future internet

MDPI

Article

3D-CNN-Based Fused Feature Maps with LSTM Applied to Action Recognition

Sheeraz Arif *, Jing Wang *, Tehseen Ul Hassan and Zesong Fei

Information and Communication Engineering, Beijing Institute of Technology, Beijing 100081, China; tehseen@bit.edu.cn (T.U.H.); feizesong@bit.edu.cn; (Z.F.)

* Correspondence: Sheeraz.arif@bit.edu.cn (S.A.); wangjing@bit.edu.cn (J.W.)

Received: 20 December 2018; Accepted: 6 February 2019; Published: 13 February 2019

Abstract: Human activity recognition is an active field of research in computer vision with numerous applications. Recently, deep convolutional networks and recurrent neural networks (RNN) have received increasing attention in multimedia studies, and have yielded state-of-the-art results. In this research work, we propose a new framework which intelligently combines 3D-CNN and LSTM networks. First, we integrate discriminative information from a video into a map called a 'motion map' by using a deep 3-dimensional convolutional network (C3D). A motion map and the next video frame can be integrated into a new motion map, and this technique can be trained by increasing the training video length iteratively; then, the final acquired network can be used for generating the motion map of the whole video. Next, a linear weighted fusion scheme is used to fuse the network feature maps into spatio-temporal features. Finally, we use a Long-Short-Term-Memory (LSTM) encoder-decoder for final predictions. This method is simple to implement and retains discriminative and dynamic information. The improved results on benchmark public datasets prove the effectiveness and practicability of the proposed method.

Keywords: action recognition; fused features; 3D convolution neural network; motion map; long short-term-memory

1. Introduction

Human activity recognition (HAR) is one of the enabling technologies behind human-computer interactions, video surveillance and video scene understanding [1]. To date, it imposes significant challenges such as the frequent presence of background clutter, view point changes, irregular motion, intra-class variations and camera motion. In addition, the huge information redundancy in video requires large amounts of memory, and also, the discovery of discriminative information from video frames is very complex and slow process.

The result of various research studies indicates that the success of action recognition problems depends on an appropriate feature extraction process. The appropriate feature extraction is very important in distinguishing samples and variations in the frames. Considerable progress has been made to address this problem by employing various specific solutions. Many local space-time visual representations have been proposed to overcome these issues in action recognition tasks. Laptev [2] detected sparse-time interest points and computed a histogram of the detected local points. Hessian [3], local trinary patterns (LTP) [4], Cuboids [5], and 3-D SIFT [6]) have also shown promising levels of HAR effectiveness, mainly thanks to their robustness against partial occlusions and noise. To facilitate a more effective usage of motion information, many trajectory-based feature extraction approaches have been proposed, such as KLT-tracker [7], SIFT matching [8], DTF [9], improved DTF [10]. However, there are number of weaknesses in these models, such as the presence of irrelevant and redundant trajectories, computational complexity and blending of unnecessary motion.

The ideal video representation method must be efficient to compute and simple to implement instead of using complicated and labor-intensive feature extraction and encoding methods. The extraction of spatiotemporal features from video frame sequences is widely used for recognizing human actions. Due to the advancement of digital camera technology, it has become possible to capture depth information, which can be embodied into a single motion map. Compared to dynamic and conventional images, motion maps can provide 3D information, and can be insensitive to changes in light conditions. Much research efforts has used depth imagery such as dynamic images [11] and depth maps [12] in the context of action recognition. These methods are able to process temporal information, but are insufficient to capture dense and discriminative information in terms of shape, appearance and motion.

Recently, deep convolutional neural networks (DCNNs) and Long Recurrent Convolutional networks (LRCNs) have shown great potential in many areas, and have yielded promising results for many computer vision tasks. These approaches have the ability to accurately identify the hidden pattern in visual data by back propagation, so features are auto-extracted without any artificial selection. It has been proven empirically that features learned from deep neural networks are much better than hand-crafted features.

In light of the above analysis, this research article examines the issue of human action recognition by using motion maps and intelligently incorporating a C3D network with a Long Recurrent Convolutional network (LRCN) network. We utilize a 3D convolutional neural network (C3D) [13] to acquire and integrate the temporal information. The C3D can model appearance and motion information simultaneously. Our model integrates a motion map of the previous frames with the next frame to generate a new motion map. We can get a motion map of the whole video after the repetitive integration of the next frame for various-length videos. We use a linear weighted fusion method to fuse feature maps to take advantage of spatiotemporal features. Finally, we use LSTM for feature encoding and action classification. The proposed method is simple to implement and acquires temporal information effectively, integrating it into a map without losing the discriminative information of videos. The proposed method shows significantly improved results over some baseline methods when applied to the various benchmark video datasets. It is worth highlighting the following contributions:

1. We propose an iterative training method for our neural network to generate a motion map from input video, which can integrate information into a motion map from each video frame.
2. We intelligently incorporate C3D and LSTM networks and capture long-range spatial and temporal dynamics. C3D features on video shots contain richer motion information; LSTM can explore the temporal relationship between video shots.
3. We introduce an effective fusion technique i.e., a linear weighted fusion method which can fuse correspondence between spatial and temporal features and boost recognition accuracy.
4. The effectiveness of our approach is evaluated on benchmark datasets, in which it obtained state-of-the-art recognition results.

The remainder of this article is organized as follows: Section 2 reviews related works. In Section 3, we present our proposed approach in detail. We demonstrate the experimental evaluation in Section 4. Finally, a conclusion is presented in Section 5.

2. Related Work

Over the last decade, researchers have presented many hand-crafted and deep-net-based methods for action recognition. Earlier works were based on hand-crafted features for non-realistic actions videos. Since the proposed method is based on deep neural network (DNN), in this section, we will only review related works based on DNN.

In recent years, different variants of deep learning models have been proposed for human activity recognition in videos, and have achieved great performance for computer vision tasks. Ji et al. [14]

applied 3D convolutional kernels on video frames in a time axis to capture both spatial and temporal information. Karpathy et al. [15] directly applied CNNs to multiple frames in each sequence and obtained the temporal relations by pooling, using single, late, early and slow fusion; however, the results of this scheme were just marginally better than those of a single frame baseline. Simonyan and Zisserman [16] used a two-stream CNN framework to incorporate both feature types, with one stream taking RGB image frames as the input and the other taking pre-computed stacked optical flows. Since optical flow contains only short-term motion information, adding it does not enable CNNs to learn long-term motion transitions. The additional stream significantly improved action recognition accuracy, indicating the importance of motion features. Tran et al. [13] avoided the need for pre-computing optical flow features through their 3D convolution (C3D) framework, which allows deep networks to learn temporal features in an end-to-end manner. However, C3D only covers a short range of the sequence. Wang et al. [17] introduced a temporal segment network (TSN) architecture, where a sparse temporal sampling strategy is adopted to model long-term temporal structures. In [18], Feichtenhofer et al. study a number of ways of fusing CNN towers in order to take advantage of this spatial-temporal information from the appearance and optical flow networks. However, the CNN-based method only extracts visual appearance features, and lacks the long-range temporal modeling capabilities. Moreover, the CNN-based method ignores the intrinsic difference between spatial and temporal domains.

