Sequential Video VLAD: Training the Aggregation Locally and Temporally

Youjiang Xu, Yahong Han, *Member, IEEE*, Richang Hong, *Member, IEEE*, and Qi Tian, *Fellow, IEEE*

***Abstract*—As characterizing videos simultaneously from spatial and temporal cues has been shown crucial for the video analysis, the combination of convolutional neural networks and recurrent neural networks, i.e., recurrent convolution networks (RCNs), should be a native framework for learning the spatio-temporal video features. In this paper, we develop a novel sequential vector of locally aggregated descriptor (VLAD) layer, named SeqVLAD, to combine a trainable VLAD encoding process and the RCNs architecture into a whole framework. In particular, sequential convolutional feature maps extracted from successive video frames are fed into the RCNs to learn soft spatio-temporal assignment parameters, so as to aggregate not only detailed spatial information in separate video frames but also fine motion information in successive video frames. Moreover, we improve the gated recurrent unit (GRU) of RCNs by sharing the input-to-hidden parameters and propose an improved GRU-RCN architecture named shared GRU-RCN (SGRU-RCN). Thus, our SGRU-RCN has a fewer parameters and a less possibility of overfitting. In experiments, we evaluate SeqVLAD with the tasks of video captioning and video action recognition. Experimental results on Microsoft Research Video Description Corpus, Montreal Video Annotation Dataset, UCF101, and HMDB51 demonstrate the effectiveness and good performance of our method.**

***Index Terms*—Video representation, deep learning, recurrent convolution networks, video captioning, action recognition.**

I. Introduction

As VIDEOS are spatio-temporal extension of images, how to incorporate the spatio-temporal information into the video representation has been shown crucial in different tasks of video analysis [1], [2]. Early methods, e.g., Dense Trajectories (DT) [3] or Improved Dense Trajectories (iDT) [4], rely on first hand-crafting spatio-temporal descriptors and then encoding them to form the final video representations via Vector of Locally Aggregated Descriptors (VLAD) [5] or Fisher Vector [6].

Along with the success in image classification [7], [8], and scene recognition [9]–[12], deep learned descriptors also show good performance in video analysis [1], [13], [14]. For example, Two-Stream [1] and Trajectory-Pooled Deep-Convolutional Descriptors (TDD) [13] successfully characterize action videos from spatial and temporal cues simultaneously. Recently, Recurrent Neural Networks (RNNs) has shown its good ability in understanding temporal sequences such as speech recognition [15]. Thus the combination of Convolutional Neural Networks (CNNs) and RNNs should be a native framework for learning the spatio-temporal video features. For example, the recent proposed Recurrent Convolution Networks (RCNs) [16], [17] first extract 2-dimensional features from independent video frames via CNNs and then feed them into a RNNs network to get the final video representations. However, the RCNs models focus more on global appearance changes but discard the fine motion information in the video. Towards this problem, several methods replace the fully-connected units with convolution operation in the recurrent unit and thus develop new RCNs units such as GRU-RCN [18] or similar frameworks such as ConvLSTM [19] and VideoLSTM [20].

Though different from the hand-crafted visual features, deep learned descriptors also resort to encoding methods to pool descriptors and form the final representation. For example, Xu *et al.* [21] proposed a CNNs based video descriptor learning method and utilized VLAD to capture information about the statistics of local descriptors, which are aggregated spatially over frames. Though aggregating local descriptors into a compact image representation has shown excellent performance in image retrieval [5], the process has two separate steps: first to learn image descriptor, and second to encode the descriptor via VLAD, which have yet been an end-to-end manner. A trainable VLAD layer, i.e., NetVLAD [22], was proposed to make the VLAD pooling differentiable in a CNNs framework, whose learnable parameters make the locally aggregated descriptors encode fine spatial information over images. More recently, Girdhar *et al.* [23] proposed an ActionVLAD which tries to aggregate appearance (RGB) and motion (optical flow) features at the same time. However, in ActionVLAD, each video descriptor is then assigned to one of the action words, while the method computes their sum inside each of the visual centers according to the assignment, which is computed individually for each frame and loses much temporal information in the sequential frames.

Manuscript received August 31, 2017; revised March 10, 2018 and May 11, 2018; accepted June 4, 2018. Date of publication June 12, 2018; date of current version July 9, 2018. This work was supported in part by NSFC under Grant U1509206, Grant 61472276, Grant 61472116, Grant 61722204, and Grant 61429201. The work of Y. Han was supported by the Tianjin Natural Science Foundation under Grant 15JCYBJC15400. The work of Q. Tian was supported in part by ARO under Grant W911NF-15-1-0290 and in part by the Faculty Research Gift Awards by the NEC Laboratories of America and Blippar. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jie Liang. *(Corresponding author: Yahong Han.)*

Y. Xu and Y. Han are with the School of Computer Science and Technology, Tianjin University, Tianjin 300350, China (e-mail: yjxu@tju.edu.cn; yahong@tju.edu.cn).

R. Hong is with the School of Computer and Information, Hefei University of Technology, Hefei 230009, China (e-mail: hongrc.hfut@gmail.com).

Q. Tian is with the Department of Computer Science, The University of Texas at San Antonio, San Antonio, TX 78249-1604 USA (e-mail: qitian@cs.utsa.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2018.2846664

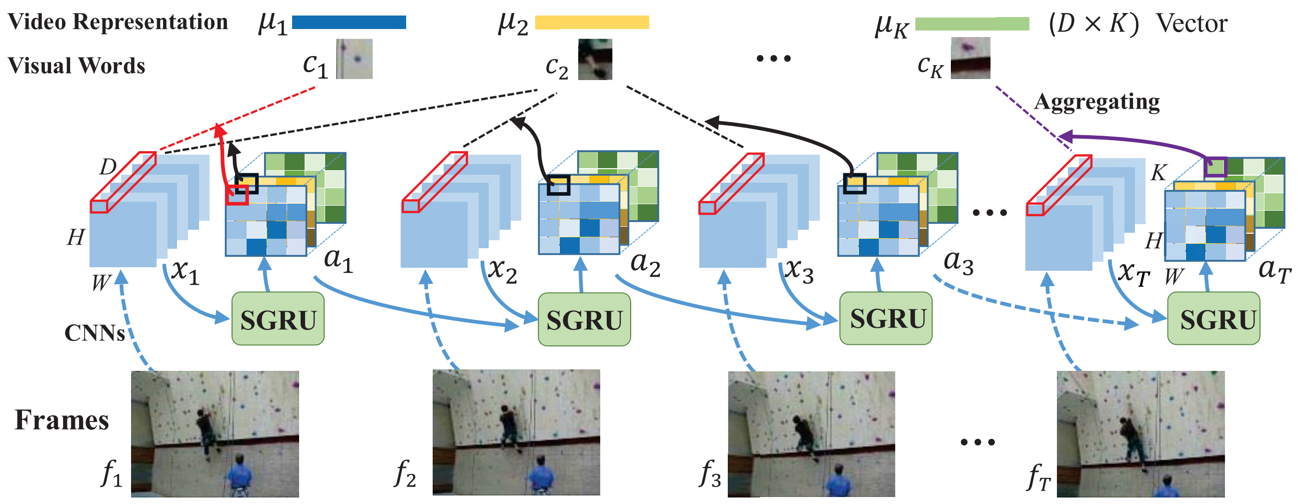


Fig. 1. The flowchart of the proposed Sequential Video VLAD. We first input successive video frames into a CNNs model to extract 3D feature maps with the size $D \times H \times W$. Then the sequential feature maps are fed into an improved RCNs unit (e.g., Shared GRU-RCN) to learn the soft spatio-temporal assignment parameters (with the size $K \times H \times W$). The assignment parameters could be taken as the weights for aggregating local descriptors ($H \times W$) to each of K visual centers.

Towards video analysis, in this paper, we target to develop a new framework of Sequential VLAD (SeqVLAD) to train the aggregation parameters both locally and temporally in an improved RCNs network. The goal is to make the video descriptor encoding learnable and aggregate detailed spatial information in separate video frames as well as fine motion information in successive video frames. The flowchart of SeqVLAD is illustrated in Fig. 1. Firstly, we extract sequential convolutional feature maps from successive video frames by CNNs. Then the learned feature maps are fed into the improved RCNs (e.g., Shared GRU-RCN) to learn soft spatio-temporal assignment parameters, which could be taken as the weights for aggregating local descriptors to certain visual centers. Thus we combine a trainable VLAD encoding process and the RCNs architecture, so as to aggregate not only detailed spatial information in localities of each frame but also temporal information in sequential video frames and then form the final representations.

