum	85.7%
Element-wise sum	86.4%
Element-wise multiplication	**87.3%**

5.1.4. Effects of components in the attention model

We investigated the effectiveness of the new attention method by fixing the temporal modeling settings and comparing the experimental results with and without the interaction-aware pyramid attention on the action recognition datasets (video-level datasets). Because the temporal modeling settings are the same, the results describe the effect of our interaction-aware pyramid attention on image-level recognition. We also tested the effectiveness of the interaction-aware pyramid attention on the CIFAR 100 image classification dataset (an image-level dataset).

We separately investigated the effect of the proposed PCA-inspired interaction-aware attention and the spatial pyramid attention. The proposed PCA-inspired interaction-aware attention is described by our interaction-aware loss

$\mathcal{L}_{\text{interactive}}$ in (8). The spatial pyramid attention is described by the pyramid attention loss $\mathcal{L}_{\text{attn}}$ in (14). We compared the results of i) using both the interaction-aware attention and the spatial pyramid attention, ii) using either the interaction-aware attention or the spatial pyramid attention alone, and iii) not using either the interaction-aware attention or the spatial pyramid attention. The results are shown in Table 4, for which the last layers of Inception-ResNet-A, B, and C without activation were chosen to construct a 3-scale interaction-aware spatio-temporal pyramid attention layer. The RGB stream, the flow stream, and the late fusion [26, 28, 39, 40] were considered. The results show that, compared with the results without both the interaction-aware attention and the spatial pyramid attention, using both the interaction-aware attention and the spatial pyramid attention improves the performance by 0.9%, 1.0%, and 0.9% on the RGB stream alone, the flow stream alone, and the late fusion of the two streams. Using either the interaction-aware attention or the spatial pyramid attention alone improves the performance individually. Using the interaction-aware attention alone slightly outperforms using the spatial pyramid attention alone. The combination of the interaction-aware attention and the spatial pyramid attention yields more accurate results.

Table 4. Evaluation of the components of the attention model with Inception-ResNet-V2 and the attention layer: K=3 for training and K=25 for testing on the UCF101 split 1 dataset.

Stream or fusion	Using the interaction-aware attention	Using the spatial pyramid attention alone	Without both the interaction-aware attention and the spatial pyramid attention	Using both the interaction-aware attention and the spatial pyramid attention
RGB alone	87.8%	87.5%	87.3%	**88.2%**
Flow alone	86.1%	85.7%	85.5%	**86.5%**
Late fusion	94.7%	94.4%	94.2%	**95.1%**

The 1 scale in Table 2 corresponds to the vanilla attention method obtained by removing the feature pyramid from the attention modeling (i.e., only using one layer of feature maps) and removing the PCA-inspired interaction-aware modeling. From Tables 2 and 4, it is seen that our attention methods with the feature pyramid attention or with the interaction-aware modeling yield more accurate results than the vanilla attention method.

We compared our attention model with the attention model which uses 1x1 convolutions shared across all the spatial locations. The use of 1x1 convolutions shared across all the spatial locations yields an attention map in which each position has an attention score. In contrast, our attention modeling method has an attention score vector (an attention kernel) at each spatial position. We implemented the method which uses 1x1 convolutions by replacing the attention score vector in each location in our method with the attention score obtained by using 1x1 convolutions shared across all the spatial locations. Using 1x1 convolutions shared across all spatial locations is simple and efficient. However, the correlations between spatial locations are not modeled and described. As shown in Table 5, our method obtains more accurate results than the method that uses 1x1 convolutions. It effectively models the attention correlations between spatial locations.

Table 5. Comparison with using 1x1 convolutions shared across all the spatial locations on the UCF101 split 1 dataset

Method	Accuracy
Our method (25frames for test + Inception-ResNet-V2)	**95.1%**
Using 1x1 convolutions shared across all the locations	94.2%

We also tested the effectiveness of the interaction-aware pyramid attention on the CIFAR 100 image classification dataset (an image-level dataset), The CIFAR100 database contains 60,000 color images. These images were divided into a training set with 50000 images and a test set with 10000 images. CIFAR100 has 20 large classes. It was further subdivided into 100 small classes. The comparison between the interaction-aware pyramid attention layer and the baseline network is shown in Table 6, where CIFAR100+ means that the images were augmented with padding and cropping randomly. The results show that the interaction-aware pyramid attention layer increases the recognition rate by 2.2% compared with the baseline Inception-ResNet-V2. It is seen that the new attention method works well on the image-level task. This supports that it works well in the video-level task when combining with spatial-temporal modeling.

Table 6. Recognition rate (top-1) of the interaction-aware pyramid attention layer and the baseline network on the CIFAR 100+ image dataset.

Method	The interaction-aware pyramid attention	Inception-ResNet-V2
Accuracy	**81.7%**	79.5%

5.1.5. Class analysis for the attention-based models

We analyzed the performance of the attention-based models on different action classes. The classes, for which the classification accuracies are low (less than 52%) from the RGB stream, are “Nunchucks”, “JumpRope”, “HighJump”, “JumpingJack”, and “Hammering”. In “Nunchucks”, in both indoor and outdoor scenes, humans are rotating nunchucks continually. It is impossible to see the motion of the nunchucks clearly with the naked eye. The RGB branch alone (i.e. without motion information) cannot effectively distinguish these actions from other actions. The actions “JumpRope” and “JumpingJack” both have sub-actions “hands up” and “jumping up”. Moreover, the rope in “JumpRope” cannot be distinguished in the image, because of its fast movement. The RGB branch alone cannot clearly distinguish these two classes of actions without the support of motion information. The classification accuracy rates for these two action classes are low if the RGB branch alone is used.

The classes with low classification accuracies (less than 51%) using only the optical flow stream are “BrushingTeeth”, “CricketBowling”, “PizzaTossing”, “Hammering”, and “FieldHockeyPenalty”. As an example, the accuracy for “BrushingTeeth” is low because the actions “BrushingTeeth” are easily misclassified as the actions “Shaving”. For both “BrushingTeeth” and “Shaving”, the main motion in the scene is the small movement of single hand. Optical information alone cannot distinguish whether a toothbrush or shaver is held in the hand.

We checked the effect of the fusion of the RGB branch and the flow branch on different action classes. It is seen that in all the action classes except for the BrushingTeeth class, the fusion of the two branches yields more accurate results than the single branch alone. For the BrushingTeeth class, the fusion accuracy is lower than the accuracy of the RGB branch because of the large differences in the accuracies of the RGB branch and the flow branch. The accuracies for most of the action classes are higher than 70%. This verifies that the two-stream network effectively fuses the appearance information from the RGB branch and the motion information from the flow branch.

We checked the effect of the attention mechanism of the RGB branch on the classification accuracies of different action classes. Table 7 shows the five action classes for which the attention mechanism increases the classification accuracy by the largest amount on the RGB branch. Table 8 shows the five action classes for which the attention mechanism reduces the classification accuracy by the largest amount on the RGB branch. It is seen that when the attention mechanism is

incorporated, on the RGB branch the accuracies for the actions of "Lunges" and "LongJump", etc, are obviously increased and the accuracies for the actions "MopppingFloor" and "BlowDryHair", etc, are reduced to some extent. The reason is that the actions "Lunges" and "LongJump" only include movements of human bodies themselves, without interaction with objects in the background. Although the video backgrounds are complex, they are useless for identifying these actions. As human motion is the core in the dataset, the attention mechanism ensures that the trained networks pay more attention to human bodies and reduces the influence of the background to some extent. Then, the actions of these classes have higher classification accuracy rates. However, the actions "MopppingFloor" and "BlowDryHair" include interactions between human bodies and objects in the background. While the video backgrounds are relatively simple, the external objects are important for classifying these actions. On decreasing the attention to objects that interact with human bodies, the classification accuracy rates of these actions decrease to some extent.