Some researchers have also presented methods by uniting the benefits of both hand-crafted and deep learned features, such as [19,20], and obtained good results. They integrate the key factors from two successful video representations, namely improved trajectories [10] and two-stream ConvNets [18]. How to combine the benefits of these two kinds of features to design good descriptors has been an active research area. Some research efforts have been carried out using depth imagery such as dynamic images and depth maps. Bilen et al. [11] introduced the dynamic image network to generate dynamic images for action videos. The order of video frames is used as the supervisory information; however, this method loses some discrimination information. Chen et al. [12] represented a model in the form of depth maps in the context of action recognition. These contributions showed good action recognition results but were insufficient to capture dense and discriminative information in terms of shape, appearance and motion. Taylor et al. [21] used a convolutional gated restricted Boltzmann machine to generate a flow field of the adjacent two frames in the video for action recognition, but this model could not generate a single map to represent a video. Rank pooling [22] and Fisher Vector [23] made an attempt to generate the desire length motion map. However, these methods are unable to model temporal dynamics among video frames.

In order to model the temporal dynamics among video frames, RNNs have been considered for video-based HAR. RNN networks provide strength to find and process hidden patterns in time-space data. In these kind of systems, data is processed in a sequential way, such that at each time t, it gets input from the previous hidden state s_{t-1} and obtains new data x_t. Most of the state-of-the-art methods [24–29] have proposed their own recurrent networks by leveraging CNNs and RNNs for action recognition, and have achieved impressive performance. However, due to the large number of calculations of parameters, and negligence of effect of initial input after few layers, vanishing gradient problems occurred. The solution to this problem is LSTM [25,27,30], which has the ability to capture long-term dependencies and preserve sequence information over time by integrating memory units. LSTM was first introduced by [31]; it has been successfully adapted to many sequential modelling tasks such as speech recognition, visual description and machine translation, and has achieved encouraging performance. In most of these networks, the inputs to the LSTM are the high level features captured from a fully-connected layer of CNN. LSTM units use multiplicative gates to control access to the error signal propagating through the networks.

In this paper, we propose a 3Dconv-based iterative training method to generate the motion map, enabling the use of existing CNN models directly on video data with fine-tuning. Our model efficiently integrates the temporal information of the motion map and video frames and generates the arbitrary

length of the motion map. The Combination of CNN-RNN provides effective representation for long-term motion and modeling of the sequential data, each of which has a time relationship with adjacent points. (RNN uses the extracted C3D features as inputs and models more robust, longer-range features.) The C3D network is able to encode local temporal features within each video unit; it cannot model across the multiple units of a video sequence. We thus introduce LSTM to capture global sequence dependencies of the input video and cues on motion information. The fused spatio-temporal features are processed by LSTM, which helps recognizing complex frame-to-frame hidden sequential patterns. After conducting extensive experiments, we observed that our method is very effective for videos of various lengths, and shows significant improvement in action recognition.

3. The Proposed Approach

In this section, the proposed approach and its related components are discussed. The process of action recognition is divided into two parts: The first part is related to the extraction of spatiotemporal fused features, so we discuss this within the relevant subsequent sections, e.g., the generation of motion maps and the training of motion map networks. Finally, we explain the encoding of the extracted features and the action classification part in the main subsequent section.

3.1. Extraction of Spatio-Temporal Fused Features

3.1.1. Generation of Motion Map

A motion map is a powerful and compact representation of a video which can be useful in computer vision tasks. The motion map can visualize motion information in good manner, and can remove a large amount of information redundancy of the video, thereby revealing discriminative information. The calculation of the motion map is fast, and takes up fewer memory resources. Hence, using a map to represent the video has realistic requirements. Our propose model is very simple to implement and can be trained by increasing the training video length iteratively. Mainly, it is very helpful to solve the problem of videos of various lengths to get the same effect of the map representation, and also to integrate the temporal information into a map without losing the discriminative information of the video. Another advantage of this method is that we can extract a constant number of video frames per second, which improves the generalization performance of the network. We can utilize a 3D-convolutional neural network for the extraction of the motion map. 3D convolution and 3D pooling operations are adopted in 3D ConvNets. Three-dimensional convolution is the extension of 2D convolution. The output of the 2D convolution are two-dimensional feature maps, while the output volume of 3D convolution can have multiple-dimensions. Each feature map of the convolutional layer is connected with some successive adjacent frames in upper layer. As a result, the temporal information is not lost and the motion of the human body can be efficiently captured. Hence, multiple 3D convolutional layers can be used to handle the spatial and temporal information of the inputs in a hierarchal way.

For a video V with N frames, we define the video frames as fi, i {1 . . . . . N}. F_i denotes the motion map from f_1 to f_i. In order to retain appearance and action information, we introduce an iterative method to generate a new motion map F_{i+1} using Equation (1) by combining the current motion map F_i with the future video frame f_{i+1} by using MMN. Symbol $\oplus$ is the pixel-wise addition between motion map and video frame. The process of generating our first and final motion map is shown in Figure 1a,b respective

$$F_{i+1} = F_i \oplus f_{i+1} \tag{1}$$

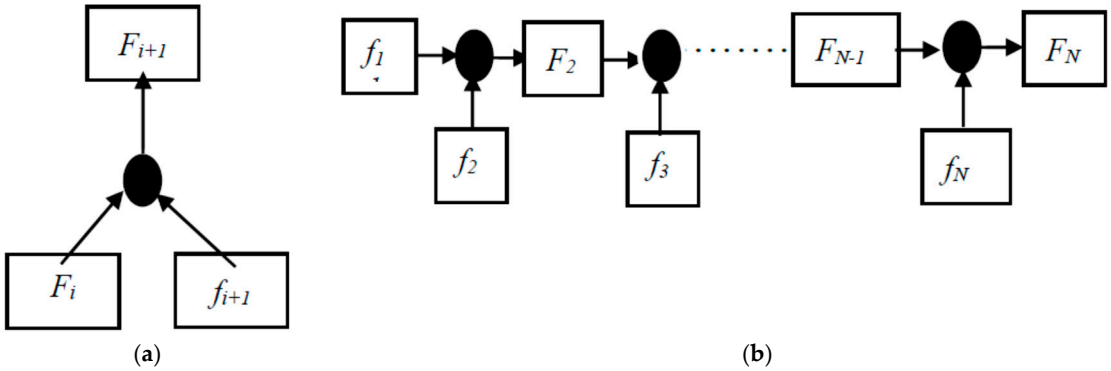

Figure 1. (**a**,**b**) Generation of our first and final Motion Map.

