HSA-RNN: Hierarchical Structure-Adaptive RNN for Video Summarization

Bin Zhao[1], Xuelong Li[2], Xiaoqiang Lu[2]
[1]School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL),
Northwestern Polytechnical University, Xi'an, Shaanxi, P. R. China
[2]Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences,
Xi'an, Shaanxi, P. R. China
binzhao111@gmail.com, xuelong_li@opt.ac.cn, luxiaoqiang@opt.ac.cn

Abstract

Although video summarization has achieved great success in recent years, few approaches have realized the influence of video structure on the summarization results. As we know, the video data follow a hierarchical structure, i.e., a video is composed of shots, and a shot is composed of several frames. Generally, shots provide the activity-level information for people to understand the video content. While few existing summarization approaches pay attention to the shot segmentation procedure. They generate shots by some trivial strategies, such as fixed length segmentation, which may destroy the underlying hierarchical structure of video data and further reduce the quality of generated summaries. To address this problem, we propose a structure-adaptive video summarization approach that integrates shot segmentation and video summarization into a Hierarchical Structure-Adaptive RNN, denoted as HSA-RNN. We evaluate the proposed approach on four popular datasets, i.e., SumMe, TVsum, CoSum and VTW. The experimental results have demonstrated the effectiveness of HSA-RNN in the video summarization task.

1. Introduction

Nowadays, video data are increasing explosively due to the popularity of video capture equipments, such as smart phones and surveillance cameras. Videos have become the most common visual data, which causes an urgent demand for automatic tools to deal with the huge amount of video data efficiently. In particular, video summarization is one of the tools rising to this challenge [32, 37].

Video summarization provides us an efficient way to browse and understand a lengthy video by shortening it into a compact version, i.e., highlighting the essence and removing redundancy [31]. Practically, video summarization can condense the video at three levels, that is, shots [27], frames [9] and objects [18]. In this paper, we focus on the first one that summarizes the video with several key shots, because it can better preserve the dynamic information and spatio-temporal consistence of the video content [1, 12].

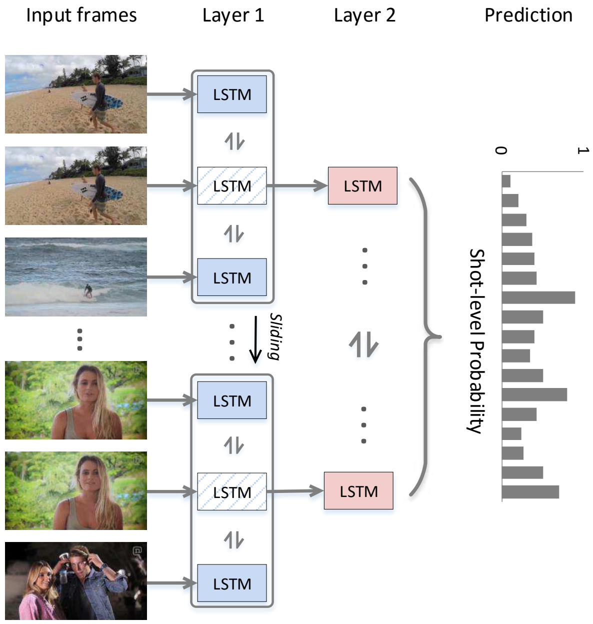


Figure 1. The diagram of the proposed HSA-RNN, where Layer 1 and Layer 2 are designed to exploit the video structure and generate the video summary, respectively. Specifically, the blue and red boxes represent the bidirectional LSTM unit in each layer. The dashed boxes indicate the locations of detected shot boundaries. The bidirectional LSTM in Layer 1 operates in a sliding manner, and the stride at each step is equal to the length of previous detected shot.