Table 7. The five action classes for which the attention mechanism increases the classification accuracy by the largest amount on the RGB branch

Action class	Classification accuracy increase rate (%)
HammerThrow	24.4
Lunges	24.3
FrontCrawl	21.6
LongJump	20.5
FieldHockeyPenalty	15.0

Table 8. The five action classes for which the attention mechanism reduces the classification accuracy by the largest amount on the RGB branch

Action class	Classification accuracy loss rate (%)
MoppingFloor	17.6
Archery	12.2
Nunchucks	11.4
Rowing	11.1
BlowDryHair	10.5

5.1.6. Extension of the interaction-aware attention layer for different deep networks

To investigate the extension of our attention layer to different networks, we incorporated it into VGGNet-16 and BN-Inception besides Inception-Resnet-V2. For VGGNet-16, the outputs without activation of the last layers of conv3, conv4, and conv5 (i.e., conv3_3 (28×28×512), conv4_3 (14×14×512), and conv5_3 (7×7×512)) were used to construct an interaction-aware spatio-temporal attention pyramid layer with 3 scales. For BN-inception, the outputs without activation of the last layers of inception-3 (28×28×480), inception-4 (14×14×832), and inception-5 (7×7×1024) were used to construct a pyramid layer with 3 scales. For Inception-ResNet-V2, the last layers of Inception-ResNet-A, B, and C without activation were used to construct a 3-scale interaction-aware spatio-temporal pyramid attention layer. Table 9 compares the results with and without our attention layer for the RGB stream, the flow stream, and the late fusion of the RGB and Flow streams. The results without our attention layer are obtained by the two-stream [26] standard process. It is seen that for the RGB stream, the flow stream, and the late fusion of the two streams our attention network improves the performance by

- 3.4%, 1.6%, and 2.1% respectively, for VGGNet-16,
- 2.2%, 0.7%, and 2.6% respectively, for BN-Inception,
- 3.0%, 3.4%, and 2.5% respectively, for Inception-Resnet-V2.

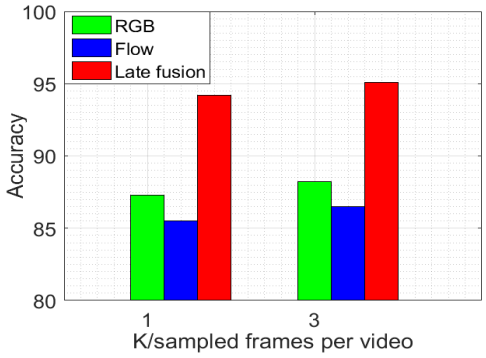
The improvement of our attention layer for all the three types of deep networks proves the generalizability of our layer for general deep CNNs.

Table 9. Performance of the proposed attention layer on popular networks, VGGNet-16, BN-Inception, and Inception-ResNet-V2 on the UCF101 split 1 dataset: K=3 for training and K=25 for testing; both $\mathcal{L}_{interactive}$ and $\mathcal{L}_{attn}$ were used.

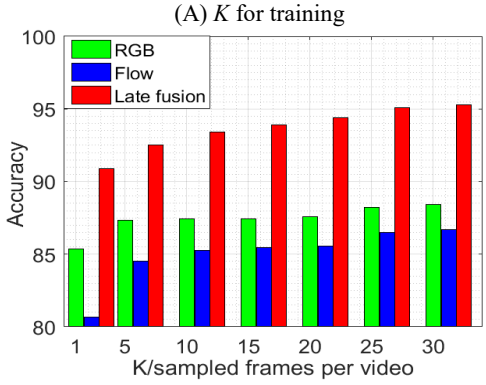
Stream	With or without attention	VGGNet-16	BN-Inception	Inception-ResNet-V2
RGB alone	Without attention	80.4%	84.5%	85.2%
	With attention	**83.8%**	**86.7%**	**88.2%**
Flow alone	Without attention	85.5%	87.2%	83.1%
	With attention	**87.1%**	**87.9%**	**86.5%**
Late fusion	Without attention	90.7%	92.0%	92.6%
	With attention	**92.8%**	**94.6%**	**95.1%**

5.2. Evaluation of temporal aggregations

We investigated how the number K of sampled frames for training and testing affects our interaction-aware attention layer on the UCF101 split1 dataset. The 3-scale spatio-temporal interaction-aware attention pyramid with Inception-ResNet-V2 was used to recognize actions in videos. In the following, the effect of different numbers of sampled frames per video is shown for training first and then for testing.



(A) K for training



(B) K for testing

Fig. 4. Comparisons of the results between different numbers of sampled frames per video for training (A) and testing (B) respectively on the UCF101 split1 dataset: (A) {K=1, 3} frames for training and (B) {K=1, 5, 10, 15, 20, 25, 30} frames for testing.

The effect of the number of frames for training was estimated by changing the number K of frames from 1 to 3 for training and fixing K to 25 for testing. For K=1, a frame was randomly sampled from a video. For K=3, a video was divided into 3 equal segments and a frame was randomly sampled from each of the 3 segments. Then, a sequence of the 3 sampled frames was input to the neural networks. The results of comparison of using different numbers K of sampled frames per video for training is shown in Fig. 4 (A). It is seen that training the model using temporal sequences increases the accuracy. This is consistent with the results obtained by the temporal segment networks in [28]. For a fair comparison, K was set to 3 for training by default for the ablation study,

taking account of the runtime and the limited GPU memory. Dense sampling decreases training speed and requires a large GPU memory. It is necessary to sample multiple videos in order to obtain a minimum number of samples for batch normalization. If too many frames are sampled from each video, then the GPU memory may overflow.

To evaluate the effect of the number K of the sampled frames for testing, we compared the accuracy of action recognition when K is changed for testing and fixed for training. Fig. 4(B) shows the results when K was set to 1, 5, 10, 15, 20, 25, and 30 for testing and fixed to 3 for training. Because of the limited GPU memory, the maximum number of frames for testing was set to 30. It is seen that the action recognition accuracy gradually increases when more frames are sampled per video. Although more frames may introduce irreverent information or even noise for action recognition, our attention layer is able to extract the effective spatio-temporal information for action recognition. A higher accuracy was obtained when K was set to 30. If a GPU with higher memory capacity is used, the accuracy of our model may further increase. For fair comparison, K was set to 25 frames for testing by default for the ablation study as in [26, 28, 39] and the K frames were uniformly sampled from each video.