In the last iteration of our MMN network, we obtain the final motion map F_N of video V, in which discriminative information is embodied and can be applied for action recognition tasks. Some motion maps generated by our C3D network are listed in Figure 2. Each map highlights the static object with its main features, and the superposed silhouette incarnates the different locations and postures of the actor and objects. For example, the action category playing violin shows that the actor, arms and violin are the main features, while the rest of the image is diluted. This shows the relationship between the arm movements and playing the violin. It reflects the motion relationship between the actor and the object, and proves that the dynamic information which is available in different sequences of the video can be retained and embodied in the motion map.

Figure 2. Output Motion map generated by our network, illustrating the discriminative information integrated into a single motion map to classify the video category.

3.1.2. C3D Network Architecture

The C3D network has the ability to learn visual patterns directly from pixels without any pre-processing step. The architecture of C3D comprises trainable filters and local pool operations, which is very useful to find hidden patterns in a video frame, and captures all changes in terms of spatial and temporal information.

The architecture of the C3D network is given in Figure 3. Table 1 illustrates the different parameter settings of each convolutional and pooling layer. We set the 3D Convolution and pooling kernel size as $d \times k \times k$, where d is kernel temporal depth and k is kernel spatial size. The 3D convolution is achieved by convolving a 3D kernel to the cube formed by stacking multiple contiguous frames together. By this construction, the feature maps in the convolution layer are connected to multiple contiguous frames in the previous layer, thereby capturing motion information. Intuitively, these different layers describe

the visual content at different level, each of which is complementary to each other for the task of recognition. The C3D network has 5 convolution layers and 5 pooling layers (each convolution layer is immediately followed by a pooling layer), 2 fully connected layers and softmax loss layer. The number of channels (filters) for 5 convolution layers from 1 to 5 is 64, 128, 256, 512, and 512 respectively. The ratio represents the spatial map size ratio. In both spatial and temporal dimensions, all convolutional layers have $3 \times 3 \times 3$ convolution filters with stride $1 \times 1 \times 1$. All pooling layers from pool2 to pool5 (except for the first layer) have $2 \times 2 \times 2$ pooling kernels with stride $2 \times 2 \times 2$, which means the size of the output signal is reduced by a factor of 8 compared with input signal. The first pooling layer, i.e., the pool1 layer, has a kernel size of $1 \times 2 \times 2$, with the goal of not merging the temporal signal and preserving the temporal information in the early phases. The output of each convolution-al layer is a kind of volume in the form of feature maps. All pooling layers lead to the same number of feature maps as convolution layers but with reduced spatial resolution; also, these pooling layers introduce scale-invariant features. The two fully connected layers have 2048 outputs, and finally, a softmax layer is used to predict action labels.

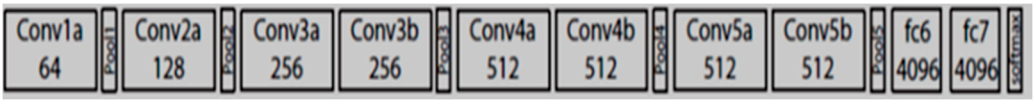

Figure 3. Complete Network architecture of C3D.

Table 1. The convolutional and pooling layers of the C3D architecture.

Layers	**Conv1a**	**Conv2a**	**Conv3a**	**Conv3b**	**Conv4a**	**Conv4b**	**Conv5a**	**Conv5b**
Size	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$
Stride	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$
Channel	64	128	256	256	512	512	512	512
Ratio	1	1/2	1/4	1/4	1/8	1/8	1/16	1/1
Layers	**Pool1**	**Pool2**	**Pool3**	**Pool4**	**Pool5**	**Fc6**	**Fc7**	
Size	$1 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	-	-	
Stride	$1 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	-	-	Softmax
Channel	64	128	256	512	512	4096	4096	Layer
Ratio	1/2	1/4	1/8	1/16	1/32	-	-	

Figure 4 illustrates the single iteration process of our Motion Map Network (MMN). The input to our network is frame-by-frame RGB clip. A motion map and the next video frame are combined into a video frame sequence as input, and a single 2D-feature map is extracted as output. At this stage, it is very important to mention that the feature maps are extracted in a frame-by-frame manner. We compute feature maps of layer conv5b from the input videos, and the rest of the pool5 and full-connected layers are abandoned in our scheme. C3D conv5b feature maps have the highest activation projected back to the image space. In each iteration, the output of the conv5b layer generates two feature maps, each with a size of $7 \times 7 \times 512$, where 7×7 is the spatial size of the feature maps with 512 channels. So, we build only one feature map of $7 \times 7 \times 512$ by taking the maximum value for each position of the both feature maps from conv5b. This process is applied for all iterations for our pipeline, except the last iteration, because the output of the last iteration is again with two feature maps. We will apply linear weighted fusion to the last two feature maps by taking advantage of spatial-temporal features to obtain our final feature map. The discriminative information embodied in the final motion map can be applied to human action recognition tasks.

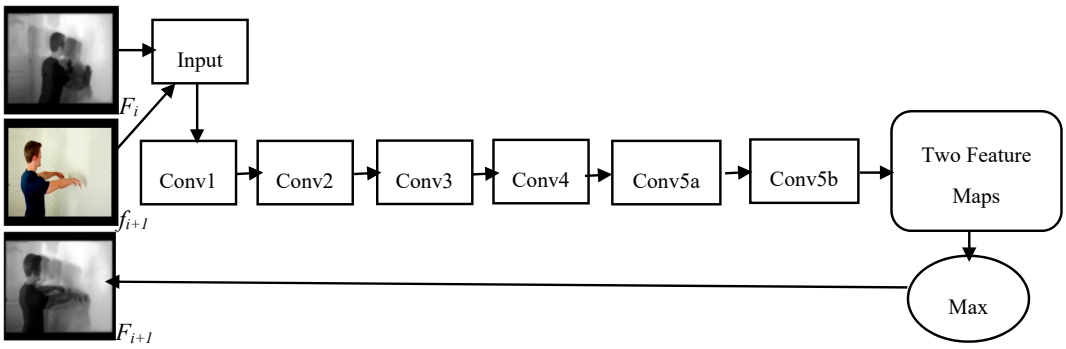

Figure 4. The structure of our Motion Map Network, which illustrates the single iteration to generate Motion Map.

3.1.3. Training of Motion Map Network (MMN)

Since the network can only handle video frames, the videos need to be processed as video frames. Some methods directly split the videos, ignoring different frame rates, so dynamic information may be inconsistent in time. Therefore, we extract a constant number video frames per second, which improves the generalization performance of the network. For the network to better capture the changes in the action, we extract two frames per second. As for some short videos, we loop the extracted frames, and fill up 16 frames per video.