There has been a steady development in shot-based video summarization. Existing approaches summarize the video following a two-stage architecture, i.e., shot segmentation and key shot selection [12, 27, 38, 23]. In particular, most approaches focus on designing various models for the key shot selection step, like clustering and dictionary learning algorithms [1, 8, 39], property models [17, 21, 13], and recently proposed sequence models [37, 38, 23]. However, few approaches pay attention to the shot segmentation part. Practically, for most approaches, the video is segmented into shots simply by abrupt changes among frames [23, 27], motion magnitude variances [12, 21], or even fixed length segmentation [38, 29, 33, 15]. These methods have not made full use of the video structure, and usually lead to unsatisfied shot segmentation results.

Actually, the summarization results are heavily dependent on the video structures [12, 27, 38]. As we know, videos share a hierarchical structure that a video is composed of shots, and a shot is composed of several frames [26, 4]. Shots are the fundamental units for people to understand the activities in the video. Thus, the poor performance of existing approaches on shot segmentation may lead to inevitable mistakes. Concretely, the inaccurate segmentation causes information chaos, i.e., the information loss of a certain shot and information mixture of several adjacent shots, which damages the integrity and independence of the activities in the video. As a result, it leads to the misjudgment of summarization approaches on the latent video structure, and finally causes interferences to generate correct summaries.

To address this problem, we propose a *Hierarchical Structure-Adaptive RNN* (HSA-RNN) that can jointly exploit the video structure and summarize the video content. As depicted in Figure 1, HSA-RNN has two layers, constructed by bidirectional *Long Short-Term Memory* (LSTMs). Specifically, the first layer is developed to exploit the video structure. The fixed length bidirectional LSTM operates on the video frames in a sliding manner, and tries to detect the shot boundaries step by step (the stride at each step is equal to the length of previous detected shot). Once the shot boundaries are detected, the hidden states corresponding to those locations are taken as the encoded shot features and input to the upper layer. The second layer is designed to capture the forward-backward temporal dependencies among shots, and predict the probability of each shot to be selected into the summary.

Overall, the contributions of this paper are summarized as follows:

1) To our knowledge, we are the first to propose a structure-adaptive video summarization approach that jointly exploits the video structure and summarizes the video content. It can make up the weakness of existing approaches in video structure exploitation, and further improve the summary quality.

2) We design a sliding bidirectional LSTM to detect shot boundaries in the video. It achieves an accurate segmentation of long videos with much shorter LSTM, so that the vanishing gradient problem is mitigated.

3) The results on four popular datasets, i.e., SumMe [12], TVsum [29], CoSum [5], and VTW [35], have demonstrated that our approach significantly improves the performance on shot segmentation and video summarization.

2. Related Works

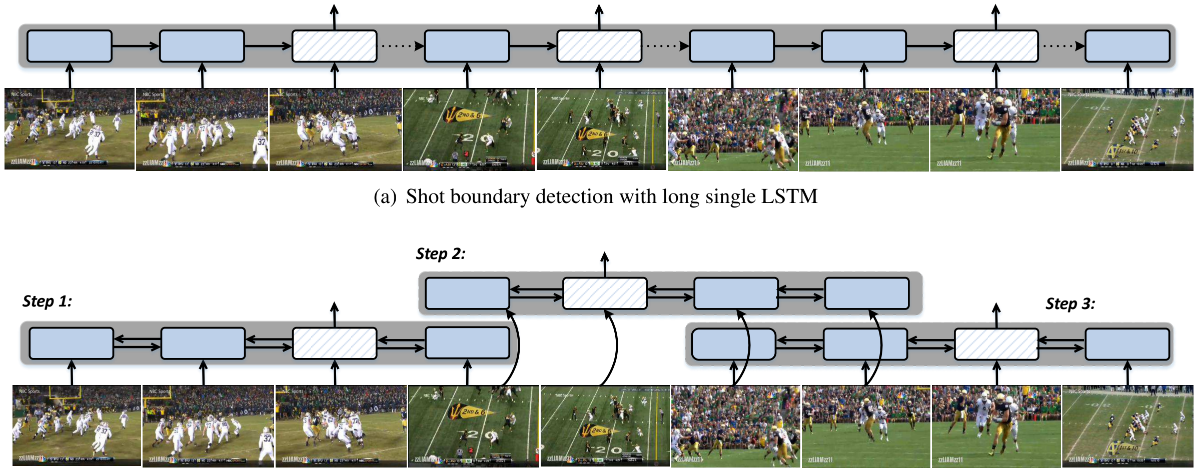
Video summarization is a long-standing problem. A large amount of approaches have been proposed in the literature. They mainly fall in two broad categories, i.e., unsupervised ones and supervised ones.