The main contributions of this paper are summarized as follows:

- We develop a novel Sequential Video VLAD layer, named SeqVLAD, to combine a trainable VLAD encoding process and the RCNs architecture into a whole framework. The sequential convolutional feature maps extracted from successive video frames are fed into the RCNs to learn soft spatio-temporal assignment parameters, so as to aggregate not only detailed spatial information in separate video frames but also fine motion information in successive video frames.
- An improved Gated Recurrent Unit (GRU) of RCNs called Shared GRU-RCN (SGRU-RCN) is employed to learn the spatio-temporal assignments. As the SGRU-RCN has fewer parameters, our SeqVLAD has less probability of overfitting and could achieve better performance.
- We conduct extensive experiments on the task of both video captioning and video action recognition. Experimental results on benchmark datasets demonstrate the good performance of the proposed method.

The rest of this paper is organized as follows. In Section II, we briefly discuss the related works. In Section III, we introduce the method of Sequential Video VLAD in detail. In Section IV, experiments are conducted with tasks of video captioning and video action recognition. Experimental results and corresponding discussions show that our method is effective and has good performance. Finally, we conclude our paper in Section V.

II. Related Works

Most of the hand-crafted and learning-based video representations follow the similar practice of first devising spatio-temporal descriptors and then aggregating them using the encoding methods like Bag-of-Visual-Words [24], Fisher Vector [6], and VLAD [5].

Inspired by the success of local spatio-temporal descriptors like Dense Trajectories [3], representative deep learned video features [1], [13], [25]–[27] try to mimic in the networks the feature extraction process of hand-crafted video descriptors. Two Stream [1] extends the ConvNets to two tightly-coupled networks: spatial stream performs recognition from still video frames whilst the temporal stream performs recognition from motions by optical flow stacking. After extracting trajectories, TDD [13] conducts trajectory-constrained sampling and pooling over convolutional feature maps. Different from TDD which requires an encoding step to form the final video representations, C3D [25] develops a 3-dimensional filter to directly extract spatio-temporal feature from successive video frames. However, good performance of C3D might depend on the training over a large-scale dataset. Zhu *et al.* [26] proposed a key volume framework to identify the key volumes in the videos and achieved excellent performance on the task of action recognition. The temporal segment network (TSN) [27] is proposed to model the long-range temporal structure through the whole action video.

As discussed in the introduction, the combination of CNNs and RNNs, i.e., RCNs methods [16], [17], should be native frameworks for learning the spatio-temporal video features. However, RCNs methods focus more on global appearance transition but may discard some of the fine motion information. Thus, Ballas *et al.* [18] proposed a GRU-RCN architecture which leverages convolutional unit in RNNs architecture to tackle this problem. There are also several similar architectures such as ConvLSTM [19] and VideoLSTM [20]. However, stacking several GRU-RCN layers (Stacked GRU-RCN [18]) burdens the performance due to the increase of the network depth and thus the possibility of overfitting. Though different but along this line, Pan *et al.* [2] proposed a hierarchical RNNs architecture for learning video representations and applied it in video captioning. More recently, Zhu *et al.* [28] proposed a Multirate Visual Recurrent Model (MVRM) to deal with motion speed variance in the videos by encoding frames of a clip with different intervals.

The VLAD encoding [5] has achieved the excellent performance of in image classification [29] and place recognition [22]. The method in [29] leverages the semantic probabilities of local patches to learn the aggregation weights and construct the semantic codebook, which overcomes the difficulty of dictionary learning in VLAD, and produces more semantic and discriminative global representations.

In this paper, we aim to capture fine motion information from videos. Thus, inspired by NetVLAD [22] and RCNs architectures, we develop a novel Sequential VLAD (SeqVLAD) layer for encoding sequential 3-dimensional feature maps for final video representation. The SeqVLAD owns the merits of both VLAD encoding method and convolutional units in RNNs architecture, which can be trained via backpropagation. Moreover, we improve the GRU-RCN by sharing the input-to-hidden parameters and propose an improved GRU-RCN architecture named Shared GRU-RCN (SGRU-RCN). Our SGRU-RCN has fewer parameters and less possibility of overfitting.

III. The Proposed Method

In this section, we introduce the proposed method of Sequential VLAD (SeqVLAD). We first review the encoding methods of VLAD and NetVLAD. Then we describe how to integrate the trainable VLAD process into the Shared GRU-RCN (SGRU-RCN) architecture and thus develop the framework of Sequential VLAD. Finally, we apply our SeqVLAD on the task of both video captioning and video action recognition.

A. Review of VLAD and NetVLAD

VLAD [5] aggregates local image descriptors into a vector of certain dimension. It can be seen as a simplification of Fisher kernel representation. Different from Bag-of-Visual-Words [24] which counts the number of visual words, VLAD accumulates the differences between local descriptors and their nearest visual word. Suppose there are K visual words (centers) $C = \{c_1, c_2, \ldots, c_K\}$ generated by K-means, given a set of D dimensional local descriptors $X = \{x_1, x_2, \ldots, x_N\}$, the VLAD encoding method can be denoted as follows:

$$\mu_k = \sum_{d=1}^{N} a_k(x_d)(x_d - c_k), \tag{1}$$

where N is the number of descriptors. x_d and c_k are the d-th descriptor and k-th center, respectively. The assignment $a_k(x_d)$ denotes the relationship between descriptor x_d and k-th visual words c_k: $a_k(x_d) = 1$ if c_k is the nearest center to the descriptor x_d and $a_k(x_d) = 0$ otherwise. Then the VLAD concatenates μ_k over all K visual words as : $\mu = [\mu_1, \mu_2, \ldots, \mu_K]$. If the dimension of visual word is D, the dimension of final representation should be $K \times D$. Power and ℓ_2 normalization are also used to post-process the representation. VLAD encoding has shown excellent performance in image retrieval. However, as the hard assignment $a_k(x_d)$ is discontinuous and non-differentiable, it is hard to train VLAD in networks via backpropagation.

NetVLAD [22] was proposed to plug VLAD into CNN architecture and make the VLAD encode over the deep neural networks. Instead of the discontinuous hard assignment $a_k(x_d)$ of descriptor x_d to certain cluster center c_k, NetVLAD utilizes a differentiable and soft assignment to aggregate local descriptors to multiple clusters. Thus the aggregation parameters could be updated and trained via backpropagation. The formulation of soft assignment is as follows:

$$a_k(x_d) = \frac{e^{\mathbf{w}_k^T x_d + b_k}}{\sum_{k'} e^{\mathbf{w}_{k'}^T x_d + b_{k'}}}, \tag{2}$$

where x_d is a D dimensional vector and $\mathbf{w}_k = 2\alpha c_k$. Scalar $b_k = -\alpha\|c_k\|^2$ and α is a positive constant. Thus, the encoding would be:

$$\mu_k = \sum_{d=1}^{N} \frac{e^{\mathbf{w}_k^T x_d + b_k}}{\sum_{k'} e^{\mathbf{w}_{k'}^T x_d + b_{k'}}}(x_d - c_k). \tag{3}$$

As the spatial size of x_d and $\mathbf{w}_k$ could be taken as 1×1, Eq. (2) could be divided into two steps: (1) A convolution operation by a set of K filters $\{\mathbf{w}_k\}$ and biases $\{b_k\}$ over local descriptor x_d; (2) A $softmax$ function is used to obtain the final assignment. Thus it can learn the relationship between local descriptor x_d and certain centers c_k by training the parameters $\mathbf{w}_k, b_k$, and c_k via backpropagation.

Note that, as VLAD only aggregates local information of spatial content of images, it is not straightforward to extend NetVLAD to tasks of video analysis. As videos are spatio-temporal extensions of images, one possible solution might be first separately encoding successive frames and then forming the final video representation by a temporal averaging which loses much temporal information in the sequential frames, e.g., ActionVLAD [23]. On the other hand, since RCN appropriately integrates CNN into a sequential network architecture, besides a native framework for learning video features, it also exhibits a good potential to develop a trainable Sequential VLAD layer.

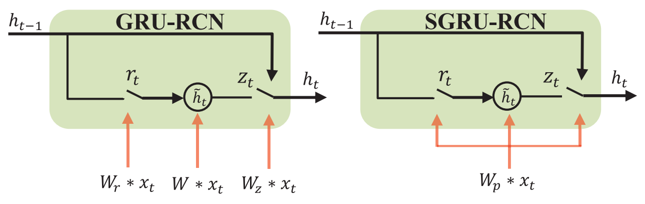


Fig. 2. Illustration of GRU-RCN and SGRU-RCN. The left one is the original GRU-RCN with three input-to-hidden parameters W_z, W_r, and W. The right one shows our SGRU-RCN which shares the input-to-hidden parameter W_p.