5.3. Number of parameters

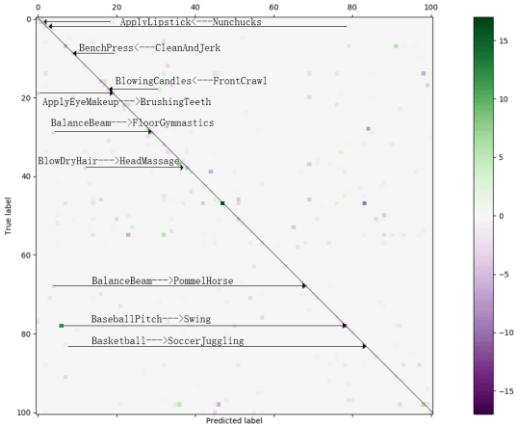
The main computational cost for our interaction-aware attention layer is from the multiplication of two matrices in (2) and (4). Therefore, the computational complexity of our interaction-aware attention layer is $O(C_i W_i H_i)$. The most popular method for investigating the computational cost of an attention layer is to compare the number of the parameters in the networks into which the attention layer is incorporated with the number of the parameters in the corresponding original networks. It has been shown that embedding the proposed attention layer into VGGNet16, BN-Inception, and Inception-ResNet-V2 improves the recognition accuracies over these three original networks. The results shown in Table 10 compare the number of the parameters between the networks into which the interaction-aware attention layer was incorporated and the original networks. The following points are apparent:

- While the original VGGNet16 has 138.56 million parameters, our 3-scale spatio-temporal pyramid attention network has additional 0.26 million parameters. This is only 0.19% of the parameters in the networks.
- While the original BN-Inception has 10.26 million parameters, our 3-scale spatio-temporal pyramid attention network has additional 0.12 million parameters. This is only 1.17% of the parameters in the networks.
- While the original Inception-ResNet-V2 has 57.11 million parameters, our 3-scale spatio-temporal pyramid attention network has additional 0.22 million parameters. This is only 0.38% of the parameters in the networks.

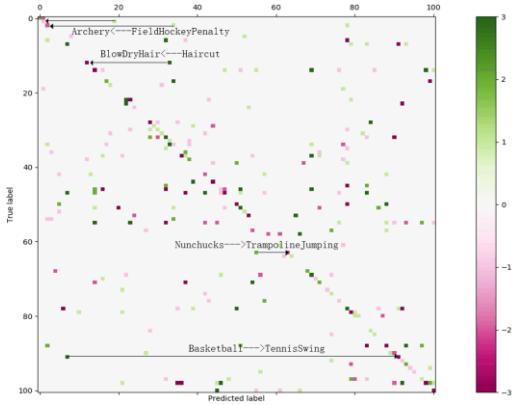
In summary, only a small proportion of new parameters are added when our attention layer is incorporated into the original networks. Therefore, our interaction-aware attention layer improves the recognition accuracy but does not increase the computational cost by much.

Table 10. The numbers (in millions) of the parameters in the networks with or without the proposed interaction-aware attention layer: 1 scale, 2 scales, and 3 scales in a spatio-temporal pyramid were considered.

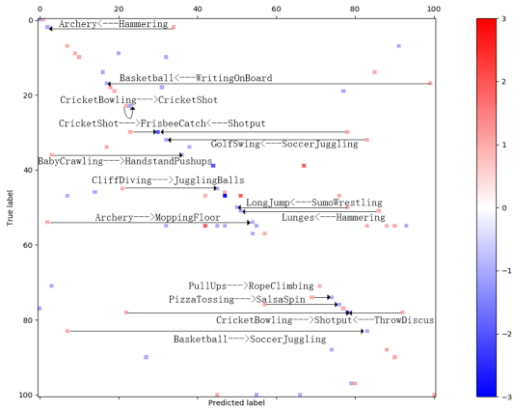
Scale	VGGNet-16	BN-Inception	Inception-ResNet-V2
Original	138.56	10.26	57.11
1 scale	138.62	10.31	57.24
2 scales	138.69	10.36	57.31
3 scales	138.82	10.38	57.33



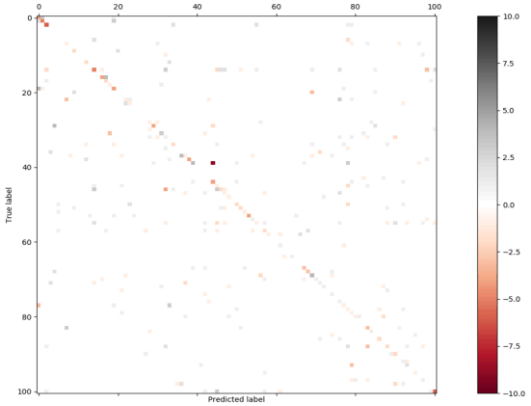
(A) Comparison between 1 scale and no attention



(B) Comparison between 2 scales and 1 scale



(C) Comparison between 3 scales and 2 scales



(D) Comparison between 3 scales and no attention

Fig. 5. Difference of confusion matrices for the RGB stream on the UCF101 split1 dataset: (A) between 1 scale and no attention layer; (B) between 2 scales and 1 scale; (C) between 3 scales and 2 scales; (D) between 3 scales and no attention layer: The larger negative changes on the diagonal and the larger positive changes on the off-diagonal denote the higher improvement; It is expected that the color denoting negative changes is on the diagonal and the color denoting positive changes is on the off-diagonal.

5.4. Analysis of difference of confusion matrices

In order to visualize the difference between the results of our interaction-aware spatio-temporal pyramid attention layers with different scales and the original networks without the attention layer, we examined the difference of their confusion matrices. A confusion matrix is a square matrix with both the numbers of rows and columns equal to the number of classes. Each row represents the class predictions for the samples. Each column represents the true labels of the samples. From the matrix, it is easy to see whether any two classes are confused with each other. The difference of confusion matrices was obtained by the subtraction of the corresponding elements of the two confusion matrices. We computed the confusion matrices of the Inception-ResNet-V2 networks with 1, 2, and 3 scale spatio-temporal pyramid attention layers and without the attention layer. The difference of two of these confusion matrices was computed.

Fig. 5 shows four difference matrices. Fig. 5(A) is the matrix of the difference between the confusion matrices of the networks with the 1-scale spatio-temporal pyramid attention layer and without the attention layer. Fig. 5(B) is the matrix of the difference between the confusion matrices of the networks with the 1 and 2 scale spatio-temporal pyramid attention layers. Fig. 5(C) is the matrix of the difference between the confusion matrices of the networks with the 2 and 3 scale spatio-temporal pyramid attention layers. Fig. 5(D) is the matrix of the difference between the confusion matrices of the networks with the 3-scale spatio-temporal pyramid attention layer and without the attention layer. In these figures, arrows mark some pairs of classes to show the change in the predicted action classes at different scales. The arrows are directed from the class into which samples are misclassified to the class into which the same samples are correctly classified. It is seen that our attention layer with different scales corrects some errors in the classifications for different classes. For example, it corrects the wrongly classified actions "Nunchucks", "CleanAndJerk", "Basketball" to "ApplyLipstick", "BenchPress", "soccerJuggling" when a spatio-temporal pyramid attention layer with 1 scale was used. When layers of more scales were used, the performance is further improved by correcting more errors in clarifying actions. It is demonstrated that the networks with our spatio-temporal pyramid attention layer is more effective than the networks without the attention layer. The inclusion of more scales yields higher recognition accuracy. Our 3-scale spatio-temporal pyramid attention layer significantly improves the performance on most of the classes on UCF101 split 1.

5.5. Fusion of the RGB and flow streams

We compared the performances of the late fusion and the attention fusion of the RGB and flow streams, and estimated the performances of the combinations with motion compensation and motion reinforcement (see the appendix for detail). Table 11 shows the results of different strategies for fusing the RGB channel and the optical flow channel evaluated on the UCF101 split 1 dataset. The following points are apparent:

- The proposed attention fusion method effectively fuses the data from different types of channels. It yields more accurate results than the traditional late fusion method. This further indicates the generalizability and effectiveness of the proposed attention network layer. However, the attention fusion method requires more memory and runs much slower than the late fusion method. Since the accuracy of the attention fusion is not much larger than the accuracy of the late fusion, the late fusion is used by default.
- The motion compensated flow stream increases the classification accuracy. The reason is that the motion compensated flows reduce the influence of background disturbances caused by camera motion and more accurately describe the foreground motion patterns in videos.
- The fusion of the motion reinforced RGB stream with the original flow stream and the motion compensated flow stream increases the classification accuracy, compared with the fusion of the original RGB stream with the two flow streams. This indicates that motion reinforcement gives the RGB branch useful motion information obtained from the optical flow sequence. Features of the foreground motion parts in RGB images are reinforced and then the extracted feature maps are more discriminative.