Unsupervised approaches typically select key shots according to heuristic criteria [39, 25, 19, 24], including representativeness and diversity, etc. One main subcategory of unsupervised video summarization is cluster-based approaches [24, 1, 8], which aggregate similar shots into the same cluster. Then cluster centers are selected as the components of the final summary. Originally, clustering algorithms are directly used for video summarization [40, 14]. Afterwards, domain knowledge is taken into account to generate better results [8, 24]. Some other works exploit more powerful models for video summarization based on clustering, e.g., Ngo *et al.* [25] transform the video into an undirected graph and generate the summary by cutting the graph into several clusters. Recently, Chu *et al.* [5] propose to select visually co-occurring shots across videos with the same topic by commonality analysis.

Dictionary learning is another subcategory of unsupervised summarization approaches [9, 22, 39, 7], which aims to find a few key shots to construct a dictionary as the representative of the video content. To further retain the local similarity of shots, Lu *et al.* [20] propose a *Locality-constrained Linear Coding* (LLC) approach based on dictionary learning. Moreover, to improve the efficiency of video summarization, Zhao *et al.* [39] exploit a quasi real-time approach to summarize videos.

Different from unsupervised ones, supervised approaches utilize human-created summaries to learn the underlying selection criteria, and achieve better results [36, 10, 27, 17, 21, 12]. In supervised approaches, [17] and [12] formulate video summarization as a scoring problem with regard to interestingness and importance, respectively. Then shots with highest scores are selected to produce the video summary. Besides, Lu *et al.* [21] propose a storyness model to make the summary follow a smooth story line. Several other works even explore auxiliary information, including web images [15], video category [27] and video titles [29], to improve the summarization process.

More recently, deep learning based approaches are gaining increasing attention [33, 37, 38, 23]. Yao *et al.* [33] propose a deep rank model based on *Convolutional Neural Networks* (CNN) to encode the input shot and output a ranking score. Zhang *et al.* [37] develop a bidirectional LSTM to predict the probability of each shot to be selected. Furthermore, a hierarchical architecture of LSTMs is constructed



(a) Shot boundary detection with long single LSTM

(b) Shot boundary detection with sliding bidirectional LSTM

Figure 2. The comparison between the long single LSTM and sliding bidirectional LSTM on shot boundary detection. Compared to the long single LSTM, the sliding bidirectional LSTM is much shorter, which can mitigate the vanishing gradient problem by avoiding extremely long temporal dependency exploitation. Moreover, it can capture both the forward and backward information. Note that the blue boxes denote the LSTM units, and the dashed boxes indicate the locations of detected shot boundaries.

to deal with the long temporal dependencies among video frames [38], which has achieved the state-of-the-art in video summarization. But it fails to capture the video structure information, where the shot are generated by fixed length segmentation. Besides, Mahasseni *et al.* [23] propose an adversarial network to summarize the video by minimizing the distance between the video and its summary.

3. Our Approach

Our approach has two layers, where the first layer is to exploit the video structure, and the second layer is to summarize the video. In this section, we describe our approach layer by layer.

3.1. Video Structure Exploitation

Our work is based on the basic assumption that video data have a hierarchical structure that frames form shots and shots form video [38]. Shot segmentation is the core problem in video structure exploitation. Different from existing approaches that segment the video by change point detection or fixed length segmentation, we try to segment the video based on the temporal dependencies among frames.

The target of this layer is to locate the shot boundaries in the video and generate the visual feature for each shot. Specifically, taken the frame feature sequence $(f_1, f_2, \ldots, f_n)$ as input, the output is the shot feature sequence $(s_1, s_2, \ldots, s_m)$, where n and m are the number of frames and shots, respectively. Ideally, we hope each shot feature is extracted exactly from the frame features belonging to that shot.