B. Sequential Video VLAD

1) Shared GRU-RCN: Though Recurrent Convolution Networks (RCN) [16], [17] successfully combines CNNs and RNNs and could be used to learn the spatio-temporal video features, it has been shown that RCN tends to focus more on global appearance changes and discard the fine motion information among successive video frames. Towards this issue, GRU-RCN [18] replaces the fully-connected unit with convolution operation in a recurrent unit. Thus both the inputs and outputs of GRU-RCN are 3-dimensional feature maps. A GRU-RCN unit is defined as follows:

$$z_t = \sigma(W_z * x_t + U_z * h_{t-1}), \tag{4}$$
$$r_t = \sigma(W_r * x_t + U_r * h_{t-1}), \tag{5}$$
$$\tilde{h}_t = tanh(W * x_t + U * (r_t \odot h_{t-1})), \tag{6}$$
$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t, \tag{7}$$

where $*$ denotes a convolution operation, $\odot$ is an element-wise multiplication, and σ is the sigmoid function. z_t is an update gate to control how much information from previous hidden states would be updated to the current state. r_t is a reset gate. The input at t-th time step x_t is 3-dimensional feature maps of $D \times H \times W$, where D, H, and W represent the number of channel, width, and height of corresponding feature maps, respectively. Output h_t of the GRU-RCN is also a 3-dimensional feature map and $h_t = (h_t(i, j))$, where $h_t(i, j)$ is a D-Dimensional vector corresponding to the location (i, j). Parameters W, W_z, W_r, U, U_z, and U_r are 2-dimensional convolution kernels.

In order to capture temporal patterns that occur at different spatial resolution, Stacked-GRU-RCN *et al.* [18] was proposed to leverage the hidden representations from layers of different depths. However, the increased depth from bottom-up connection not only burdens the performance but also tends to overfit.

In this paper, we propose to improve the GRU-RCN by sharing parameters of the input-to-hidden convolutional kernels, namely Shared GRU-RCN (SGRU-RCN), which significantly reduces the number of parameters and has less possibility of overfitting. In Fig. 2, we illustrate the difference between GRU-RCN and SGRU-RCN. A SGRU-RCN unit is defined as follows:

$$p_t = W_p * x_t, \tag{8}$$
$$z_t = \sigma(p_t + U_z * h_{t-1}), \tag{9}$$
$$r_t = \sigma(p_t + U_r * h_{t-1}), \tag{10}$$
$$\tilde{h}_t = tanh(p_t + U * (r_t \odot h_{t-1})), \tag{11}$$
$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t, \tag{12}$$

where W_p is a shared convolutional kernel for z_t, r_t, and $\tilde{h}_t$. In this formulation, the status of gates z_t, r_t, as well as the candidate hidden representation $\tilde{h}_t$ all depend on the hidden-to-hidden parameters U_z, U_r, U, and shared p_t. In the following, based on SGRU-RCN, we develop the Sequential VLAD.

2) Training Aggregation Locally and Temporally: As discussed above, SGRU-RCN preserves spatial topology by convolution operation and captures temporal information by the RNNs architecture. We thus utilize SGRU-RCN to train the aggregation locally and temporally because the output of SGRU-RCN can be considered as a native spatio-temporal assignment of VLAD encoding. Thus, we define the SGRU-RCN for VLAD encoding as follows:

$$p_t = W_p * x_t, \tag{13}$$
$$z_t = \sigma(p_t + U_z * a_{t-1}), \tag{14}$$
$$r_t = \sigma(p_t + U_r * a_{t-1}), \tag{15}$$
$$\tilde{a}_t = tanh(p_t + U * (r_t \odot a_{t-1})), \tag{16}$$
$$a_t = (1 - z_t)a_{t-1} + z_t\tilde{a}_t, \tag{17}$$

where the input x_t is a 3-dimensional feature map with $D \times H \times W$. W_p is with the shape $K \times D \times 1 \times 1$, which converts input x_t into the hidden space. The shape of hidden-to-hidden parameter U_z, U_r, and U is $K \times K \times 1 \times 1$, respectively. The output a_t is also a 3-dimensional feature map with the shape $K \times H \times W$, which can be taken as the spatio-temporal assignment weights for VLAD encoding. Thus $a_t = (a_t^k(i, j))$ and $a_t^k(i, j)$ denote the assignment weights of aggregating the descriptor at location (i, j) of t-th frame to the k-th visual word. Moreover, as the values of assignment range from 0 to 1, the output of SGRU-RCN will be followed a $softmax$ function, which is defined as follows:

$$softmax(a_t^k(i, j)) = \frac{e^{a_t^k(i,j)}}{\sum_{k'=1}^{K} e^{a_t^{k'}(i,j)}}, \tag{18}$$

where K denotes there are total K visual words.

To this end, we define the Sequential VLAD (SeqVLAD) encoding to be:

$$\mu_k = \sum_{t=1}^{T}\sum_{j=1}^{W}\sum_{i=1}^{H} a_t^k(i, j)(x_t(i, j) - c_k), \tag{19}$$

where $x_t(i, j) \in \mathbb{R}^D$ denotes the local descriptor at location (i, j). T denotes the length of input sequences and is set to 10 in this paper. From Eq. (19), we can see that, for the k-th visual center, we accumulate the differences between local descriptors and c_k both locally and over temporal sequences. Similar with VLAD encoding, SeqVLAD also concatenates μ_k over K visual words and thus the dimension of final representation should be $K \times D$.

To clarify the differences between the ActionVLAD and our proposed SeqVLAD, we illustrate the differences with Fig. 3. Furthermore, we illustrate the characteristics of different aggregation methods in the Table I and summarize our SeqVLAD as follows:

- Compared with NetVLAD and ActionVLAD, the SeqVLAD learns the aggregation parameters not only

TABLE I

The Characteristics of Different Aggregation Methods. We Set the Dimension of Input Feature Map to $1024 \times 7 \times 7$. The Number of Visual Centers is Set to $K = 64$. "# Parameters-Aggregation" Denotes the Number of Parameters in the Layer of Aggregation. "# Parameters-Classifier" Denotes The Number of Parameters in the Classifier. "# parameters-total" Denotes the Sum of "# Parameters" and "# Parameters-Classifier"

Method	Input	Local	Temporal	# parameters-aggregation	# parameters-classifier	# parameters-total
Two-Stream ConvNet [1]	image	-	-	-	1.0×10^5	1.0×10^5
TSN [27]	video	-	✓	-	5.2×10^4	5.2×10^4
GRU-RCN [18]	video	-	✓	2.1×10^7	2.6×10^4	2.1×10^7
NetVLAD [22]	image	✓	-	1.3×10^5	3.3×10^6	3.4×10^6
ActionVLAD [23]	video	✓	-	1.3×10^5	3.3×10^6	3.4×10^6
SeqVLAD (GRU-RCN)	video	✓	✓	2.7×10^5	3.3×10^6	3.6×10^6
SeqVLAD (SGRU-RCN)	video	✓	✓	1.4×10^5	3.3×10^6	3.4×10^6

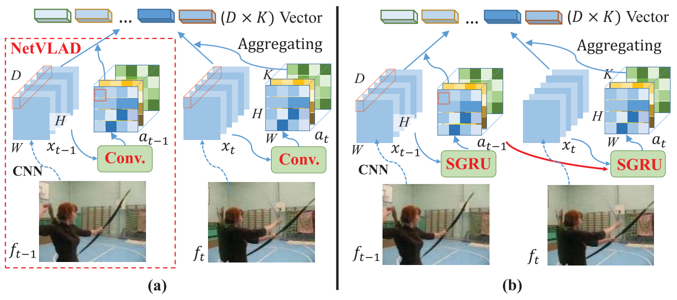


Fig. 3. The comparison of the ActionVLAD and SeqVLAD. ActionVLAD computes the assignment with the feature of current frame individually which ignores the temporal relationship of the sequential frames. Combining the feature of current frame and the assignment of last frame, SeqVLAD obtains the video representation with spatial and temporal information.

from the locally spatial content of each frame but also from the temporal information of successive frames. Note that the ActionVLAD can take successive video frames as inputs. However, it utilizes the temporal information in the sequential frames of the video in a relatively simple way. The method simply computes the sum of the aggregation results of each individual frame, which ignore the temporal relationship of the sequential frames. Note that the aggregation procedure of each frame in ActionVLAD method is indeed a NetVLAD encoding procedure. See Fig. 3 (a).

- Compared with the methods of NetVLAD, ActionVLAD, and GRU-RCN, the proposed SeqVLAD contains fewer parameters by SGRU-RCN and thus may be easy to train and less possible to overfit. Note that the GRU-RCN could be seen as a method that contains only one center and all values in it are zero.
- Combining trainable VLAD and RCN, we have good potential to obtain a discriminative video representation with fine and rich spatio-temporal information encoded. Note that our model could be easily added on the top of CNNs. Similar with ActionVLAD, we could combine SeqVLAD layer and CNN architecture into a whole framework and could be trained end-to-end. More details can be seen in Section IV-B.
- Compared to the methods of Two-Stream ConvNet [1] and TSN [27], the proposed SeqVLAD method could utilize both local information and temporal information contained in the videos to form a discriminative video representation. However, one drawback of the VLAD aggregation method is that it will produce a high dimensional video representation, which leads to more parameters in the classifier layer. But note that compared with other encoding methods, our SeqVLAD could produce a video representation contains more temporal information with similar number parameters.