We introduce an iterative method to train the Motion Map Network (MMN). We use video V with N frames and video labels L to train our MMN. S is defined as maximum training iteration length. We train MMN using training length s from 2 to S. The training length s is the round of iteration. We cut the training video V_i into s-length clips C_j^i ($j \in 1 \ldots .N_\mathrm{i}/s$) with overlap 0.7 and assign the labels L_i to clip C_j^i. We define the MMN as a function $Z_{\theta_s}(I_a, I_b)$, where θ_s denotes the parameters of MMN after the iteration of training length s. The initial parameters of the network are defined as θ_1. For each s, we generate the motion map $F_{1\sim s-1}^{c_j^i}$ using $Z_{\theta_{s-1}}$, and train the MMN using the motion map $F_{s-1}^{c_j^i}$, video frame $f_s^{c_j^i}$ and video clip label L_i. Finally, we can get θ_s which is the parameter of our trained motion map network. The detail of the training steps is summarized in Algorithm 1.

Algorithm 1. Training of Our Motion Map Network

Input: V is Video dataset; Frame number of video dataset, N;
Video labels, L; Maximum training iteration length, S;
Parameters of our model, θ_1;
Output: Final parameters of Network, θ_s;
1: Initialize the parameter θ_1 for our model;
2: ***for*** each $s \in 2,3, \ldots, S$ ***do***
3: cut V_i into s-length clips $C_i^j (j \in 1 \ldots N_\mathrm{i}/s)$ with overlap 0.7;
4: Extract the video frames from C_i^j as $f^{C_i^j}$;
5: ***for*** each $j \in 1, 2 \ldots N/s$ *do*
6: ***for*** $k \in 1, 2 \ldots s-1$ ***do***
7: Generate the motion map $F_k^{C_i^j}$ using $Z_{\theta_{s-1}}(F_k^{C_i^j} - 1, f_k^{C_i^j})$ ***end for***
8: Train the MMN using $F_{s-1}^{C_i^j}, F_s^{C_i^j}$ and L_i ***end for***
9: Get the MMN parameters θ_s; ***end for***

3.1.4. Fusion Method

The motion of the object can be observed via changes in both appearance and semantics. Based on this, we follow a feature fusion strategy to combine spatial and temporal information. Given, $X_s \in \mathbb{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{T}}$, $X_t \in \mathbb{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{T}}$ are the extracted frame level spatial and temporal features, where H and W

are the height and width of the feature maps, T is the number of frames. Before the fusion operation, we have to reshape both features maps (spatial and temporal) into vectors, which can be given as:

$$X = [X_s, X_t] \tag{2}$$

Now, we perform a pixel-wise addition which is known as linear weighted fusion between X_s and X_t to compute a single feature map F.

$$F = w_s F_s \oplus w_t F_t \tag{3}$$

where, $X \in \mathbb{R}^{H \times W \times T}$, $\oplus$ is a matrix addition, w_s and w_t are weights of appearance and motion for spatial and temporal features maps, respectively. The weights are used to measure the significance of spatial and temporal features. After performing the fusion operation, we can define the new representative features as $x_{f,t}$ for the video clip. So, for the input video, a set of fused features ($x_{f,1}$, ... , $x_{f,t}$, ... , $x_{f,N}$) can be generated. Finally, we apply LSTM on these generated features to perform temporal encoding for human activity prediction.

3.2. Encoding and Activity Classification

This is the second and final part of our approach, which starts from detailed discussion on LSTM features and its architecture; then, we present the encoding and activity classification method.

3.2.1. Long Short-Term Memory (LSTM)

To analyze the hidden sequential patterns, it is natural choice to use RNN to encode the temporal structure of extracted sequential features. In video, visual information is represented in many frames which help in understanding the context of an action. RNN can interpret such sequences, but in cases of long term sequences, it usually forgets the earlier input sequence. LSTM has been designed to mitigate the vanishing problem and to learn long-term contextual information from temporal sequences. LSTM is a kind of recurrent network which can capture long-term dynamics, and which preserves sequence information over time. In addition, the LSTM gradient does not tend to vanish when trained with back propagation through time. Its special structure with input, output and, control gates control long-term sequence pattern identification. The gates are adjusted by a sigmoid unit that learns during training when to open and close. We adopt LSTM for encoding and decoding to recognize human actions.

The architecture of a LSTM unit is depicted in Figure 5. x_t, c_t, h_t and y_t stand for input vector, cell state, hidden state and output at the t-th state, respectively. The output y_t depends on hidden state h_t, while h_t depends on not only the cell state c_t, but also on its previous state. Intuitively, the LSTM has the capacity to read and write to its internal memory, and hence, to maintain and process information over time. The LSTM neuron contains an input gate i_t, a memory cell c_t, a forget gate f_t, and an output gate o_t. At each time step t, it can choose to write, read or reset the memory cell through these three gates. This strategy helps LSTM to access and memorize information in many steps. Equations (4)–(9) demonstrate the operation of temporal modelling performed in LSTM unit.

W and b are the parameters of the LSTM known as weights of the input vector and bias term. S means a sigmoid function, *tanh* is the activation function and $\otimes$ is the element-wise multiplication. The cell state and output are computed step by step to extract long-term dependencies. The input to LSTM is x_t, which is the feature vector. A forget gate is used to clear the information from the memory unit, and an output gate keeps the information about the upcoming step. We also have g_t, which is computed from the input of the current frame and state of the previous state h_{t-1}. The hidden state of LSTM step is computed by using a *tanh* activation function and memory cell c_t.

$$i_t = S(w_{xi}\, x_t + W_{hi} h_{t-1} + b_i) \tag{4}$$

$$f_t = S\Big(w_{xf}\, x_t + W_{hf} h_{t-1} + b_f\Big) \tag{5}$$

$$o_t = S(w_{xo}\, x_t + W_{ho} h_{t-1} + b_o) \tag{6}$$

$$g_t = tanh(w_{xc}\, x_t + W_{hc} h_{t-1} + b_c) \tag{7}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes g_t \tag{8}$$

$$z_t = \; h_t = o_t \otimes tanh\,(c_t) \tag{9}$$

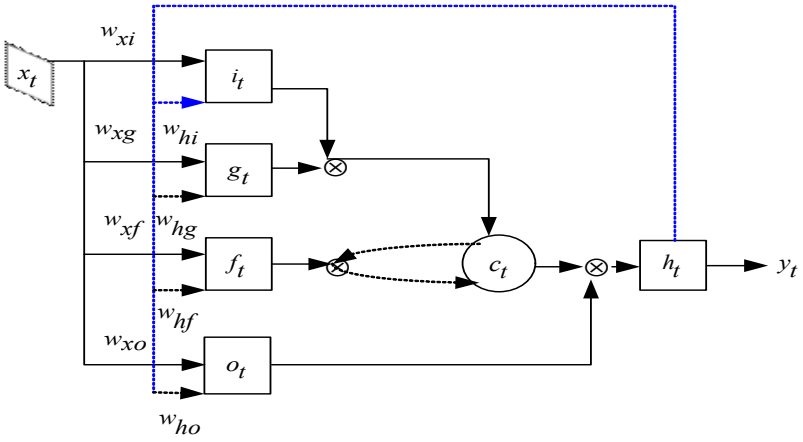

Figure 5. The architecture of LSTM Unit.