Since LSTM is good at sequence modeling, the most straightforward idea is to apply a long LSTM to the video and detect the shot boundaries by a threshold of the output at each step [4, 6], as depicted in Figure 2(a). But this scheme has natural defects:

1) It is unfeasible to apply bidirectional LSTM to this architecture. Actually, it is hard to detect the boundary just utilizing the forward information, the information behind the boundary is also very important.

2) The threshold for judging the boundary is hard to set. Worse still, the threshold destroys the differentiability of the architecture, which makes the end-to-end training difficult to carry out.

To address above problems, a sliding bidirectional LSTM is designed in this part. As depicted in Figure 2(b), it is like a fixed-length one-dimensional filter that operates on the frame sequence in a sliding manner. The length of the bidirectional LSTM, k, is empirically selected according to the shot lengths ($k = 240$ in this paper), and the stride at each step is equal to the length of detected shot. The intuition lying behind the sliding bidirectional LSTM is that: 1) The sliding operation enables short LSTM to process long videos. It avoids long temporal dependency exploitation among thousands of frames, which can mitigate the vanishing gradient problem. 2) The bidirectional LSTM jointly captures the forward and backward information in frame sequence, which can detect the shot boundary effectively. 3) The sliding bidirectional LSTM just processes the local frames at each step, which reduces the interference of irrelevant global information.

Specifically, at the first step, the bidirectional LSTM operates on the frame subsequence $(f_1, f_2, \ldots, f_k)$ by the following equations,

$$h_t^{1,f} = LSTM\left(f_t, h_{t-1}^{1,f}\right), \tag{1}$$

$$h_t^{1,b} = LSTM\left(f_t, h_{t+1}^{1,b}\right), \tag{2}$$

where $LSTM(\cdot)$ stands for the operations in each LSTM unit. Specifically, the LSTM proposed in [34] is employed in our work. Eq. (1) and (2) denote the computations of forward and backward LSTM, respectively. It can be observed that the main difference between them lying in that the backward LSTM operates reversely. $h_t^{1,f}$ and $h_t^{1,b}$ denote the hidden states of the bidirectional LSTM in the first layer. They capture the forward and backward temporal dependencies among frames, respectively.

Then, $h_t^{1,f}$ and $h_t^{1,b}$ are utilized to compute the confidence of each frame to be the shot boundary,

$$c_t = softmax\left(Relu\left(W_c\left[h_t^{1,f}; h_t^{1,b}\right] + b_c\right)\right), \tag{3}$$

where W_c and b_c are parameters to be learned, $[\cdot\,;\cdot]$ indicates the concatenation of vectors, $Relu(\cdot)$ is the activation function, $softmax(\cdot)$ is utilized to normalize the values in c_t. Specifically, c_t is a two-dimensional vector, where the element $c_t(1)$ and $c_t(2)$ denote the confidence of frame t to be the shot boundary or not, respectively. Therefore, the shot boundary is determined as the frame with the maximum $c_t(1)$, i.e.,

$$t^* = \underset{t}{\operatorname{argmax}}\left\{c_1(1), c_2(1), \ldots, c_k(1)\right\}, \tag{4}$$

and $h_{t^*}^{1,f}$ is taken as the first shot feature s_1. Then, the upper layer is activated with the input of s_1, and the sliding bidirectional LSTM moves to the next step taking the following frame sequence $\{f_{t^*+1}, f_{t^*+2}, \ldots, f_{t^*+k}\}$ as input. Finally, it will find out all the shot boundaries and the shot feature sequence $(s_1, s_2, \ldots, s_m)$ is computed. Note that m is not fixed in our approach, it varies with different videos.

3.2. Structure-Adaptive Video Summarization

After shot features are extracted, they will be input to the second layer for video summarization. As aforementioned, this layer is also a bidirectional LSTM, which is utilized to capture the forward and backward temporal dependencies among shots, and predict which shots are most representative to the video content.