C. Video Captioning

In this section, we introduce how we apply our SeqVLAD method to the task of video captioning. Following the common practice, we choose the general encoder-decoder framework. In this framework, the encoder maps video sequences to feature vectors and then the video captions are generated by the decoder. As we know that both the encoder module and the decoder module are important for generating video descriptions. And thus we employ our proposed SeqVLAD as the encoder and a one-layer LSTM (the simplest decoder module) as the decoder for generating video captions. In addition, we utilize the attention mechanism the same as [30] for fair comparison.

To optimize all the parameters θ in the caption model over all training data, we maximize the log-likelihood:

$$\max_{\theta} \sum_{t=1}^{T} \log Pr(y_t | z, y_{t-1}; \theta), \tag{20}$$

where y_t denotes the t-th word along the time and z is the feature vector output by the encoder of SeqVLAD.

D. Video Action Recognition

As both the video representation (the encoder) and the linguistic model (the decoder) may impact the performance of video captioning, we furthermore validate our proposed SeqVLAD on the task of video action recognition which purely depends on the video representation. Given T successive video frames, we first extract $D \times H \times W$ feature maps from a convolutional layer (or a pooling layer) of CNNs architecture for each video frame. Thus we can easily obtain the $K \times D$ dimensional video representation by feeding the feature maps into our proposed SeqVLAD. To evaluate the effectiveness of our proposed SeqVLAD method, the video representation followed a dropout layer is fed into a fully connected layer with a softmax activation. All of the model parameters are optimized by minimizing the cross-entropy loss.

For performing video action recognition, there are several different kinds of modalities as model inputs, such as still RGB images, stacked optical flow fields, and warped optical flow fields. For common practice [23] and [27], we choose still RGB images and stacked optical flow as inputs of our SeqVLAD.

IV. Experiments

In this section, we evaluate the proposed SeqVLAD on the tasks of both video captioning and video action recognition. We conduct experiments on four benchmark datasets of MSVD, M-VAD, UCF101 and HMDB51, respectively. We also compare our methods with various state-of-the-art methods.

A. Sequential VLAD for Video Captioning

1) Dataset: For evaluating SeqVLAD on the task of video captioning, we conduct experiments on the dataset of both Microsoft Research Video Description Corpus (MSVD) [31] and Montreal Video Annotation Dataset (M-VAD) [32].

MSVD is widely used in video captioning. This dataset contains 1,970 videos clips with several natural language descriptions per video clip. There are about 80,000 video-description pairs in this dataset. For fair comparison, we utilize the same splits as [33] and [28]. The number of videos for training, validation, and testing is 1,200, 100, and 670, respectively.

M-VAD is a large-scale movie description dataset. The dataset consists of around 49,000 movie clips which are extracted from 92 DVD movies. There is a single description for each movie clip. Following [32] and [34], we also utilize the standard splits, which consists of 36,921 for training, 4,651 for validation, and 4,951 for testing, to evaluate our SeqVLAD.

2) Experimental Setup: For the evaluation of video captioning, commonly used metrics are BLEU [35], METEOR [36], ROUGE-L [37], and CIDEr [38]. Following previous empirical studies in [2] and [18], we use METEOR, CIDEr, and BLEU@4 as our metrics for evaluation and calculate the scores of each metric on the MS-COCO evaluation server [39].

We utilize the same parameters setting for both datasets of MSVD and M-VAD. For the encoder process, we first use ConvNets to extract convolutional features of each frame, which are 3-Dimensional feature maps. Then we fed these feature maps into the SeqVLAD encoder to obtain the SeqVLAD representation. In this paper, we use two different ConvNets, i.e., GoogleNet [7] and ResNet-200 [40], to extract convolutional features. As for the one-layer decoder, the hidden size of LSTM, the feature embedding size and the word embedding size are set to 512. As the length of sentences varies in different video descriptions, we add tags of <BOS> and <EOS> to the beginning and ending of sentences, respectively. In the training process, the codebook is initialized randomly and we train the aggregation weights and the codebook together. The size of mini-batch is set to 64. We utilize Adam to minimize the negative log-likelihood loss and optimize all the parameters in the framework. The learning rate is set to 2×10^{-4}. The attention size is set to 100. In testing stage, we adopt the beam search strategy and set the beam size to 5.

TABLE II
Evaluation of SeqVLAD on the MSVD Dataset. We Utilize ResNet-200 to Extract Features of Video Frames. "# Centers" Denotes the Number of Visual Centers

# Centers	METEOR	BLEU@4	CIDEr
SGRU-RCN+mean pooling			
	33.80	50.88	76.53
NetVLAD+sum pooling			
16	33.05	49.61	73.49
32	33.20	49.36	72.92
64	32.55	47.11	73.32
SeqVLAD with GRU-RCN			
16	34.06	50.24	83.02
32	34.49	53.77	82.35
64	33.74	50.04	83.06
SeqVLAD with SGRU-RCN			
16	34.19	50.46	83.41
32	34.61	**54.57**	83.85
64	**35.15**	51.00	**86.04**
Bi-direction SeqVLAD with SGRU-RCN			
16	34.96	52.67	83.79
32	34.89	52.47	85.56
64	**35.17**	49.86	84.42

3) Experiments on the MSVD Dataset: We first evaluate SeqVLAD with different number of visual centers on the MSVD dataset. Then we compare our SeqVLAD with different aggregation methods. The video features are extracted from Resnet-200. The experimental results are shown in Table II. "SGRU-RCN+mean pooling" denotes only using SGRU-RCN to encode video. "NetVLAD+sum pooling" denotes only using the VLAD encoding to encode video. From Table II, we observe that SeqVLAD outperforms SGRU-RCN+mean pooling and NetVLAD+sum pooling when the number of centers is set to 16, 32 and 64. SeqVLAD with SGRU-RCN performs better than SeqVLAD with GRU-RCN under the setting of 16, 32 and 64 centers. And SeqVLAD with SGRU-RCN obtains the highest value of 35.15 and 86.04 for METEOR and CIDEr, respectively.

As Table II shows, the performance of Bi-direction SeqVLAD with SGRU-RCN outperforms SeqVLAD with SGRU-RCN when the number of center is set to 16. When the number of center is set to 32, SeqVLAD with SGRU-RCN outperforms Bi-direction SeqVLAD with SGRU-RCN by 2.1% on BLEU@4. When the number of center is set to 64, SeqVLAD with SGRU-RCN outperforms Bi-direction SeqVLAD with SGRU-RCN by 1.14%and 1.62% on BLEU@4 and CIDEr, respectively.

Then we compare SeqVLAD with several state-of-the-art methods. We first briefly introduce these methods (listed in Table III) as follow:

TABLE III

COMPARISON RESULTS BETWEEN SEQVLAD AND THE STATE-OF-THE-ART METHODS ON MSVD DATASET. WE LET 'C' DENOTES C3D, 'V' DENOTES VGGNET, 'G' DENOTES GOOGLENET, 'R' DENOTES RESNET-200, AND 'BI' DENOTES BIDIRECTIONAL

Methods	METEOR	BLEU@4	CIDEr
SA [30]	29.60	41.92	51.67
S2VT [41]	29.20	-	-
G+Bi+GRU-RCN$_1$ [18]	31.70	48.42	65.38
G+Bi+GRU-RCN$_2$ [18]	31.60	43.26	68.01
V+LSTM-E [42]	29.50	40.20	-
C+LSTM-E [42]	29.90	41.70	-
G+HRNE [2]	33.10	43.80	-
V+p-RNN [43]	31.10	44.30	62.10
C+p-RNN [43]	30.30	47.40	53.60
LSTM-TSA$_I$ [45]	32.40	50.20	71.50
LSTM-TSA$_V$ [45]	32.60	50.50	71.70
G+MVRN [28]	32.79	48.12	73.21
R+MVRN [28]	33.91	52.49	78.41
Boundary-aware [34]	32.40	42.50	63.50
SCN-LSTM [44]	33.40	50.20	77.00
G+MVRN+pre-train [28]	33.39	49.50	75.45
R+MVRN+pre-train [28]	34.45	53.82	81.20
LSTM-TSA$_{IV}$ [45]	33.50	52.80	74.00
DMRM [46]	33.60	51.10	74.80
G+SeqVLAD (K=32) (ours)	33.17	50.48	77.13
R+SeqVLAD (K=32) (ours)	**34.61**	**54.57**	**83.85**
R+SeqVLAD (K=64) (ours)	**35.15**	51.00	**86.04**