3.2.2. Encoding and Classification Process by LSTM

The generated fused features $x_{f,t}$ are fed into LSTM as inputs to conduct encoding and decoding for activity prediction. LSTM can be jointly trained, and our proposed model provides a trainable platform which is ideal for large-scale cognitive intelligence. The unique feature of LSTM is that it processes variable length inputs and produces high-level variable length predictions (Output). As shown in Figure 6, LSTM consists of an encoder and decoder; the encoder transforms input data x_t to a corresponding activation h. The decoder in the output layer is trained to reconstruct an approximation y of the input from activation h.

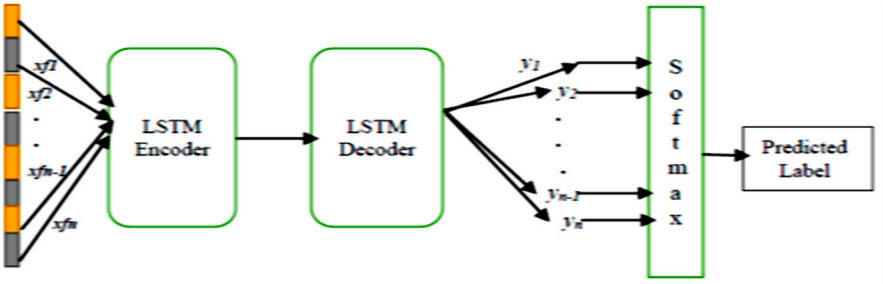

Figure 6. The framework of LSTM (Encoder-Decoder).

In general, the LSTM model has parameters i.e., W and b, which denotes the weights and the biases of input layer and the hidden layer respectively, generates an output z_t of given input x_t and a previous hidden state at time step $t-1$ i.e., h_{t-1}, and also updates the current hidden state h_t.

$$z_t = S(W_1 x_t + b_1) \tag{10}$$

The next step is decoding, which is similar as the encoding step given in Equation (10), where W_2 and b_2 denotes the weights and biases of the hidden layer and the output layer.

$$y_t = (W_2 z_t + b_2) \tag{11}$$

The final single label prediction for a video can be produced by using softmax classifier. A Softmax layer can be utilized to achieve the M-way class scores for a given video sequence. This single prediction can be achieved by averaging the label probabilities, which is the output of our decoder, and can be represented by the Equation (11).

$$P(y^{q}_{(t)} = 1) = softmax(y_t) = softmax(W_{zt} + b_t) \tag{12}$$

where W and b_t are the trained parameters of the LSTM model, $q \in Q$ is the prediction and t is the current time step.

4. Experiments

We conduct several experiments to validate the effectiveness of our system. Three well-known benchmark human action datasets, UCF101 [32], HDMB51 [33], and UCF Sports [34], have been used. The description of datasets with their validation schemes, experimental setup, results and comparative analysis are presented in subsequent sections.

4.1. Datasets

The UCF101 dataset is the extension of UCF50; it contains 101 different action categories. Each action category consists of at least 100 video. There are 13,320 video clips in total. Most of the video clips are realistic, clean and user-uploaded videos with cluttered background, illumination and camera motion. The dataset is divided into a training set containing 9.5 K videos and testing set containing 3.8 K videos. We adopt the evaluation scheme of the THUMOS13 challenge [35] and follow the three testing/training split for performance evaluation by reporting average recognition accuracy over these three splits.

The HDMB51 dataset comprises of variety of realistic videos collected from YouTube and Google video. There are 6766 manually annotated video clips of 51 different action classes and each action class containing about 100 video clips. For experimental setting, we follow the original evaluation guidelines using three-test splits, and each split with an action class has 30 sequences for testing and 70 sequences for training. The average accuracy over these three splits is used to measure the final performance.

The UCF sports dataset encompasses 150 videos from 10 action classes. These videos were recorded in real sports environments, taken from different television channels. This dataset exhibiting the occlusion, illumination conditions and variations in background make it a complex and challenging dataset. The average accuracy is used to measure the final performance.

Some sample frames from three datasets are given in Figure 7.

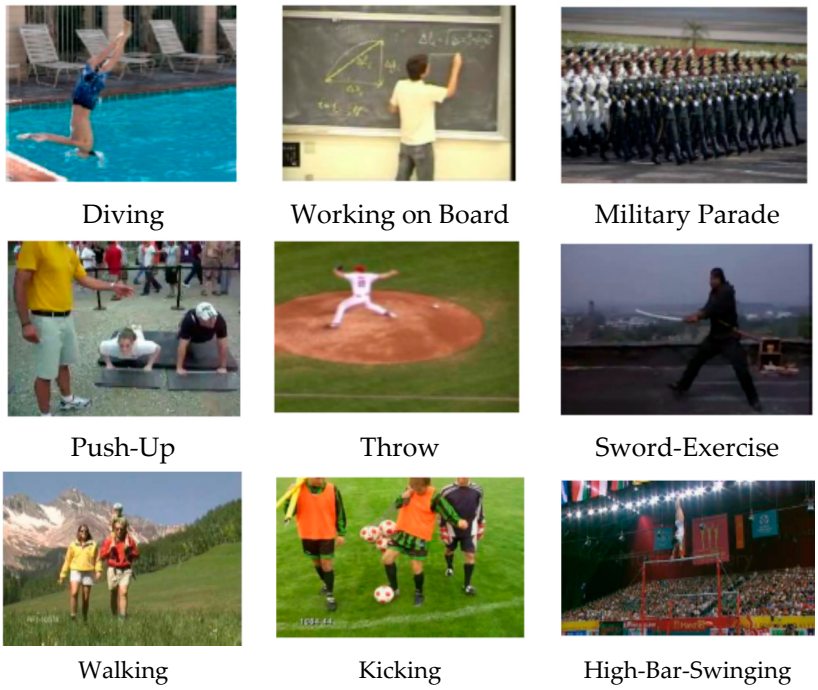

Figure 7. Sample frames from UCF101 (**first row**), HDMB51 (**second row**) and UCF Sports (**third row**).