Practically, the calculation in this layer is formulated as:

$$h_t^{2,f} = LSTM\left(s_t, h_{t-1}^{2,f}\right), \tag{5}$$

$$h_t^{2,b} = LSTM\left(s_t, h_{t+1}^{2,b}\right), \tag{6}$$

where $h_t^{2,f}$ and $h_t^{2,b}$ denote the hidden states of the forward and backward LSTM, respectively. They capture the global temporal dependencies among shots, which are essential to generate the video summary.

Then, the output hidden state is employed to predict the probability of each shot to be selected into the summary. It is formulated as:

$$p_t = softmax\left(Relu\left(W_p\left[h_t^{2,f}; h_t^{2,b}; s_t\right] + b_p\right)\right). \tag{7}$$

Similar to c_t, p_t is also a two-dimensional vector, and each element reflects the probability of the t-th shot to be key or non-key. Therefore, the key shots in the summary are finally selected according to p_t. Besides, it can be observed from Eq. (8) that p_t is jointly determined by the shot feature s_t and the hidden states of the bidirectional LSTM, i.e., $h_t^{2,f}$ and $h_t^{2,b}$. This is because that s_t encodes the intra-shot temporal dependencies among frames, $h_t^{2,f}$ and $h_t^{2,b}$ capture the forward and backward inter-shot dependencies of the video. All of these information are important to determine the representativeness of shot t.

In the training procedure, given human-created summaries as references, the parameters in HSA-RNN are learned by the following function:

$$\Omega = \underset{\Omega}{\arg\min} \frac{1}{T} \sum_{i=1}^{T} \sum_{t=1}^{n^{(i)}} L\left(\hat{p}_t^{(i)}, g_t^{(i)}\right), \tag{8}$$

where Ω denotes all parameters in the proposed approach. T is the total number of training videos, $n^{(i)}$ are the number of frames in video i. Considering that the generated shots may have different durations with human-created shots, in this paper, the predicted shot-level probability $p_t^{(i)}$ is extended to the frame-level $\hat{p}_t^{(i)}$ by assigning frames with the probability values of their shots. In this case, Eq. (8) can optimize not only the summarization results, but also the boundary detection results, since frames in the same shot share the same g_t scores while others not (making the boundary discriminative). The loss function L measures the cross-entropy of the generated distribution $\hat{p}_t^{(i)}$ and the reference distribution $g_t^{(i)}$.

Practically, to achieve a faster coverage, we take a layer-wise training strategy for the proposed HSA-RNN. In other words, the first layer is pre-trained, and then the parameters are fixed to train the second layer. Finally, the whole architecture is fine-tuned end-to-end. Specifically, the first layer is pre-trained with an extra loss function,

$$\Omega^1 = \underset{\Omega^1}{\arg\min} \frac{1}{S} \sum_{i=1}^{S} \sum_{t=1}^{F^{(i)}} L\left(c_t^{(i)}, b_t^{(i)}\right), \tag{9}$$

where Ω^1 stands for the training parameters in the first layer. $b_t^{(i)}$ is the boundary labels, S is the total number of training samples, $F^{(i)}$ denotes the number of frames in sample i.

Each training sample is composed of two shots and contains exactly one boundary. Note that Our approach doesn't need shot boundary labels of the whole video, the boundaries of key shots are enough.

For both the first and second layer, training is performed by minimizing the cross-entropy loss with the RMSProp Optimizer, with a learning rate of $r = 1 \times 10^{-4}$, decay parameter $\rho = 0.9$ and epsilon $\epsilon = 1 \times 10^{-5}$. The dimensionality of the LSTM hidden state is fixed as 256, and $dropout\ rate = 0.5$ is applied for each LSTM to avoid overfitting.

4. Experiments

The proposed approach is tested on four datasets, i.e., SumMe [12], TVsum [29], CoSum [5] and VTW [35]. Both the results on shot boundary detection and video summarization are presented.