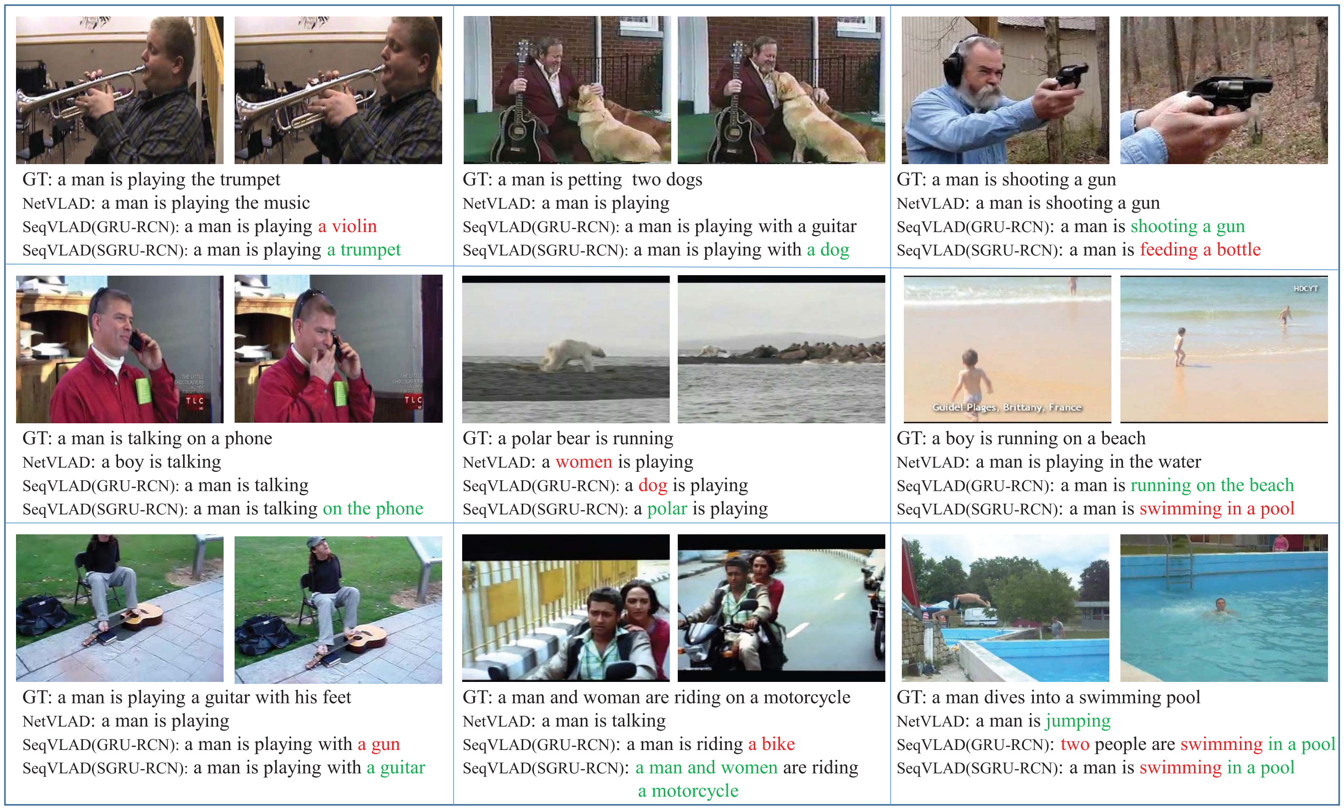
- Temporal attention (SA) [30] is an attention based method which focuses on the temporal locations and generates video captions by LSTM decoder.
- S2VT [41] is based on encoder-decoder framework. One LSTM is used to form video representation and another LSTM is used for generating video description.
- GRU-RCN [18] utilizes convolutional RNNs to encoding features of successive video frames and capture high-level percepts and low-level percepts to construct a better spatio-temporal feature representations for videos.
- LSTM-E [42] projects the visual features and text features into the same subspaces and generates video captions with a modified loss between description and visual features.
- HRNE [2] utilizes a hierarchical RNNs to capture temporal information from long-duration time which composites multiple successive frames at a high level.
- Paragraph RNN (p-RNN) [43] devises a sentence generator and a paragraph generator, which hierarchically generates multiple sentences to describe a video.
- Boundary-aware [34] utilizes LSTM cell to identify discontinuity points between frames or segments and modify the temporal connections of the encoding layer accordingly.
- MVRN [28] utilizes a multirate RNNs to deal with motion speed variance in videos. The model is pre-trained by an unsupervised video representation learning schema.
- SCN-LSTM [44] designs a Semantic Compositional Network (SCN) to utilize semantics concepts detected from the image for video description generating.
- LSTM-TSA [45] devises CNNs and RNNs architecture to combine semantics extracted from both images and videos. The detected semantic attributes are used to generate video captions.
- DMRM [46] utilizes a Dual Memory Recurrent Model to incorporate the temporal structure of global features and regions-of-interest features for video captioning.

As shown in Table III, all the compared methods are divided into two blocks. We first compare our SeqVLAD with those methods that only take single feature as their inputs. From the results listed in Table III (see the first block) we can observe: (1) Our SeqVLAD with GoogLeNet feature outperforms most of previous methods except R+MVRN [28]. Particularly, compared with SA [30], and S2VT [41], SeqVLAD with GoogLeNet features obtains a performance improvement about 2.5% on the metric of METEOR. While comparing two GRU-RCN [18] architecture, our method also outperforms by 1.4%, 2.0%, and 9.0% for METEOR, BLEU@4, and CIDEr, respectively. It illustrates that our SeqVLAD could perform well while incorporating SGRU-RCN and VLAD encoding procedure into a whole framework. With similar performance to G+HRNE [2] on the metric of METEOR, SeqVLAD, however, outperforms it by 6.6% on BLEU@4. (2) Compared with recently proposed LSTM-TSA [45] which utilizes the semantic information extracted from images or videos, SeqVLAD also outperforms it by 0.5% on METEOR. (3) SeqVLAD outperforms MVRN [28] by 0.4% and 0.7% for GoogLeNet and ResNet-200 features, respectively. It means that our SeqVLAD achieves state-of-the-art performance under the same CNNs features. (4) Note that R+SeqVLAD ($K = 32$) only takes single features as input and obtains the best performance for all three metrics. (5) With the number of visual centers $K = 64$, SeqVLAD obtains higher scores on both METEOR and CIDEr but lower score on BLEU@4.

Then we compare SeqVLAD with other methods (see the second block in Table III) which utilize more than one feature layers extracted from CNNs architecture. From Table III (see the second block), we can easily found that our G+SeqVLAD (K = 32) which only utilizes single layer feature could also obtain comparable performance with these methods. While using features extracted from ResNet-200, SeqVLAD achieves 34.61% on METEOR and achieves the state-of-the-art performance. All of these results show that SeqVLAD obtains better performance by modeling the fine motion information between successive video frames.

Finally, we list some captioning examples of MSVD dataset in Fig. 4. Comparing with the ground truth, captions generated by SeqVLAD are promising.

4) Experiments on the M-VAD Dataset: Furthermore, we test SeqVLAD with SGRU-RCN on the M-VAD dataset, which is a large scale dataset on the movie domain. Note that we only take single layer feature extracted from ResNet-200. We do not utilize two different kinds of features, which are extracted from two different kinds of layers or networks. All the results are shown in Table IV. From Table IV, it is easy



GT: a man is playing the trumpet
NetVLAD: a man is playing the music
SeqVLAD(GRU-RCN): a man is playing a violin
SeqVLAD(SGRU-RCN): a man is playing a trumpet

GT: a man is petting two dogs
NetVLAD: a man is playing
SeqVLAD(GRU-RCN): a man is playing with a guitar
SeqVLAD(SGRU-RCN): a man is playing with a dog

GT: a man is shooting a gun
NetVLAD: a man is shooting a gun
SeqVLAD(GRU-RCN): a man is shooting a gun
SeqVLAD(SGRU-RCN): a man is feeding a bottle

GT: a man is talking on a phone
NetVLAD: a boy is talking
SeqVLAD(GRU-RCN): a man is talking
SeqVLAD(SGRU-RCN): a man is talking on the phone

GT: a polar bear is running
NetVLAD: a women is playing
SeqVLAD(GRU-RCN): a dog is playing
SeqVLAD(SGRU-RCN): a polar is playing

GT: a boy is running on a beach
NetVLAD: a man is playing in the water
SeqVLAD(GRU-RCN): a man is running on the beach
SeqVLAD(SGRU-RCN): a man is swimming in a pool

GT: a man is playing a guitar with his feet
NetVLAD: a man is playing
SeqVLAD(GRU-RCN): a man is playing with a gun
SeqVLAD(SGRU-RCN): a man is playing with a guitar

GT: a man and woman are riding on a motorcycle
NetVLAD: a man is talking
SeqVLAD(GRU-RCN): a man is riding a bike
SeqVLAD(SGRU-RCN): a man and women are riding a motorcycle

GT: a man dives into a swimming pool
NetVLAD: a man is jumping
SeqVLAD(GRU-RCN): two people are swimming in a pool
SeqVLAD(SGRU-RCN): a man is swimming in a pool

Fig. 4. Examples of video captioning from the proposed SeqVLAD with SGRU-RCN. 'GT' denotes the ground truth captions annotated in MSVD dataset. We set the number of visual words K = 32. The first two columns are positive examples and last column is negative examples.