4.2. Experimental Setup and Implementation Details

As UCF101 is the largest dataset among the three datasets, we use it to train the C3D model initially, and then transfer it to the learnt model to HMDB51 and UCF sports for feature extraction. RGB clips are resized to have a frame size of 128 $\times$ 171. On training, we randomly crop input clips into 16 $\times$ 112 $\times$ 112 crops. We also horizontally flip them with 55% probability. We fine-tune the model parameters on the UCF101 dataset, where the initial learning rate is set as 0.003, which is divided by 2 every 150 K iterations. The optimization is stopped at 1.9 M iterations.

Since the network can only handle video frames, the videos need to be processed as video frames. Therefore, we extract a constant number video frames per second, which improves the generalization performance of the network. For the network to better capture the changes in the action, we extract two frames per second (fps) and loop the extracted frames, and fill up to 16 frames per video.

4.3. Results and Comparison Analysis

We conduct extensive experiments to evaluate the performance of our proposed method. In this section, we presented relevant experimental results and performance analysis.

4.3.1. Effect of Different Feature Fusion Techniques

In this section, we analyze the effect of different early fusion methods such as element-wise sum, concatenation, element-wise max and linear weighted fusion on our proposed framework. We show the recognition accuracy for UCF Sports dataset and also each split of UCF 101 dataset, and the average recognition accuracy over the three splits. The results are reported in Table 2. We observe that linear weighted fusion enhances the recognition accuracy of our approach by a fair margin, compared to other fusion methods. This enhancement may be due to the fact that the linear weighted fusion method efficiently fuses spatial and motion features. Therefore, we choose the linear weighted fusion method as our fusion scheme to fuse spatio-temporal features.

Table 2. Effect of different earlier fusion methods on our model. The accuracy (%) is computed on a UCF Sports dataset, and all three splits and their average on UCF101.

Fusion Method	UCF Sports	Split 1 (UCF 101)	Split 2 (UCF 101)	Split 3 (UCF 101)	Average (UCF 101)
Element-wise max	91.8	90.5	89.9	89.6	90.0
Element-wise sum	92.1	90.8	90.1	89.9	90.2
Concatenation	92.8	91.2	90.5	91.0	90.9
Linear weighted	93.9	91.6	90.9	91.7	91.4

4.3.2. Class-Wise Accuracy for Activity Recognition

This section computes the class-wise accuracy for action recognition. We investigate the recognition accuracy of our method by making a confusion matrix and considering 10 action classes of the UCF Sports dataset. Table 3 demonstrates the accuracy of each action category, the x-axis denotes the predicted labels and the y-axis represents the ground truth labels. The intensity of the true score is high (diagonal) for each category, and our method achieves 94% for all 10 classes. It is interesting to note that some of categories with similar actions are more easily confused with each other, such as golf swing, kicking, running, swing bench and walking; these categories interfere with each other and yield low scores. A possible reason for this is the similarity of the features and representations among actions. In addition, the number of training samples is too small, so the result is confusing and misclassification occurs. The confusion matrix of HMDB51 dataset is shown in Figure 8, which is well diagonalized. However, some categories are easily misclassified; nonetheless, our proposed approach still performs well with most action categories.

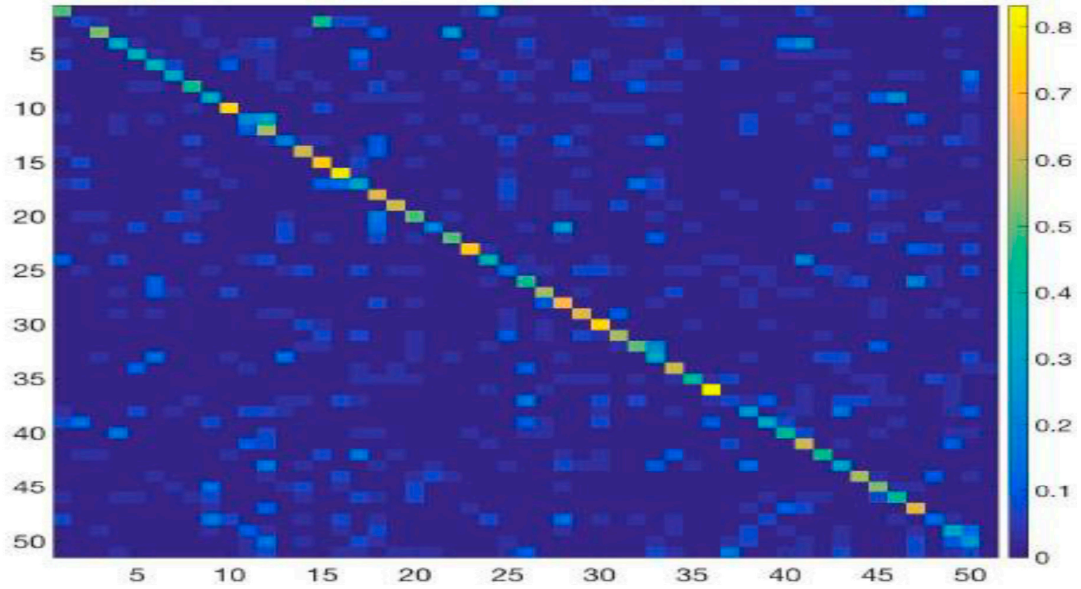

Figure 8. Confusion matrix on the HMDB51 dataset using our model.

Table 3. Confusion matric of UCF sports dataset.

Categories	Diving	Golf -Swing	Kicking	Lifting	Horse Riding	Running	Skate Boarding	Swing Bench	Swing-Side	Walking
Diving	1.00	0	0	0	0	0	0	0	0	0
Golf-Swing	0	0.91	0.07	0	0	0	0	0.02	0	0
Kicking	0	0.06	0.94	0	0	0	0	0	0	0
Lifting	0	0	0	0.95	0	0	0	0	0	0
Riding Horse	0	0	0	0	0.90	0	0	0	0	0
Running	0	0.06	0.01	0	0.01	0.91	0	0	0	0.01
Skateboarding	0	0	0	0	0	0	0.93	0	0	0
Swing Bench	0	0	0	0	0	0	0	1.00	0	0
Swing Side	0	0.01	0	0	0	0	0	0	0.99	0
Walking	0	0.07	0	0	0	0	0	0.04	0	0.89
Average Accuracy										0.94

4.3.3. Comparison to the State-Of-The-Art Methods

In this section, we further verify the effectiveness of our model, and compare our proposed approach to different existing state-of-the-art Human Action Recognition approaches on UCF101 and HDMB51 benchmark datasets. The comparison of results is reported in Table 4. We organize these baseline methods into different categories with respect to the type of features and network being used, including traditional, deep-learned features, very deep-learned features and hybrid features.