Table 11. Comparison between different strategies for fusing the RGB channel and the optical flow channel on the UCF101 split 1 dataset

Fusion strategy	Accuracy
Attention fusion (25frames for each test + Inception-ResNet-V2)	95.4%
Late fusion (25frames for each test + Inception-ResNet-V2)	95.1%
Fusion of the RGB stream, the original flow stream, and the motion compensated flow.	95.8%
Fusion of the motion reinforced RGB stream with the original flow stream and the motion compensated flow stream	**96.2%**

5.6. Visualization analysis

We visualized the salient regions that the proposed attention layer extracts from different spatial positions of the feature maps over the frames. Let $\mathbf{x}_{(t,w_n,h_n)} \in \mathbb{R}^{C_i \times 1}$ denote the channel vector of position (w_n, h_n) for frame t in the feature maps $\mathfrak{I}_i \in \mathbb{R}^{K \times W_i \times H_i \times C_i}$. It is a local feature vector describing the receptive field centered at the position (w_n, h_n) in the t-th frame. As stated in Section 3.4, an element $a_{t*w_n*h_n, w_m*h_m}$ in $\mathbf{A}$ ($t \in \{1,2,..,K\}$, $w_m, w_n \in [1, W_i]$, $h_m, h_n \in [1, H_i]$) represents the attention score that local feature vector $\mathbf{x}_{(t,w_n,h_n)}$ contributes to the spatial position (w_m, h_m) in the attention maps $\mathcal{M}_i' \in \mathbb{R}^{C_i \times W_i \times H_i}$. The receptive fields with high attention scores are defined as salient receptive fields.

We visualized the salient receptive fields that the K input frames contribute to the fixed position (w_m, h_m) in the feature maps $\mathfrak{I}_i$. For one frame, the salient receptive fields are centered at the positions satisfying:

$$\{(w_n, h_n) \mid a_{t*w_n*h_n, w_m*h_m} > threshold, t = 1,..., K, w_m = 1, h_m = 1\}. \quad (17)$$

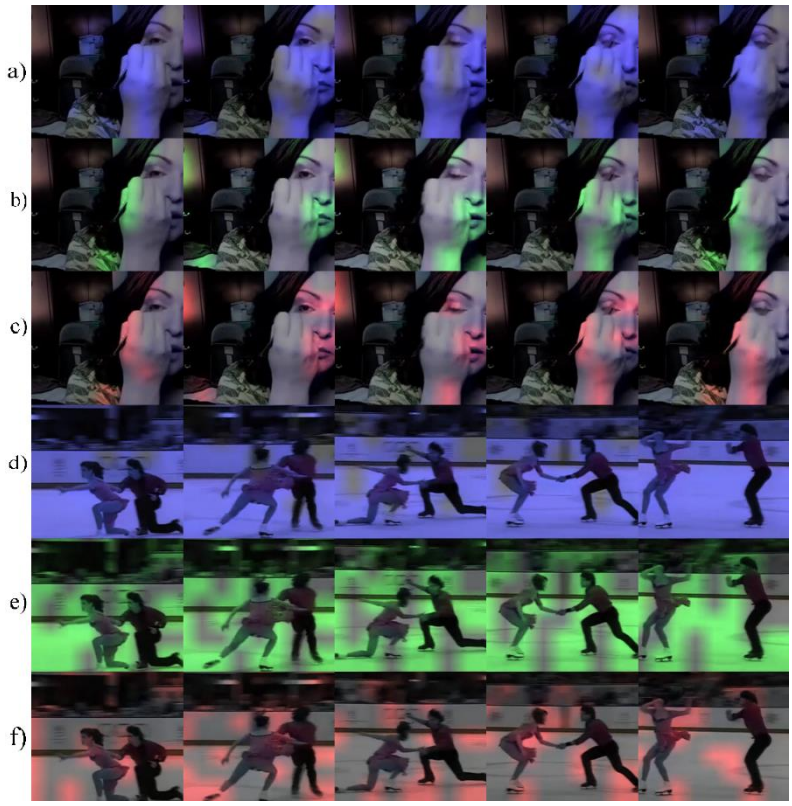
The threshold in (17) was set to 0.5 to show salient attention regions for 5 input frames. Fig. 6 shows some results including the salient regions obtained by the proposed attention layer with 1 scale, 2 scales, and 3 scales (See the supplied video "Fig. 6.mp4"). It is seen that our attention layer pays attention to different salient regions related to human actions over the frames. Using 3 scales pays attention to more specific and accurate action regions in every frame.

For one fixed input frame, we visualized the salient receptive fields contributing to different positions (w_m, h_m) in the attention feature maps $\mathcal{M}_i'$. For one position, every salient spatial region centered at the position satisfies

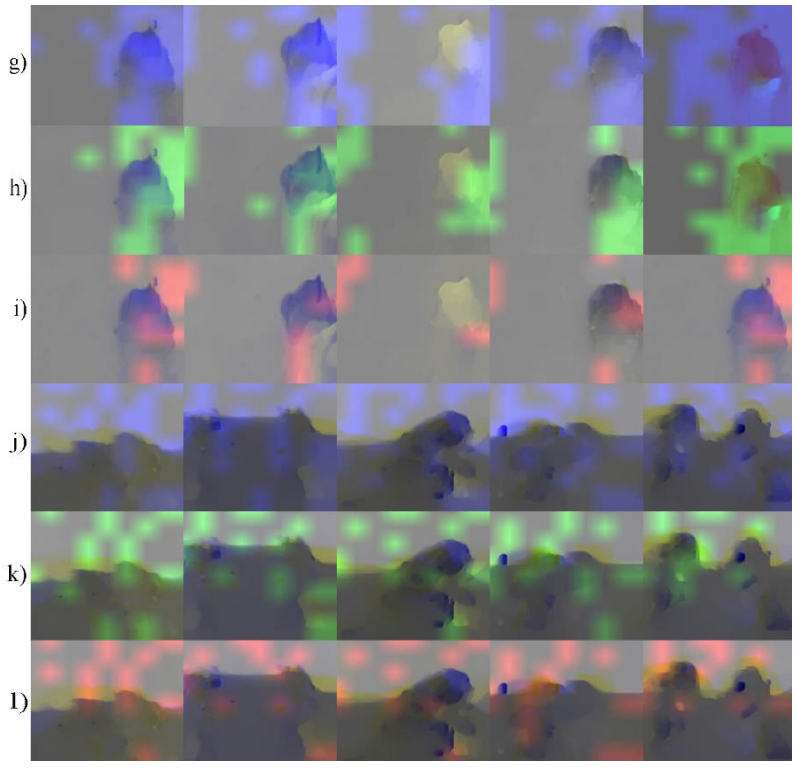
$$\{(w_n, h_n) \mid a_{t*w_n*h_n, w_m*h_m} > threshold, \; t = 1, w_m = 1,..., W_i, h_m = 1,..., H_i\}. \quad (18)$$

We set a higher threshold of 0.7 for stronger discrimination. Some results are shown in Fig. 7 (See the supplied video "Fig. 7.mp4"). It is seen that different positions (w_m, h_m) have

different scopes of attention. For example, in Fig. 7(a) for the action ‘PlayingGuitar’, different parts (‘microphone’, parts of ‘guitar’, hands) receive attention from different positions in the attention feature maps. Similar results can be seen in Fig. 7(b)-(d).