TABLE IV

EXPERIMENT RESULTS ON M-VAD DATASET. NOTE THAT WE ONLY TAKE SINGLE LAYER FEATURE EXTRACTED FROM RESNET-200

Methods	METEOR
SA-GoogleNet+3D-CNN [30]	4.1
HRNE [2]	5.8
HRNE with attention [2]	6.8
S2VT-RGB(VGG) [41]	6.7
DMRM [46]	6.8
LSTM-TSA$_I$ [45]	6.4
LSTM-TSA$_V$ [45]	6.9
LSTM-TSA$_{IV}$ [45]	7.2
Boundary-aware (C3D+ResNet) [34]	**7.3**
SeqVLAD with SGRU-RCN (ResNet, K=16) (ours)	6.8
SeqVLAD with SGRU-RCN (ResNet, K=32) (ours)	7.1
SeqVLAD with SGRU-RCN (ResNet, K=64) (ours)	7.2

to find that our SeqVLAD still obtains better performance on this dataset. Compared with SA-GoogleNet+3D-CNN [30], SeqVLAD ($K = 32$) outperforms it about 3.0% on METEOR. SeqVLAD also obtains a performance improvement about 0.3% over several methods, such as HRNE with attention [2], S2VT-RGB(VGG) [41], and DMRM [46]. While comparing with LSTM-TSA$_I$ [45] and LSTM-TSA$_V$ [45], SeqVLAD ($K = 64$) also outperforms them by 0.8% and 0.3%, respectively. Moreover, we achieve the competitive performance compared with LSTM-TSA$_{IV}$ [45], which combines semantic attributes from both images and videos. Finally, we also notice that Boundary-aware (C3D+ResNet) [34] outperforms our method by a small margin (e.g., 0.1% only). Note that, however, SeqVLAD model only takes a single layer feature extracted from ResNet-200, whereas the Boundary-aware (C3D+ResNet) [34] takes as input the features from both C3D and ResNet. Moreover, we have illustrated that SeqVLAD significantly outperforms Boundary-aware [34] on the MSVD dataset.

As discussed above, our SeqVLAD also performs well on the task of video captioning task, which demonstrates the effectiveness and good performance of our method.

B. Sequential VLAD for Video Action Recognition

1) Dataset: For evaluating SeqVLAD on the task of action recognition, we conduct experiments on two popular benchmarks, UCF101 [47] and HMDB51 [48].

UCF101 contains 101 action classes with each class including at least 100 video clips. It is divided into 25 groups according to the action performer. There are 13,320 trimmed video clips in this dataset.

HMDB51 dataset contains real-world videos collected from movies and web videos, which are more complicated in visual content and thus bring a great challenge in video action recognition. HMDB51 includes 6,766 video clips from 51 action classes with each of them containing at least 100 video clips.

Following the evaluation scheme from THUMOS13 challenge [49], we also employ the standard three train/test splits for evaluation. To evaluate our SeqVLAD, we use the UCF101 split 1 and HMDB51 split 1. For comparison with the state-of-the-art methods, we report the average accuracy over three splits on both UCF101 and HMDB51.

2) Experimental Setup: In the experiments, we adopt two different ways of training our model: The first way is to extract video features from deep network first, then train the SeqVLAD and the linear classifier on top of it. The second way is an end-to-end training similar to ActionVLAD [23], which takes the feature extraction, SeqVLAD and the linear classifier as a whole process to train our model.

For the first way to train our model, we use Inception with Batch Normalization (BN-Inception) [50], which is supported by [27], to extract frame-level features. We set $T = 10$ frames per video and extract the outputs of "inception_5b/output" with the size of $1024 \times 7 \times 7$ as the input of our SeqVLAD model. We use Adam to optimize our model and the initial learning rate is set to 0.0001. The mini-batch is set to 128. A dropout rate of 0.5 over the representation is used to avoid overfitting. Following [27] and [51], we perform the same data augmentation (4 corners and 1 center cropping, and their flips) for training. For testing, we sample 25 equally spaced segments from each video and average the final prediction scores to obtain the final video-level prediction. In the following experiments, the notation "RGB" and "Flow" represent still RGB images and stacked optical flow, respectively. Following the common practice in [27] and [51], we also set the weight for spatial (RGB) stream to 1 and set the weight for temporal (Flow) stream to 1.5 when conducting late fusion on these two streams (i.e., denoted by "RGB+Flow").

For the second way (end-to-end training), we also stack our SeqVLAD layer followed a linear classifier on the top of "inception_5b/output" layer of the BN-Inception network. The same as previous work [23], we also set $T = 25$ frames per video (for both RGB and Flow) to learn the video-level representation. In the stage of training, a dropout layer between the SeqVLAD layer and the classifier is used to avoid overfitting. We set the dropout ratio of 0.8 and 0.7 for RGB and Flow, respectively. The mini-batch is set to 64. The data augmentation is the same as previous work [27]. All the parameters of our whole model including BN-Inception network, the SeqVLAD layer, and the classifier are optimized by SGD with a momentum of 0.9. Similar with the training strategy of the work [23], we also use a two stage method. In the first stage, we only train the parameters in the SeqVLAD layer and the classifier. In the second stage, all the parameters of our whole model are optimized. For the RGB stream, there are 120 and 90 epochs for the first and the second stage, respectively. The learning rate is initialized as 0.02, and decreased to its 0.1 on the 90th and 150th epoch. For the Flow stream, both two stages have 120 epochs. The learning rate is initialized as 0.01, and decreased to its 0.1 on the 90th, 180th, and 210th epochs. In the stage of testing, we report our performance using 10 crops (4 corners and 1 center cropping, and their flips) per videos. More details could be found in our source code: https://github.com/youjiangxu/seqvlad-pytorch.

3) Effect of Different Center Numbers: Similar to the experiments in the task of video captioning, we first compare SeqVLAD with different numbers of visual words on the HMDB51 split 1 using the first way to train our model.

TABLE V

COMPARISON OF SEQVLAD WITH DIFFERENT NUMBER OF CENTERS ON THE HMDB51 SPLIT 1. "# CENTERS" DENOTES THE NUMBER OF VISUAL CENTERS

# Centers	RGB	Flow	RGB+Flow
32	53.7	59.8	69.4
64	54.1	60.2	71.2
128	54.8	60.7	69.7

TABLE VI

ABLATION STUDIES CONDUCTED ON THE HMDB51 SPLIT 1. "E" DENOTES THAT END-TO-END TRAINING, "I_3" DENOTES THE INITIALIZATION METHOD, AND "B" DENOTES THE BIDIRECTIONAL SGRU-RCN. DETAILED DISCUSSION CAN BE FOUND IN TEXT

Model	RGB	Flow	RGB+Flow
NetVLAD (sum pooling)	53.1	57.6	69.2
SeqVLAD (GRU-RCN)	53.2	59.2	69.7
SeqVLAD (SGRU-RCN)	54.1	60.2	71.2
SeqVLAD (SGRU-RCN) + E	54.9	64.4	72.2
SeqVLAD (SGRU-RCN) + E + I_3	55.2	65.4	72.9
SeqVLAD (SGRU-RCN) + E + I_3 + B	54.4	65.6	73.3

All results are shown in Table V. When increasing the number of visual words, the performance is improved on both RGB and Flow. However, the SeqVLAD with 64 visual words achieves the best performance of 71.2% when using both RGB and Flow modalities (e.g., "RGB+Flow"). Following [22] and [23], we thus set the number of visual words K to 64 for the following experiments.

4) Ablation Analysis: We now study in detail the impact of each component of our proposed SeqVLAD layer, and report the performance on the HMDB51 split 1 in Table VI. Note that in Table VI, the models shown in the first block are trained by the first way (not end-to-end training), and the models shown in the second block are trained by the second way (end-to-end training). More details are as follows:

- NetVLAD (sum pooling) simply accumulates the representation of video frames along the time.
- SeqVLAD (GRU-RCN) does not share the input-to-hidden parameters, which means that it uses a standard GRU-RCN to compute the spatio-temporal assignments.
- SeqVLAD (SGRU-RCN) utilizes a shared GRU-RCN to compute the spatio-temporal assignments, but isn't trained in an end-to-end manner.
- SeqVLAD (SGRU-RCN) + E utilizes the second way (end-to-end training) to train the whole model.
- SeqVLAD (SGRU-RCN) + E + I_3 adds the third initialization method, which is discussed in detail in Table VII.
- SeqVLAD (SGRU-RCN) + E + I_3 + B utilizes the bidirectional SGRU-RCN to compute spatio-temporal assignments.

From Table VI, we can observe that SeqVLAD (GRU-RCN) outperforms NetVLAD (sum pooling) about 0.5% (RGB+Flow). When sharing the input-to-hidden parameters, SeqVLAD (SGRU-RCN) improves the performance of 1.5% (RGB+Flow). The results demonstrate that SeqVLAD could capture the temporal information between successive video frames and obtain better performance with fewer parameters.