4.1. Setup

4.1.1 Datasets

In this paper, the first dataset for training is composed of three popular small datasets, i.e., SumMe [12], TVsum [29], CoSum [5]. The three datasets contain 25, 50, 51 videos, respectively. These videos share similar contents, styles and durations. Therefore, to enrich the training data, it is more proper to combine them into one dataset, namely Combined dataset in this paper. Similar setting can be found in many existing summarization approaches [37, 38]. Concretely, the Combined dataset contains 126 videos, the duration for each video is about two minutes. In our work, the Combined dataset is divided into two parts, i.e., 100 videos for training and 26 videos for testing.

The second dataset is VTW [35]. It is much bigger than previous three datasets, which consists of 2529 videos with the average duration of 1.5 minutes. These videos are downloaded form the YouTube website in open domain. Each video is annotated with the locations of human selected key shots. For training, the annotations are transfered to scores, where key shots are scored with 1 and others are with 0. In our work, the VTW dataset is split into a training set of 2000 videos and a testing set of 529 videos.

Although all the above four datasets provides the summary labels of each video, only the CoSum dataset provides the shot boundary labels. The SumMe dataset and VTW dataset just has the boundaries of key shots. In this case, to augment the training samples, for CoSum dataset, any two adjacent shots are used for training. For SumMe dataset and VTW dataset, a random number of frames surrounding the shot boundary are cut from the video, and the generated segments are taken as the training samples. TVsum is not considered in shot boundary detection, because its shots are manually generated by fixed length segmentation.

4.1.2 Feature Extraction

To analyze the influence of different features to the performance, both shallow feature and deep feature are extracted for the frame. Specifically, the shallow feature is extracted by the concatenation of color histogram, SIFT and optical flow [38]. They are widely used in video summarization, and each of them captures the appearance, key point and motion information in the frame, respectively. Besides, the deep feature is extracted from the fc7 layer of the VGGnet-16, which is the most popular CNN feature in computer vision tasks [28].

4.1.3 Evaluation Metrics

The quality of generated summaries are evaluated by their similarities to human-created reference summaries. In this paper, the similarity is measured by three popular metrics, i.e., precision (P), recall (R) and F-measure (F), where

$$P = \frac{\#\text{correct shots}}{\#\text{generated shots}}, \quad R = \frac{\#\text{correct shots}}{\#\text{reference shots}},$$

F is the harmonic mean of P and R. The three metrics are most frequently used in the video summarization task. Note that one shot in the generated summary is considered as the correct shot if there is a reference shot sharing more than half duration overlap with it.

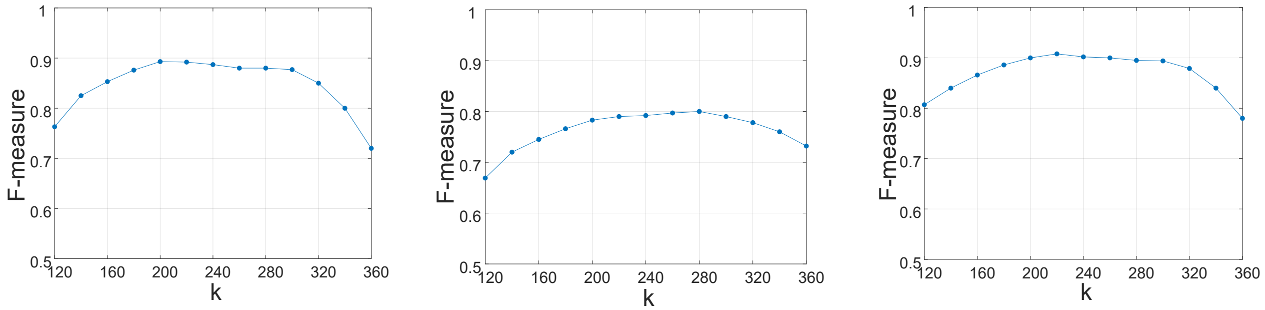
Besides, we also provide the results of shot boundary detection in this paper. The evaluation also employs the above three metrics. In particular, a shot is taken as correctly segmented if the interval between the detected boundary and annotated boundary is less than 10 frames. The rationality lying behind is that, compared to the length of the shot, the difference caused by 10 frames is almost invisible.