(A) Sampled RGB frames per video



(B) Flow frames per video

Fig. 6. Visualization of salient receptive fields in different frames from the appearance (RGB) and motion (Flow) streams: Each row shows 5 frames from videos, and blue, green, and red regions correspond to the centers of salient receptive fields obtained by using 1 scale, 2 scales, and 3 scales respectively; a), b) and c) and g), h) and i) show the results of action ‘ApplyEyeMakeup’ from the RGB and Flow streams respectively; d), e) and f) and j), k) and l) show the results of action ‘iceDancing’ from the RGB and Flow streams respectively.

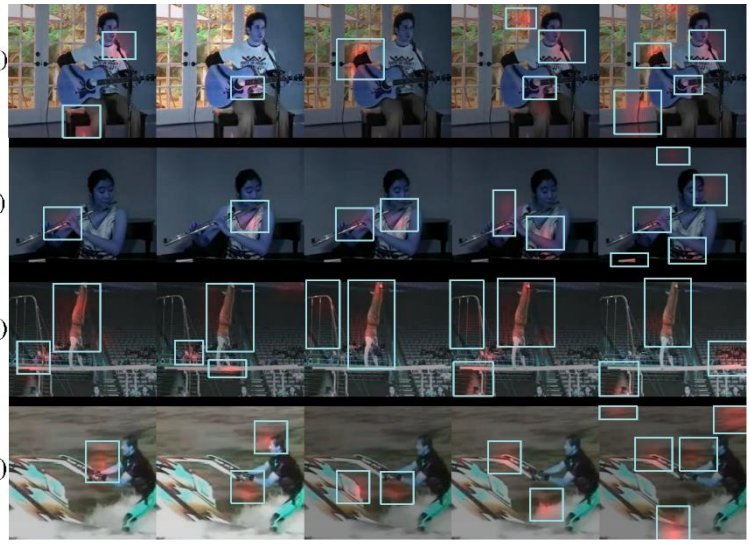


Fig. 7. Visualization of salient receptive fields for different positions in the attention feature maps from the appearance (RGB) stream: a) ‘PlayingGuitar’, b) ‘PlayingFlute’, c) ‘ParallelBars’, d) ‘Skijet’. Three scales of attention were used; Each image shows the results from different positions (w_m, h_m).

5.7. Comparison with the-state-of -the-art

Comparison with the-state-of -the-art was carried out on the UCF101, HMDB51, Kinetics-400, and Charades datasets.

5.7.1. On the UCF101 and HMDB51 datasets

Table 12 compares, on the UCF101 and HMDB51 datasets, the results of incorporating our interaction-aware attention layer into BN-Inception and Inception-Resnet-V2 with the results of recent state-of-the-art methods and with the results of comparable methods. The following points are noted:

Table 12. Comparisons with the-state-of-the art methods on the UCF101 and HMDB51 datasets over 3 splits

Algorithm	UCF101	HMDB51
C3D [34]	85.2%	--
Soft Attention + LSTM [13]	--	41.3%
Two-Stream + LSTM [38]	88.6%	--
TDD+FV [19]	90.3%	63.2%
RNN+FV [58]	88.0%	54.3%
LTC [33]	91.7%	64.8%
ST-ResNet [29]	93.5%	66.4%
TSN (BN-Inception) [28]	94.0%	68.5%
TSN (Journal version) [78]	94.9%	71.0%
AdaScan [59]	89.4%	54.9%
ActionVLAD [39]	92.7%	66.9%
TLE (BN-Inception) [40]	95.6%	71.1%
Attention Cluster (ResNet-152) [47]	94.6%	69.2%
TesNet (ImageNet pre-trained) [84]	95.2%	71.5%
Two-in-one two stream [85]	92.0%	--
R(2+1) [86]	85.8%	54.8%
3D-SqueezeNet [87]	74.9%	--
Algorithm in [88](Pre-trained on Kinetics)	61.2%	33.4%
TSM [89]	95.9%	73.5%
T-STFT [90]	94.7%	71.5%
TEA [91]	96.9%	73.3%
Ours (25 frames+BN-Inception)	94.8%	69.6%
Ours (25 frames+Inception-ResNet-V2)	95.3%	70.5%
Ours (30 frames+Inception-ResNet-V2)	95.5%	70.7%
Fusion of the RGB stream, the original flow stream, and the motion compensated flow.	96.0%	71.1%
Fusion of the motion reinforced RGB stream with the original flow stream and the motion compensated flow stream	96.3%	71.4%

- The results of our methods are much better than the result

of the attention-based method, soft attention + LSTM [13].

- Our method with BN-Inception as the backbone outperforms the conference version of TSN [28] with BN-Inception as the backbone by 0.8% and 1.1% on the UCF101 and HMDB51 datasets respectively, where both the methods uniformly sample 25 frames from each video for testing. Our method performs slightly worse than the journal version of TSN [78]. The reason is that our method sets the number K of sampled frames per video to 3 for training as does the conference version of TSN [28]. The journal version of TSN [78] sets K to 7 for training, which increases the accuracy but requires more runtime. The TSN is currently one of the most effective action recognition methods. Our method yields performance comparable to TSN. This indicates the effectiveness of the proposed method.
- Our method with BN-Inception as the backbone outperforms the attention cluster method with ResNet-152 [47] as the backbone, although ResNet-152 [7] yields more accurate classification results than BN-Inception [52] on the ImageNet dataset [53].
- Our method with Inception-ResNet-V2 as the backbone has a performance which is comparable to the best performance obtained by the TLE method in [40].
- On increasing the number of sampled frames per video from 25 to 30, the performance of our method with Inception-ResNet-v2 as the backbone slightly increases.
- Although many non-trivial adjustments used in the competing algorithms cannot be duplicated in our algorithms, the results of our algorithms are better than the results of most of the competing algorithms and comparable to the results of the competing algorithms which were pre-trained using large datasets [84, 88].
- Our method uses the multi-scale feature fusion to obtain more accurate attention scores. It uses the semantic features in the top level to classify actions, as in other methods, such as Two stream, ActionVLAD, and Attention Clusters. More accurate attention scores make the semantic features more accurate. Therefore, the comparison with other state of the art results indicates the effectiveness of the new attention method.

Table 13. Comparisons with the methods that use iDT [62] and the hybrid methods on the UCF101 and HMDB51 datasets

Algorithm	UCF101	HMDB51
DT+MVSV [60]	83.5%	55.9%
iDT+FV [18]	85.9%	57.2%
C3D+iDT [34]	90.4%	--
LTC+iDT [33]	92.7%	67.2%
ST-ResNet+iDT [29]	94.6%	70.3%
AdaScan+iDT [59]	91.3%	61.0%
ActionVLAD+iDT [39]	93.6%	69.8%
TSN (3-modality) [28]	94.2%	69.4%
AdaScan+iDT+C3D [59]	93.2%	66.9%
Ours (25 frames+Inception-ResNet-V2+iDT)	**95.8%**	**71.8%**
Ours (30 frames+Inception-ResNet-V2+iDT)	**95.9%**	**72.0%**

Fusion with iDT scores [18] is usually able to increase the accuracy of action recognition. The iDT features [18] were obtained by combining the trajectory-based MBH, HOG, and HOF features. An SVM classifier was used to yield the iDT scores for classifying actions. Table 13 compares the results of our method fused with iDT with the results of the methods that use iDT [62] and the hybrid methods on the UCF101 and HMDB51 datasets. The following points are shown:

- Fusion of our method with the iDT increases the accuracy of action recognition.
- The results of our method fused with iDT are more accurate than the results of the competing methods that use iDT [62] and the hybrid methods.
- Increasing the number of sampled frames per video from 25 to 30 slightly increases the accuracy of our method fused with iDT.