TABLE VII
EVALUATION OF DIFFERENT INITIALIZATION METHODS ON THE UCF101 SPLIT 1 AND HMDB51 SPLIT 1. "I_1," "I_2," AND "I_3" ARE THREE DIFFERENT KINDS OF INITIALIZATION METHODS. DETAILED DISCUSSION CAN BE FOUND IN TEXT

Model	UCF101	HMDB51
SeqVLAD (SGRU-RCN) + I_1	93.5	72.3
SeqVLAD (SGRU-RCN) + I_2	93.6	72.2
SeqVLAD (SGRU-RCN) + I_3	93.9	72.9

Compared with "SeqVLAD (SGRU-RCN)" (not end-to-end training), "SeqVLAD (SGRU-RCN) + E" (end-to-end training) obtains a performance improvement about 1.0%. When the parameters in SGRU-RCN are initialized by "I_3," our proposed method has 0.7% gain. The details about "I_3" initialization method is discussed in Table VII. Finally, our method can obtain a performance gain about 0.4% by using Bidirectional SGRU-RCN.

5) Effect of Different Initialization Methods: With end-to-end training, we study the impact of different initialization methods of trainable parameters in SeqVLAD layer. It is worth noting that, we evaluate the effect of different initialization methods on both split 1 of the UCF101 and HMDB51 datasets. The models to compare are as follows:

- SeqVLAD (SGRU-RCN) + I_1 uses uniform distribution to initialize W_p, U_z, U_r, U and C in the model.
- SeqVLAD (SGRU-RCN) + I_2 uses "xavier normal" [52] to initialize W_p, U_z, U_r, U and C in the model.
- SeqVLAD (SGRU-RCN) + I_3 uses "xavier normal" [52] to initialize W_p and "orthogonal" [53] to initialize U_z, U_r, U and C in the model.

From Table VII, the performance of SeqVLAD (SGRU-RCN) + I_3 outperforms SeqVLAD (SGRU-RCN) + I_1 about 0.4% on UCF101 and 0.6% on HMDB101. Moreover, SeqVLAD (SGRU-RCN) + I_3 outperforms SeqVLAD (SGRU-RCN) + I_2 about 0.3% on UCF101 and 0.7% on HMDB51. The results illustrate that how to initialize the parameters in the SeqVLAD layer is important to the performance of the models.

6) Comparison With the State-of-the-Art: In Table VIII, we compare our best performance of SeqVLAD with several state-of-the-art methods on UCF101 and HMDB51 averaged over 3 splits. As our SeqVLAD takes both still images and stacked optical flow as inputs, we first compare SeqVLAD with **Two-Stream based methods** (see the first block of Table VIII) which also use still images for the spatial stream and stacked optical flow for temporal stream. The performance outperforms all previous methods on UCF101 and HMDB51. Specifically, SeqVLAD outperforms Two-Stream (VGG-M) [1] about 6.5% on UCF101 and 12.1% on HMDB51. Compared with TSN (BN-Inception, 2-modality) [27] which also utilizes BN-Inception networks to extract features of video frames, SeqVLAD improves the performance about 0.5% on UCF101 and 3.0% on HMDB51. When compared with ActionVLAD (LateFuse, VGG-16) [23], SeqVLAD obtains a performance improvement about 1.8% on UCF101 and 4.6% on HMDB51.

TABLE VIII
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON UCF101 AND HMDB51 AVERAGED OVER 3 SPLITS

Methods	UCF101	HMDB51
Factorized ConvNet [54]	88.1	59.1
VideoDarwin [55]	-	63.7
Two-Stream (VGG-M) [1]	88.0	59.4
Two-Stream (VGG-16) [51]	91.4	58.5
Two-Stream Fusion (VGG-16) [56]	92.5	65.4
TDD+FV [13]	90.3	63.2
Transformations [57]	92.4	62.0
LTC [58]	91.7	64.8
KVMF [26]	93.1	63.3
TSN (BN-Inception, 2-modality) [27]	94.0	68.5
ActionVLAD (LateFuse, VGG-16) [23]	92.7	66.9
AdaScan [59]	89.4	54.9
ST-ResNet [60]	93.5	66.4
SeqVLAD (BN-Inception, RGB+Flow)	**94.5**	**71.5**
DT+MVSV [61]	83.5	55.9
iDT+FV [4]	85.9	57.2
iDT+HSV [62]	87.9	61.1
MoFAP [63]	88.3	61.7
LTC+iDT [58]	92.7	67.2
C3D+iDT [25]	90.4	-
Two-Stream Fusion+iDT [56]	93.5	69.2
TSN (BN-Inception, 3-modality) [27]	94.2	69.4
ActionVLAD (VGG-16)+ iDT [23]	93.6	69.8
AdaScan+iDT [59]	91.3	61.0
AdaScan+iDT+C3D [59]	93.2	66.9
Cool-TSN [64]	94.2	69.5
ST-VLMPF(DF) [65]	93.6	69.5
ST-ResNet+iDT [60]	94.6	70.3
DT+Hybrid architectures [66]	92.5	70.4
SeqVLAD (BN-Inception, RGB+Flow)+iDT	**95.0**	**73.7**

Then we compare SeqVLAD with several **Two-Stream + "X" based methods** (the second block of Table VIII). The **"X"** could be one of DT, iDT, C3D, or leveraging additional data. Though SeqVLAD only takes as inputs both RGB images and optical flow, it outperforms most of Two-Stream+"X" methods. Compared with TSN (BN-Inception, 3-modality) [27], which takes RGB images, optical flow, and warped flow as inputs, SeqVLAD still outperforms them about 0.8% on UCF101 and 4.3% on HMDB51. SeqVLAD also outperforms ActionVLAD (VGG-16)+iDT [23] about 1.4% on UCF101 and 3.9% on HMDB51. Moreover, compared with very recently proposed methods [59], [60], [64]–[66], SeqVLAD also obtains a better performance.

Finally, we notice that the Two-Stream I3D [67] proposed recently outperforms our method by roughly 2.9% on UCF101 and 6.5% on HMDB51. However, the Two-Stream I3D is pre-trained on a large external dataset of Kinetics [68], which has 400 human action classes and over 400 clips per class. Considering that we only have 3,570 videos for training on the HMDB51 dataset, the proposed SeqVLAD achieves a good performance with a relatively low training cost.

V. CONCLUSIONS

In this paper, for learning video representations, we propose a novel model of Sequential Video VLAD, which takes the merits of VLAD encoding method and a Shared GRU-RCN architecture. The contributions of our Sequential VLAD are as follows. First, our model is trainable via backpropagation. Then, we can learn a discriminative video representation by a trainable VLAD encoding and capture the temporal information by the aggregation parameters learned from our proposed Shared GRU-RCN. Moreover, our model has fewer parameters and thus less possibility of overfitting with our Shared GRU-RCN. Finally, we conduct extensive experiments on both video captioning and video action recognition. Experimental results on benchmark datasets demonstrate the effectiveness and good performance of our method.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. NIPS*, 2014, pp. 568–576.

[2] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang, "Hierarchical recurrent neural encoder for video representation with application to captioning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1–10.

[3] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 3169–3176.

[4] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 3551–3558.

[5] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3304–3311.

[6] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, no. 3, pp. 222–245, 2013.

[7] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[9] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. NIPS*, 2014, pp. 487–495.

[10] J. Han, D. Zhang, G. Cheng, N. Liu, and D. Xu, "Advanced deep-learning techniques for salient and category-specific object detection: A survey," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 84–100, Jan. 2018.

[11] J. Han, R. Quan, D. Zhang, and F. Nie, "Robust object co-segmentation using background prior," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1639–1651, Apr. 2018.

[12] J. Han, G. Cheng, Z. Li, and D. Zhang, "A unified metric learning-based framework for co-saliency detection," *IEEE Trans. Circuits Syst. Video Technol.*, to be published, doi: 10.1109/TCSVT.2017.2706264.

[13] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4305–4314.

[14] Z. Ma, X. Chang, Y. Yang, N. Sebe, and A. G. Hauptmann, "The many shades of negativity," *IEEE Trans. Multimedia*, vol. 19, no. 7, pp. 1558–1568, Jul. 2017.

[15] A. Graves, A. R. Mohamed, and G. Hinton, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, vol. 14. 2014, pp. 1764–1772.

[16] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 843–852.

[17] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2625–2634.

[18] N. Ballas, L. Yao, C. Pal, and A. Courville, "Delving deeper into convolutional networks for learning video representations," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, 2016.

[19] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. NIPS*, 2015, pp. 802–810.

[20] Z. Li, E. Gavves, M. Jain, and C. G. M. Snoek. (2016). "Videolstm convolves, attends and flows for action recognition." [Online]. Available: https://arxiv.org/abs/1607.01794

[21] Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative CNN video representation for event detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1798–1807.