4.2. Results and Discussion

One of the main novelties of our approach is to integrate video structure exploitation (i.e., shot boundary detection) and video summarization into one architecture. Therefore, to verify the effectiveness of our approach, the results on shot boundary detection and video summarization are discussed, respectively.

4.2.1 Results of Shot Boundary Detection

To verify the performance of our approach on shot boundary detection, the results of various approaches on SumMe, CoSum and VTW are presented in Table 1. Generally, the compared approaches can be roughly divided into two categories, i.e., non-RNN-based (the first five) and RNN-based (the last four). For a fair comparison, all RNN-based approaches are equipped with the same shallow feature and deep feature. But for non-RNN-based approaches, the features are extracted by the methods reported in their original



(a) Results on the SumMe dataset (b) Results on the CoSum dataset (c) Results on the VTW dataset

Figure 3. Distributions of the shot boundary detection results varying with the length of the sliding bidirectional LSTM.

Table 1. The results (F-measure) of various approaches on shot boundary detection. (The scores in bold indicate the best values.)

Feature	shallow feature			deep feature		
Datasets	SumMe	CoSum	VTW	SumMe	CoSum	VTW
Super Frame [12]	–	0.405	–	–	–	–
KTS [27]	–	0.412	–	–	0.421	–
Frame Similarity [2]	0.506	0.397	0.502	–	–	–
Hierarchical Clustering [3]	0.525	0.414	0.512	0.549	0.423	0.546
FCNN [11]	–	–	–	0.871	**0.795**	0.893
Multiscale RNN [6]	0.838	0.740	0.865	0.846	0.752	0.869
Boundary-aware RNN [4]	0.824	0.738	0.871	0.826	0.753	0.873
Sliding single LSTM	0.841	0.732	0.872	0.845	0.750	0.875
Sliding bidirectional LSTM	**0.864**	**0.774**	**0.891**	**0.887**	0.792	**0.902**

papers, since most of them are quite feature-dependent. In Table 1, we can clearly see that, for most approaches, they perform better with deep feature than with shallow feature. Besides, the results on CoSum are lower than the other two datasets. It is because that the shot length in CoSum varies largely, from dozens to hundreds of frames, which makes the shot boundary detection more challenging. Fortunately, our approach can still get satisfactory results, which indicates the effectiveness of the sliding bidirectional LSTM. Note that the results of most compared approaches are reproduced by the released source code, except for *Frame Similarity* and *FCNN*. They are implemented by ourselves, since their source codes are not available..

In Table 1, The first five approaches are non-RNN-based, where *Super Frame* and *KTS* detect shot boundaries by the motion magnitude and change point in the frame sequence. They are widely used in existing summarization approaches. *Frame Similarity* and *Hierarchical Clustering* segment the video by grouping similar frames into the same cluster chronologically. Strategies similar to the two approaches are frequently used in the shot segmentation of summarization tasks. The poor performance of the above four approaches indicates the necessity of developing professional tools for exploiting the video structure before summarization. *FCNN* is a fully convolutional neural network that specially designed for the shot boundary detection task and has achieved state-of-the-art results. We can see that our sliding bidirectional LSTM gets comparable performance with *FCNN*, which demonstrates the superiority of our approach in shot boundary detection.

In Table 1, the last four are RNN-based approaches. In fact, *Multiscale RNN* and *Boundary-aware RNN* are not originally designed for the shot boundary detection or video summarization tasks. But they are capable of discovering the latent hierarchical structure of sequences. In this case, they are modified to detect the shot boundaries in this part. Specifically, *Multiscale RNN* and *Boundary-aware RNN* share similar architectures that they apply a long single LSTM to the video, and detect the shot boundary by a threshold of the LSTM output at each step. Once the boundary is detected, the hidden state and memory cell of the next LSTM unit is re-initialized. It can be observed that they get comparable results with the baseline of our approach, i.e., *Sliding single LSTM*, where the bidirectional LSTM is replaced by a single LSTM. But our *Sliding bidirectional LSTM* performs significantly better than them, which indi-

Table 2. The summarization results (F-measure) of various approaches on the Combined dataset. (The scores in bold indicate the best values.)