Table 4. Comparison to the state-of-the-art methods.

Modality	Method	Year	UCF101	HDMB51
Traditional	iDT+fisher vector [10]	2013	84.7	57.2
	Ordered Trajectory [36]	2015	72.8	47.3
	MPR [37]	2015	-	65.5
	MoFAP [38]	2016	88.3	61.7
	Trajectory Rejection [39]	2016	85.7	58.9
Deep	Two-Stream [16]	2013	88.9	59.4
	FSTCN [40]	2015	88.1	59.1
	EMV-CNN [41]	2016	86.4	-
	DANN [42]	2016	89.2	63.3
	Dynamic Images [11]	2016	89.1	65.2
	LTC-CNN [43]	2018	92.7	67.2
Very deep	C3D [13]	2015	85.2	-
	LSTM [27]	2015	88.6	-
	LRCN [25]	2015	82.9	-
	VideoLSTM [44]	2016	89.2	56.4
	3D Convolution [14]	2016	91.8	64.6
	STPP-LSTM [45]	2017	91.6	69.0
	FCNs-16 [46]	2017	90.5	63.4
	Hidden-Two-stream [47]	2017	90.3	58.9
	Multi-LSTM [48]	2018	90.8	-
Hybrid-Model	TDD-iDT [19]	2015	91.5	65.9
	C3D-iDT [13]	2015	90.4	-
	TSN [17]	2016	94.2	69.4
	3D conv + iDT [14]	2016	93.5	69.2
	SCLSTM [49]	2017	84.0	55.1
	LTC-iDT [43]	2018	92.7	67.2
Ours	LSTM–3D ConvNet	-	92.9	70.1

Compared to traditional methods, our model performs the best by 4.5% on both datasets, Compared with RNN-based methods such as (LRCN) [25] and (LSTM) [27], our model outperforms these two methods by 4.3% and 10% on UCF101 datasets respectively. Different experiments indicated that our approach possesses higher discriminative power, even using fewer parameters. It can be also seen that some methods with both features such as TSN [17] and 3D conv—iDT [14] lead to a performance gain by a minimal margin on the UCF101 dataset. We can explain the decrease in prediction rate by fact that this dataset contains action classes with cluttered backgrounds and illumination changes, and TSN is pre-trained on the large-scale ImageNet dataset, which provides large scale size and diversity. Our approach is based on C3D, which is pre-trained on the UCF101 dataset. However, our introduced method outperformed the 3D conv—iDT by 0.9% and the TSN method by 0.7% on the HDMB51 dataset, and showed the highest recognition rate on small-scale datasets. A possible reason for this higher recognition accuracy is that our model is based on a hybrid deep learning model, and the introduction of LSTM temporally works well by capturing the long-term dependencies and boosting the recognition accuracy for complex action categories in the HDMB51 dataset. We can conclude that a combination of LSTM with a 3D convolutional network for the spatiotemporal stream achieves better results and obtains recognition rates of 92.9% and 70.1%

on UCF101 and HDMB51 datasets respectively. This shows that there is a degree of complimentary between LSTM and convolutional neural network.

5. Conclusions

In this paper, we proposed an action recognition framework by utilizing frame-level deep features of the 3D-CNN and processing it through LSTM. First, we introduced a 3Dconv-based model MMN and its iterative training method to integrate the discriminative information of a video into motion maps. Three-dimensional convolutional components extract compact and efficient spatiotemporal features from the input video in the form of feature maps. Moreover, we design a linear weighted fusion method to effectively fuse spatial and temporal feature maps. Finally, we adopt LSTM encoder/decoder to obtain video level representations to conduct video classification. According to the experimental results, our model takes the complementary information contained in multiple features (both spatial and motion features). It is also proof that the motion maps generated by our model intuitively integrate the dynamic information in an efficient manner, and that they retain more discriminative aspects. Moreover, our fusion method makes the features more detailed and specific. To verify the effectiveness of our framework, extensive experiments have been carried out on benchmark datasets, and the obtained results showed that our approach achieves promising performance.

Author Contributions: For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used "Conceptualization, S.A. and J.W.; Methodology, S.A. and Z.F.; Software, T.U.H.; Validation, S.A., T.U.H. and J.W.; Formal Analysis, S.A. and T.U.H.; Investigation, S.A. and T.U.H.; Resources, J.W.; Data Curation, T.U.H.; Writing-Original Draft Preparation, S.A.; Writing-Review & Editing, S.A. and Z.F.; Visualization, S.A.; Supervision, J.W.; Project Administration, J.W. and Z.F.

Acknowledgments: The research was supported by Research Institute of Communication Technology (RICT) in Beijing Institute of Technology.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Poppe, R. A survey on vision-based human action recognition. *Image Vis. Comput.* **2010**, *28*, 976–990. [CrossRef]
2. Laptev, I. On space-time interest points. *Int. J. Comput. Vis.* **2005**, *64*, 107–123. [CrossRef]
3. Willems, G.; Tuytelaars, T.; Gool, L. An efficient dense and scale-invariant spatio-temporal interest point detector. In Proceedings of the 10th European Conference on Computer Vision, Marseille, France, 12–18 October 2008; pp. 650–663.
4. Yeffet, Y.; Wolf, L. Local trinary patterns for human action recognition. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 492–497.
5. Dollr, P.; Rabaud, V.; Cottrell, G.; Belongie, S. Behavior recognition via sparse spatio-temporal features. In Proceedings of the 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Beijing, China, 15–16 October 2005; pp. 65–72.
6. Scovanner, P.; Ali, S.; Shah, M. A 3-dimensional SIFT descriptor and its application to action recognition. In Proceedings of the 15th ACM international conference on Multimedia, Augsburg, Germany, 25–29 September 2007; pp. 357–360.
7. Matikanen, P.; Hebert, M.; Sukthankar, R. Trajectons: Action recognition through the motion analysis of tracked features. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, Kyoto, Japan, 27 September–4 October 2009; pp. 514–521.
8. Sun, J.; Wu, X.; Yan, S.; Cheong, L.; Chua, T.S.; Li, J. Hierarchical spatio-temporal context modeling for action recognition. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 2004–2011.
9. Wang, H.; Klser, A.; Schmid, C.; Liu, C.L. Dense trajectories and motion boundary descriptors for action recognition. *Int. J. Comput. Visi.* **2013**, *103*, 60–79. [CrossRef]