[22] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5297–5307.

[23] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "Actionvlad: Learning spatio-temporal aggregation for action classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Apr. 2017, pp. 971–980.

[24] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Proc. Workshop Statist. Learn. Comput. Vis. (ECCV)*, 2004, pp. 1–2.

[25] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[26] W. Zhu, J. Hu, G. Sun, X. Cao, and Y. Qiao, "A key volume mining deep framework for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1991–1999.

[27] L. Wang *et al.*, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. ECCV*, 2016, pp. 20–36.

[28] L. Zhu, Z. Xu, and Y. Yang, "Bidirectional multirate reconstruction for temporal modeling in videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 1339–1348.

[29] Z. Wang, L. Wang, Y. Wang, B. Zhang, and Y. Qiao, "Weakly supervised patchnets: Describing and aggregating local patches for scene recognition," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 2028–2041, Apr. 2017.

[30] L. Yao *et al.*, "Describing videos by exploiting temporal structure," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4507–4515.

[31] D. L. Chen and W. B. Dolan, "Collecting highly parallel data for paraphrase evaluation," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics*, 2011, pp. 190–200.

[32] A. Torabi, C. Pal, H. Larochelle, and A. Courville. (2015). "Using descriptive video services to create a large data source for video annotation research." [Online]. Available: https://arxiv.org/abs/1503.01070

[33] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko, "Translating videos to natural language using deep recurrent neural networks," in *Proc. NAACL HLT*, Jun. 2014, pp. 1494–1504.

[34] L. Baraldi, C. Grana, and R. Cucchiara, "Hierarchical boundary-aware neural encoder for video captioning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3185–3194.

[35] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics*, 2002, pp. 311–318.

[36] M. Denkowski and A. Lavie, "Meteor universal: Language specific translation evaluation for any target language," in *Proc. 9th Workshop Statist. Mach. Transl.*, 2014, pp. 376–380.

[37] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Proc. Workshop Text Summarization Branches Out*, vol. 8, 2004, pp. 74–81.

[38] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based image description evaluation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4566–4575.

[39] X. Chen *et al.* (2015). "Microsoft COCO captions: Data collection and evaluation server." [Online]. Available: https://arxiv.org/abs/1504.00325

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*, 2016, pp. 630–645.

[41] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence—Video to text," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4534–4542.

[42] Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui, "Jointly modeling embedding and translation to bridge video and language," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4594–4602.

[43] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu, "Video paragraph captioning using hierarchical recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4584–4593.

[44] Z. Gan *et al.*, "Semantic compositional networks for visual captioning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1141–1150.

[45] Y. Pan, T. Yao, H. Li, and T. Mei, "Video captioning with transferred semantic attributes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 984–992.

[46] Z. Yang, Y. Han, and Z. Wang, "Catching the temporal regions-of-interest for video captioning," in *Proc. ACM Multimedia Conf. (MM)*, 2017, pp. 146–153.

[47] K. Soomro, A. R. Zamir, and M. Shah. (Dec. 2012). "UCF101: A dataset of 101 human actions classes from videos in the wild." [Online]. Available: https://arxiv.org/abs/1212.0402

[48] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Nov. 2011, pp. 2556–2563.

[49] Y.-G. Jiang *et al.*, "THUMOS challenge: Action recognition with a large number of classes," in *Proc. ICCV 1st Int. Workshop Action Recognit. Large Number Classes (THUMOS)*, Sydney, NSW, Australia, Dec. 2013. [Online]. Available: http://crcv.ucf.edu/ICCV13-Action-Workshop/

[50] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[51] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao. (2015). "Towards good practices for very deep two-stream convnets." [Online]. Available: https://arxiv.org/abs/1507.02159

[52] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[53] A. M. Saxe, J. L. McClelland, and S. Ganguli. (2013). "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks." [Online]. Available: https://arxiv.org/abs/1312.6120

[54] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi, "Human action recognition using factorized spatio-temporal convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4597–4605.

[55] B. Fernando, E. Gavves, M. J. Oramas, A. Ghodrati, and T. Tuytelaars, "Modeling video evolution for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5378–5387.

[56] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1933–1941.

[57] X. Wang, A. Farhadi, and A. Gupta, "Actions~ transformations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2658–2667.

[58] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1510–1517, Jun. 2018.

[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. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5699–5708.

[60] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Proc. NIPS*, 2016, pp. 3468–3476.

[61] Z. Cai, L. Wang, X. Peng, and Y. Qiao, "Multi-view super vector for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 596–603.

[62] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Comput. Vis. Image Understand.*, vol. 150, Sep. 2016, pp. 109–125.

[63] L. Wang, Y. Qiao, and X. Tang, "MoFAP: A multi-level representation for action recognition," *Int. J. Comput. Vis.*, vol. 119, no. 3, pp. 254–271, 2016.

[64] C. R. De Souza, A. Gaidon, Y. Cabon, and A. M. López, "Procedural generation of videos to train deep action recognition networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2594–2604.

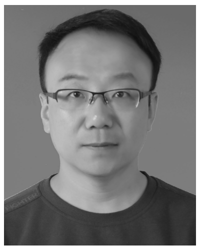
[65] I. C. Duta, B. Ionescu, K. Aizawa, and N. Sebe, "Spatio-temporal vector of locally max pooled features for action recognition in videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, Jul. 2017, pp. 3205–3214.

[66] C. R. De Souza, A. Gaidon, E. Vig, and A. M. López, "Sympathy for the details: Dense trajectories and hybrid classification architectures for action recognition," in *Proc. ECCV*, 2016, pp. 697–716.

[67] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.

[68] W. Kay *et al.* (May 2017). "The kinetics human action video dataset." [Online]. Available: https://arxiv.org/abs/1705.06950

Youjiang Xu received the B.S. and M.S. degrees from Tianjin University in 2015 and 2018, respectively. His current research interests include video action recognition, video captioning, and scene text detection.



Yahong Han (M'15) received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2012. From 2014 to 2015, he visited Prof. B. Yu's Group, UC Berkeley, as a Visiting Scholar. He is currently a Full Professor with the School of Computer Science and Technology, Tianjin University, Tianjin, China. His current research interests include multimedia analysis, computer vision, and machine learning.

Richang Hong (M'12) received the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2008. He was a Research Fellow with the School of Computing, National University of Singapore, from 2008 to 2010. He is currently a Professor with the Hefei University of Technology, Hefei. He has co-authored over 100 publications in the areas of his research interests, which include multimedia content analysis and social media. He is a member of the ACM and the Executive Committee Member of the ACM SIGMM China Chapter. He was a recipient of the Best Paper Award in the ACM Multimedia 2010, the Best Paper Award in the ACM ICMR 2015, and the Honorable Mention of the IEEE TRANSACTIONS ON MULTIMEDIA Best Paper Award. He served as the Technical Program Chair of the PCM 2018 and the MMM 2016. He served as an Associate Editor for the *IEEE Multimedia Magazine*, *Information Sciences* (Elsevier), and *Signal Processing* (Elsevier).

Qi Tian (M'96–SM'03–F'16) was a tenure-track Assistant Professor from 2002 to 2008 and a tenured Associate Professor from 2008 to 2012. From 2008 to 2009, he took one-year faculty leave at Microsoft Research Asia as a Lead Researcher with the Media Computing Group. He is currently a Full Professor with the Department of Computer Science, The University of Texas at San Antonio (UTSA).

He received the B.E. degree in electronic engineering from Tsinghua University in 1992, the M.S. degree in ECE from Drexel University in 1996, and the Ph.D. degree in ECE from the University of Illinois at Urbana–Champaign in 2002. He has published over 360 refereed journal and conference papers. He has co-authored a best paper in the ACM ICMR 2015, PCM 2013, MMM 2013, the ACM ICIMCS 2012, and the Top 10% Paper Award in MMSP 2011, and a best student paper in ICASSP 2006. He has also co-authored a best student paper in ICME 2015 and a best paper in PCM 2007. His research interests include multimedia information retrieval, computer vision, pattern recognition, and bioinformatics.

His research projects were funded by ARO, NSF, DHS, Google, FXPAL, NEC, SALSI, CIAS, Akiira Media Systems, HP, Blippar, and UTSA. He received the 2017 UTSA President's Distinguished Award for Research Achievement, the 2016 UTSA Innovation Award, the 2014 Research Achievement Award from the College of Science, UTSA, the 2010 Google Faculty Award, and the 2010 ACM Service Award. He is an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the *ACM Transactions on Multimedia Computing, Communications, and Applications*, and *Multimedia System Journal*. He serves on the Editorial Board of the *Journal of Multimedia* and the *Journal of Machine Vision and Applications*. He is a Guest Editor of the IEEE TRANSACTIONS ON MULTIMEDIA and the *Journal of Computer Vision and Image Understanding*.