5.7.2. On the Kinetics-400 dataset

On the Kinetics-400 dataset, in order to fairly compare with the state of the art methods, we used ResNet-50 with the temporal shift module as the backbone, which has a good balance between accuracy and speed. Similar to the setups on the UCF-101 and HMDB-51 datasets, we embedded our interaction-aware pyramid attention layer after stage 5 in ResNet-50. The feature maps output from the last res-blocks in stages 3-5 were used to construct the feature pyramid.

Because of the large volume of the Kinetics-400 dataset, its optical flow extraction and recognition consume too much computation and storage space. Therefore, only RGB frames were utilized as the input. We randomly initialized the parameters of the interaction-aware pyramid attention layer. The pre-trained model on ImageNet was used to initialize the backbone. When our attention layer was combined with the TSN version in [45], the sampling settings in [45] were followed. Namely, for training, each video was uniformly divided into 8 segments and one frame was sampled randomly from each segment. For testing, 25 frames were uniformly sampled from each video. When our attention layer was combined with TSM, the dense sampling settings in [32] were followed. Namely, for training, 64 consecutive frames were randomly chosen from each video and 8 frames were uniformly sampled from these 64 frames. For testing, each video was uniformly divided into 10 clips. From the first 64 consecutive frames of each clip, 8 frames were uniformly sampled.

The results on the Kinetics-400 dataset are shown in Table 14. It is seen that embedding our interaction-aware spatio-temporal pyramid attention layer into TSN increases the accuracy by 1.1%. Embedding our attention layer into TSM increases the accuracy by 0.8%. The results of our method are more accurate than the result of the R(2+1)D method and comparable to the results of the non-local networks [15]. These clearly validate the effectiveness of our attention model. Although the accuracy of our method is less than some 3D convolution-based methods, such as the SlowFast networks-based method [64], these 3D convolution-based methods require many more network parameters and much more computation than our 2D convolution-based method.

Table 14. The state of the art results on the Kinetics-400 dataset

Methods	Backbone	Frames×Crops×Clips	Top-1(%)
TSN[45](RGB)	BN-Inception	25 × 10 × 1	69.1
	ResNet-50		71.8
R(2+1)D [95]	ResNet-34	32 × 1 × 10	72.0
STM [83]	ResNet-50	16 × 3 × 10	73.7
TSM [89]	ResNet-50	8 × 3 × 10	73.8
I3D [32]	3D Inception-v1	64 × N/A × N/A	72.1
Non-local networks [15]	2D ResNet-50	32x3x10	73.8
	3D ResNet-50		74.9
SlowFast [64]	3D ResNet-50	8 × 3 × 10	77.0
TSN+ours	ResNet-50	25 × 10 × 1	72.9
TSM+ours	ResNet-50	8 × 3 × 10	74.6

5.7.3. On the Charades dataset

We evaluated our method on the challenging untrimmed Charades dataset by following the pipeline of another spatio-temporal aggregation method ActionVLAD [39]. Since a video in the Charades dataset can have multiple labels, the

evaluation was carried out using mean average precision (mAP) and weighted average precision (wAP). For each class, average precision was computed. Mean average precision (mAP) is the mean of the average precision of all the classes. Weighted average precision (wAP) is the average precision across all the classes, weighted by the number of test samples in each class. For our method, two backbones into which our attention layer was embedded were used:

- One is BN-inception without pre-training. The sampling mode in TSN was used.
- The other is the ResNet-50 pre-trained on the Kinetics-400 dataset. For training, 64 consecutive frames were randomly chosen from each video and 8 frames were uniformly sampled from these 64 frames. For testing, each video was uniformly divided into 10 clips. From the first 64 consecutive frames of each clip, 8 frames were uniformly sampled as the input to the neural networks.

Because of the multiple labels for each video in the Charades dataset, during the training the per-category sigmoid loss was used. The comparison results are shown in Table 15, where the results of both versions of the temporal relation network (TRN) in [81] and [57] are shown. The following points are noted:

- Using BN-inception as the backbone, our method with 3 scales and the entire loss exceeds the TSN method by 3.4% of mAP and 5.4% of wAP. Our method outperforms the ActionVLAD method by 2.6% of mAP and 3.4% of wAP. It is clear that on this dataset our interaction-aware loss and multi-scale attention loss have better performance gains than on the other datasets. There is more information irrelevant to human actions in the videos in this dataset and our attention network effectively localizes and detects key regions of human actions and extracts from them more discriminative information for action recognition.
- The mAP of our method with ResNet-50 as the backbone is higher than those of the 2D convolution-based methods, Two-stream [4], ActionVLAD [39], and Asyn-TF [63]. This validates the effectiveness of our attention layer. This mAP of our method is comparable to those of the TRN in [81, 57]. While our method specializes in image information interaction and fusion, the TRN specializes in explicit temporal reasoning. Therefore, the TRN is more adaptable to the Charades dataset in which the videos have strong temporal relations.

Table 15. Comparisons on the untrimmed Charades dataset

Algorithm		mAP	wAP
Two-stream+iDT (best reported) [4]		18.6%	--
RGB stream (BN-inception, TSN style training)		16.8%	23.1%
ActionVLAD (RGB, BN-inception) [39]		17.6%	25.1%
ActionVLAD+iDT [39]		21.0%	29.9%
Asyn-TF [63]		22.4%	--
TRN version in [81]		25.2%	--
TRN version in [57]		20.0%	--
Ours (RGB, BN-Inception, 3 scales)	Entire loss	20.2%	28.5%
	$\mathcal{L}_{\text{inter}}$	19.8%	
	$\mathcal{L}_{\text{attn}}$	18.7%	
	No loss	18.3%	
Ours (ResNet-50, pre-trained using Kinetics-400)		24.3%	33.2%

6. Conclusion

We have proposed an interaction-aware self-attention model, which is inspired by PCA, to extract correlations between local feature vectors in feature maps and yield accurate attention scores. We have introduced the feature pyramid into attention modeling. The feature maps of different scales have been combined to further increase accuracy of the attention scores. We have naturally extended our spatial interaction-aware self-attention model to a temporal model forming a video-level end-to-end network for action classification. The temporal model accepts input of variable numbers of frames and then allows for different numbers of training frames and test frames. Our interaction-aware pyramid attention layer can be included in any CNNs to form end-to-end attention networks. Methods for combining the RGB stream and the optical flow stream have been investigated to increase the accuracy of action recognition. We have investigated the performance of the proposed methods on four popular deep networks, VGG16, BN-Inception, Inception-ResNet-V2, and ResNet-50. State-of-the-art results have been obtained.