10. Wang, H.; Schmid, C. Action recognition with improved trajectories. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, NSW, Australia, 1–8 December 2013; pp. 3551–3558.
11. Bilen, H.; Fernando, B.; Gavves, E.; Vedaldi, A.; Gould, S. Dynamic image networks for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3034–3042.
12. Chen, C.; Liu, K.; Kehtarnavaz, N. Real-time human action recognition based on depth motion maps. *J. Real-Time Image Process.* **2016**, *12*, 155–163. [CrossRef]
13. Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning spatiotemporal features with 3d convolutional networks. *arXiv* **2015**, arXiv:1412.0767.
14. Ji, S.; Xu, W.; Yang, M.; Yu, K. 3D convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 221–231. [CrossRef] [PubMed]
15. Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; Li, F.F. Large-scale video classification with convolutional neural networks. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014.
16. Simonyan, K.; Zisserman, A. Two-stream convolutional networks for action recognition in videos. *arXiv* **2014**, arXiv:1406.2199.
17. Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; Van Gool, L. Temporal segment networks: Towards good practices for deep action recognition. *arXiv* **2016**, arXiv:1608.00859.
18. Feichtenhofer, C.; Pinz, A.; Zisserman, A. Convolutional two-stream network fusion for video action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1933–1941.
19. Wang, L.; Qiao, Y.; Tang, X. Action recognition with trajectory-pooled deep-convolutional descriptors. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.
20. Lu, X.; Yao, H.; Zhao, S. Action recognition with multi-scale trajectory-pooled 3D convolutional descriptors. *Trans. Multimedia Tools Appl.* **2017**, 1–17. [CrossRef]
21. Taylor, G.; Fergus, R.; LeCun, Y.; Bregler, C. Convolutional learning of spatiotemporal features. In Proceedings of the 11th European Conference on Computer Vision, Heraklion, Crete, Greece, 5–11 September 2010; pp. 140–153.
22. Fernando, B.; Gavves, E.; Oramas, J.M.; Ghodrati, A.; Tuytelaars, T. Modeling video evolution for action recognition. In Proceedings of the 28th IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5378–5387.
23. Perronnin, F.; S´anchez, J.; Mensink, T. Improving the fisher kernel for large-scale image classification. In Proceedings of the 11th European Conference on Computer Vision, Heraklion, Crete, Greece, 5–11 September 2010; pp. 143–156.
24. Zaremba, W.; Sutskever, I.; Vinyals, O. Recurrent neural network regularization. *arXiv* **2014**, arXiv:1409.2329.
25. Donahue, J.; Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 677–691. [CrossRef] [PubMed]
26. Veeriah, V.; Zhuang, N.; Qi, G.J. Differential recurrent neural networks for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Santiago, Chile, 7–13 December 2015; pp. 4041–4049.
27. Yue-Hei, J.; Hausknecht, M.; Vijayanarasimhan, S. Beyond short snippets: Deep networks for video classification. In Proceedings of the 28th IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4694–4702.
28. Wu, Z.; Wang, X.; Jiang, Y. Modelling spatial-temporal clues in a hybrid deep learning framework for video classification. In Proceedings of the 23rd ACM international conference on Multimedia, Brisbane, Australia, 26–30 October 2015; pp. 461–470.
29. Ji-Hae, K.; Gwang-soo, H.; Byung-Gyu, K.; Debi, D. deepGesture: Deep learning-based gesture recognition scheme using motion sensors. *Displays* **2018**, *55*, 38–45.
30. Srivastava, N.; Mansimov, E.; Salakhutdinov, R. Unsupervised Learning of Video Representations using LSTMs. *arXiv* **2015**, arXiv:1502.04681.

31. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
32. Soomro, K.; Zamir, A.R.; Shah, M. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv* **2012**, arXiv:1212.0402.
33. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; Serre, T. Hmdb: A large video database for human motion recognition. In Proceedings of the IEEE 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2556–2563.
34. Rodriguez, M.D.; Ahmed, J.; Shah, M. Action MACH a spatiotemporal maximum average correlation height filter for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern (CVPR), Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
35. Jiang, Y.G.; Liu, J.; Zamir, R.; Laptev, I.; Piccardi, M.; Shah, M.; Sukthankar, R. THUMOS challenge: Action Recognition with a Large Number of Classes. The First International Workshop on Action Recognition with a Large Number of Classes, in Conjunction with ICCV'13, Sydney, Australia. 2013. Available online: http://crcv.ucf.edu/ICCV13-Action-Workshop/ (accessed on 28 January 2019).
36. Murthy, V.R.; Goecke, R. Ordered trajectories for large scale human action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 25–27 June 2013; pp. 412–419.
37. Ni, B.; Moulin, P.; Yang, X. Motion part regularization: Improving action recognition via trajectory selection. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3698–3706.
38. Wang, L.; Qiao, Y.; Tang, X. Mofap: A multi-level representation for action recognition. *Int. J. Comput. Vis.* **2016**, *119*, 119–254. [CrossRef]
39. Seo, J.; Kim, H.; Ro, Y.M. Effective and efficient human action recognition using dynamic frame skipping and trajectory rejection. *J. Image Vis. Comput.* **2017**, *58*, 76–85. [CrossRef]
40. Sun, L.; Jia, K.; Shi, B.E. Human action recognition using factorized spatio-temporal convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 4597–4605.
41. Zhang, B.; Wang, L.; Wang, Z.Y. Real-time action recognition with enhanced motion vector CNNs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2718–2726.
42. Wang, J.; Wang, W.; Wang, R. Deep alternative neural network: Exploring contexts as early as possible for action recognition. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS), Barcelona, Spain, 4–9 December 2016; pp. 811–819.
43. Varol, G.; Laptev, I.; Schmid, C. Long-term temporal convolutions for action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1510–1517. [CrossRef] [PubMed]
44. Li, Z.; Gavves, E.; Jain, M. VideoLSTM convolves, attends and flows for action recognition. *Comput. Vis. Image Underst.* **2016**, *166*, 41–50. [CrossRef]
45. Wang, X.; Gao, L.; Wang, P.; Liu, X. Two-stream 3D convNet Fusion for Action Recognition in Videos with Arbitrary Size and Length. *IEEE Trans. Multimedia* **2017**, *20*, 1–11.
46. Yu, S.; Cheng, Y.; Xie, L. Fully convolutional networks for action recognition. *Inst. Eng. Technol. Comput. Vis.* **2017**, *11*, 744–749. [CrossRef]
47. Zhu, Y.; Lan, Z.; Newsam, S. Hidden two-stream convolutional networks for action recognition. *arXiv* **2017**, arXiv:1704.00389.
48. Yeung, S.; Russakovsky, O.; Jin, N. Every moment counts: Dense detailed labelling of actions in complex videos. *Int. J. Comput. Vis.* **2018**, *126*, 375–389. [CrossRef]
49. Wang, X.; Gao, L.; Song, J.; Shen, H. Beyond frame-level CNN: Saliency-aware 3-D CNN with LSTM for video action recognition. *IEEE Signal Process. Lett.* **2017**, *24*, 510–514. [CrossRef]