References

1. S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," arXiv preprint arXiv:1609.08675, 2016.
2. G. Awad, J. Fiscus, D. Joy, M. Michel, A. Smeaton, W. Kraaij, M. Eskevich, R. Aly, R. Ordelman, M. Ritter, G.J.F. Jones, B. Huet, and M. Larson, "TRECVID 2016: Evaluating video search, video event detection, localization, and hyperlinking," TREC Video Retrieval Evaluation (TRECVID), Nov 2016, Gaithersburg, MD, United States, hal-01854776.
3. H. Idrees, A.R. Zamir, Y.G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah, "The THUMOS challenge on action recognition for videos 'in the wild'," *Computer Vision and Image Understanding*, vol. 155, pp. 1-23, 2017.
4. G.A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, "Hollywood in homes: crowdsourcing data collection for activity understanding," *in Proc. of European Conference on Computer Vision*, pp. 510–526. 2016.
5. H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb, "A large video database for human motion recognition," *in Proc. of IEEE International Conference on Computer Vision*, pp. 2556–2563, 2011.
6. K. Soomro, A.R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," arXiv preprint arXiv:1212.0402, 2012.
7. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *in Proc. of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
8. K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *in Proc. of European conference on computer vision*, pp. 630–645, 2016.
9. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *in Proc. of International Conference on Learning Representations*, San Diego, USA, May 2015.
10. C. Szegedy, S. Ioffe, V. Vanhoucke, and A.A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," *in Proc. of AAAI Conference on Artificial Intelligence*, pp. 4278–4284, 2017.
11. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, and I. Polosukhin. "Attention is all you need," *in Proc. of Advances in neural information processing systems*, pp. 5998–6008, 2017.
12. Z. Lin, M. Feng, C.N.D. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," *in Proc. of International Conference on Learning Representations*, Toulon, France, April 2017.
13. S. Sharma, R. Kiros, and R. Salakhutdinov, "Action recognition using visual attention," arXiv preprint arXiv:1511.04119, 2015.
14. Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C.G. Snoek, "VideoLSTM convolves, attends and flows for action recognition," *Computer Vision and Image Understanding*, vol. 166, pp. 41–50, Jan. 2018.
15. X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," *in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, 2018.
16. K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in

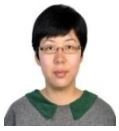
deep convolutional networks for visual recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 904–1916, 2015.
17. T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, 2017.
18. H. Wang and C. Schmid, "Action recognition with improved trajectories," *in Proc. of the IEEE International Conference on Computer Vision*, pp. 3551–3558, 2013.
19. L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4305–4314, 2015.
20. D.G. Lowe, "Object recognition from local scale-invariant features," *in Proc. of IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157, 1999.
21. I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.
22. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 886–893, 2005.
23. A. Klaser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," *in Proc. of British Machine Vision Conference*, pp. 275–279, 2008.
24. N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," *in Proc. European Conference on Computer Vision*, pp. 428–441, 2006.
25. G. Willems, T. Tuytelaars, and L.V. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," *in Proc. of European Conference on Computer Vision*, pp. 650–663, 2008.
26. K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *in Proc. of Advances in Neural Information Processing Systems*, pp. 568–576, 2014.
27. A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," *in Proc. of IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.
28. L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L.V. Gool, "Temporal segment networks: towards good practices for deep action recognition," *in Proc. of European Conference on Computer Vision*, pp. 20–36, 2016.
29. C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," *in Proc. of Advances in Neural Information Processing Systems*, pp. 3468–3476, 2016.
30. S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2012.
31. S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: speed-accuracy trade-offs in video classification," *in Proc. of European Conference on Computer Vision*, pp. 318-335, 2018.
32. J. Carreira and A. Zisserman, "Quo Vadis, action recognition? a new model and the kinetics dataset," *in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017.
33. G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1510–1517, 2018.
34. D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," *in Proc. of the IEEE International Conference on Computer Vision*, pp. 4489–4497, 2015.
35. J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.
36. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
37. J. Donahue, L.A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," *in Proc. of IEEE conference on Computer Vision and Pattern Recognition*, pp. 2625–2634, 2015.
38. J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: deep networks for video classification," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4694–4702, 2015.
39. R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "ActionVLAD: Learning spatio-temporal aggregation for action classification," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 971–980, 2017.
40. A. Diba, V. Sharma, and L.V. Gool, "Deep temporal linear encoding networks," *in Proc. of IEEE conference on Computer Vision and Pattern Recognition*, pp. 2329–2338, 2017.
41. V. Mnih, N. Heess, and A. Graves, "Recurrent models of visual attention," *in Proc. of Advances in Neural Information Processing Systems*, pp. 2204–2212, 2014.
42. J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," arXiv preprint arXiv:1412.7755, 2014.
43. G. Gkioxari, R. Girshick, and J. Malik, "Contextual action recognition with R*CNN," *in Proc. of IEEE International Conference on Computer Vision*, pp. 1080–1088, 2015.
44. A. Mallya and S. Lazebnik, "Learning models for actions and person-object interactions with transfer to question answering," *in Proc. of European Conference on Computer Vision*, pp. 414–428, 2016.
45. https://github.com/open-mmlab/mmaction2.
46. R. Girdhar and D. Ramanan, "Attentional pooling for action recognition," *in Proc. of Advances in Neural Information Processing Systems*, pp. 34–45, 2017.
47. X. Long, C. Gan, G. De Melo, J. Wu, X. Liu, and S. Wen, "Attention clusters: purely attention based local feature integration for video classification," *in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7834–7843, 2018.
48. C.-Y. Ma, A. Kadav, I. Melvin, Z. Kira, G. AlRegib, and H.P. Graf, "Attend and interact: higher-order object interactions for video understanding," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6790–6800, 2018.
49. H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3304–3311, 2010.
50. Y. Du, C. Yuan, B. Li, L. Zhao, Y. Li, and W. Hu, "Interaction-aware spatio-temporal pyramid attention networks for action classification," *in Proc. of European Conference on Computer Vision*, pp. 388-404, 2018.
51. J. Yang, D. Zhang, A. F. Frangi, and J.-Y. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131–137, 2004.
52. S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," *in Proc. of International Conference on Machine Learning*, pp. 448–456, 2015.
53. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
54. J. Lei, Y. Jia, B. Peng, and Q. Huang, "Channel-wise temporal attention network for video action recognition," *in Proc. of IEEE International Conference on Multimedia and Expo*, pp. 562–567, July 2019.
55. J. Chen, Y. Song, and Y. Zhang, "Spatial mask ConvLSTM network and intra-class joint training method for human action recognition in video," *in Proc. of IEEE International Conference on Multimedia and Expo*, pp. 1054–1059, July 2019.
56. C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L1 optical flow," *in Proc. of DAGM Conference on Pattern recognition*, pp. 214–223, 2007.
57. https://github.com/gsig/PyVideoResearch/blob/master/baseline_exp/temporal_relational_networks_charades.py.
58. G. Lev, G. Sadeh, B. Klein, and L. Wolf, "RNN Fisher vectors for action recognition and image annotation," *in Proc. of European Conference on Computer Vision*, pp. 833–850, 2016.
59. A. Kar, N. Rai, K. Sikka, and G. Sharma, "AdaScan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3376–3385, 2017.
60. Z. Cai, L. Wang, X. Peng, and Y. Qiao, "Multi-view super vector for action recognition," *in Proc. of IEEE Conference on Computer*

Vision and Pattern Recognition, pp. 596–603, 2014.
61. W. Dong, Z. Zhang, and T. Tan, "Attention-aware sampling via deep reinforcement learning for action recognition," *in Proc. of AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8247–8254. July 2019.
62. T. Bouwmans, "Recent advanced statistical background modeling for foreground detection-a systematic survey," *Recent Patents on Computer Science*, vol. 4, no. 3, pp. 147–176, 2011.
63. G.A. Sigurdsson, S. Divvala, A. Farhadi, and A. Gupta, "Asynchronous temporal fields for action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 585–594, 2017.
64. C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast Networks for video recognition," *in Proc. of IEEE International Conference on Computer Vision*, pp. 6202–6211, 2019.
65. C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 591–600, 2020.
66. R. Gao, T.-H. Oh, K. Grauman, and L. Torresani, "Listen to look: action recognition by previewing audio," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10457–10467, 2020.
67. J. Zhang, F. Shen, X. Xu, and H.T. Shen, "Temporal reasoning graph for activity recognition," *IEEE Trans. on Image Processing*, vol. 29, pp. 5491–5506, 2020.
68. J. Ji, R. Krishna, L. Fei-Fei, and J.C. Niebles, "Action genome: actions as compositions of spatio-temporal scene graphs," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10236–10247, 2020.
69. D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12046–12055, 2019.
70. J. Li, J. Liu, Y. Wang, S. Nishimura, and M.S. Kankanhalli1, "Weakly-supervised multi-person action recognition in 360◦ videos," *in Proc. of IEEE Winter Conference on Applications of Computer Vision*, pp. 497–505, March 2020.
71. L. Wang, Y. Xiong, D. Lin, and L.V. Gool, "UntrimmedNets for weakly supervised action recognition and detection," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4325–4334, 2017.
72. C. Zhuang, T. She, A. Andonian, M.S. Mark, and D. Yamins, "Unsupervised learning from video with deep neural embeddings," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9563–9572, 2020.
73. X. Wang and A. Gupta, "Videos as space-time region graphs," *in Proc. of European Conference on Computer Vision*, pp. 413–431, 2018.
74. F. Baradel, N. Neverova, C. Wolf, J. Mille, and G. Mori, "Object level visual reasoning in videos," *in Proc. of European Conference on Computer Vision*, pp. 105–121, 2018.
75. H. Huang, L. Zhou, W. Zhang, J.J. Corso, and C. Xu, "Dynamic graph modules for modeling object-object interactions in activity recognition," *in Proc. of British Machine Vision Conference*, pp. 1–3, 2019.
76. Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," *International Journal of Computer Vision*, vol. 128, no. 1, pp. 74–95, 2020.
77. S. Zhang, S. Guo, L. Wang, W. Huang, and M. Scott, "Knowledge integration networks for action recognition," *in Proc. of AAAI Conference on Artificial Intelligence*, pp. 12862–12869, 2020.
78. L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L.V. Gool, "Temporal segment networks for action recognition in videos," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2740–2755, 2019.
79. D. He, Z. Zhou, C. Gan, F. Li, X. Liu, Y. Li, L. Wang, and S. Wen, "StNet: Local and global spatial-temporal modeling for action recognition," *in Proc. of AAAI Conference on Artificial Intelligence*, pp. 8401–8408, 2019.
80. Z. Zheng, G. An, D. Wu, and Q. Ruan, "Global and local knowledge-aware attention network for action recognition," *IEEE Trans. on Neural Networks and Learning Systems*, Accepted.
81. B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," *in Proc. of European Conference on Computer Vision*, pp. 803–818, 2018. http://openaccess.thecvf.com/content_ECCV_2018/papers/Bolei_Zhou_Temporal_Relational_Reasoning_ECCV_2018_paper.pdf.
82. J. Li, X. Liu, M. Zhang, and D. Wang, "Spatio-temporal deformable 3D ConvNets with attention for action recognition," *Pattern Recognition*, vol. 98, Article no. 107037, pp. 1–9, Feb. 2020.
83. B. Jiang, M.M. Wang, W. Gan, W. Wu, and J. Yan. "STM: Spatiotemporal and motion encoding for action recognition," *in Proc. of IEEE International Conference on Computer Vision*, pp. 2000–2009, 2019.
84. G. Huang and A.G. Bors, "Learning spatio-temporal representations with temporal squeeze pooling," *in Proc. of International Conference on Acoustics, Speech and Signal Processing*, pp. 2103–2107, May 2020.
85. J. Zhao and C.G.M. Snoek, "Dance with flow: two-in-one stream action detection," *in Proc. of Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9935–9944, 2019.
86. R. Girdhar, D. Tran, L. Torresani, and D. Ramanan, "DistInit: Learning video representations without a single labeled video," *in Proc. of IEEE International Conference on Computer Vision*, pp. 852–861, 2019.
87. O. Kopuklii, N. Kose, A. Gunduz, and G. Rigoll, "Resource efficient 3D convolutional neural networks," *in Proc. of IEEE International Conference on Computer Vision Workshop*, pp. 1910–1919, Oct. 2019.
88. J. Wang, J. Jiao, L. Bao, S. He, Y. Liu, and W. Liu, "Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4006–4015, 2019.
89. J. Lin, C. Gan, and S. Han, "TSM: Temporal shift module for efficient video understanding," *in Proc. of IEEE International Conference on Computer Vision*, pp. 7083–7093, 2019.
90. S. Kumawat, M. Verma, Y. Nakashima, and S. Ramanar, "Depthwise spatio-temporal STFT convolutional neural networks for human action recognition," Xiv:2007.11365v1[cs.CV], Jul 2020.
91. Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, and L. Wang, "TEA: Temporal excitation and aggregation for action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 909–918, 2020.
92. C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1933–1941, June 2016.
93. C. Feichtenhofer, A. Pinz, and R.P. Wildes, "Spatiotemporal multiplier networks for video action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4768–4777, 2017.
94. W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," CoRR, vol. abs/1705.06950, 2017, http://arxiv.org/abs/1705.069502017.
95. D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. "A closer look at spatiotemporal convolutions for action recognition," *in Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6450–6459, 2018.

Weiming Hu received the Ph.D. degree from the department of computer science and engineering, Zhejiang University in 1998. From April 1998 to March 2000, he was a postdoctoral research fellow with the Institute of Computer Science and Technology, Peking University. Now he is a professor in the Institute of Automation, Chinese Academy of Sciences. His research interests are in visual motion analysis, recognition of web objectionable information, and network intrusion detection.

Haowei Liu received his B.S. degree in Automation at Tsinghua University in 2019. He is currently a first-year Ph.D. candidate in the Institute of automation, University of Chinese Academy of Sciences (UCAS). His research is mainly on computer vision and machine learning, particularly the theory and applications of human action recognition.

Yang Du received B.S. degree of automation in Beijing Jiaotong University, Beijing, China, in 2014, and received Ph.D. degree in the Institute of Automation, University of Chinese Academy of Sciences (UCAS) in 2019. He mainly does research on action recognition and architectures of deep networks.



Chunfeng Yuan received the doctoral degree from the Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, China, in 2010. She is currently an associate professor at the CASIA. Her research interests and publications range from statistics to computer vision, including sparse representation, motion analysis, action recognition, and event detection.

Bing Li received the PhD degree from the Department of Computer Science and Engineering, Beijing Jiaotong University, China, in 2009. Currently, he is a professor in the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences. His research interests include color constancy, visual saliency and web content mining.

Stephen Maybank received a BA in Mathematics from King's College Cambridge in 1976 and a PhD in computer science from Birkbeck college, University of London in 1988. Now he is a professor in the Department of Computer Science and Information Systems, Birkbeck College. His research interests include the geometry of multiple images, camera calibration, visual surveillance etc.