Feature	shallow feature			deep feature		
Datasets	SumMe	TVsum	CoSum	SumMe	TVsum	CoSum
VSUMM [8]	0.328	0.390	0.407	0.335	0.391	0.412
LiveLight [39]	0.357	0.460	0.525	0.384	0.477	0.511
CSUV [12]	0.393	0.532	–	–	–	–
LSMO [13]	0.397	0.548	–	0.403	0.568	–
Summary Transfer [36]	0.397	0.543	0.636	0.409	0.541	0.653
vsLSTM [37]	0.370	0.534	0.638	0.416	0.579	0.644
dppLSTM [37]	0.407	0.579	0.644	0.429	0.596	0.655
Hierarchical RNN [38]	0.394	0.566	0.656	0.411	0.577	0.663
HSA-RNN	**0.425**	**0.584**	**0.682**	**0.441**	**0.598**	**0.692**

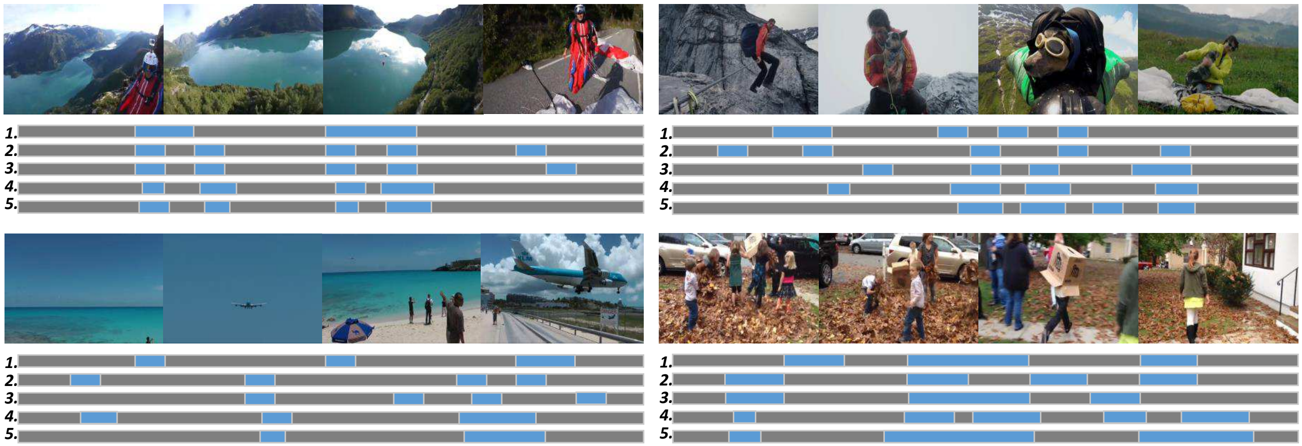


Figure 4. Four exemplar results from the Combined dataset. Each video is depicted by four frames. The five bars below each video represent the summaries generated by *vsLSTM*, *dppLSTM*, *Hierarchical RNN*, *HSA-RNN* and human beings, respectively. Specifically, the long gray bar stands for the whole video stream, and the short blue bar denotes the selected key shot.

cates the necessity of both forward and backward information in shot boundary detection.

In Figure 3, the analysis of the hyper parameter k (the length of the sliding bidirectional LSTM) is presented. It can be observed that the performance of our approach rises when $k < 200$ and begins to decline when $k > 300$, and there is a long steady stage when $200 < k < 300$. It shows that, to a certain extent, the performance of our approach is robust to the variance of k.

4.2.2 Video Summarization on the Combined Dataset

Table 2 shows the performance of different approaches on the Combined dataset. The compared approaches are separated into two parts by the double horizontal line in the middle, where the above five are non-RNN-based and the below four are RNN-based. Note that, in Table 2, the results of the above five approaches are reported in the literature, and the last three compared approaches are tested with their source codes. For the above five, they are all popular approaches of different types. Specifically, *VSUMM* and *LiveLight* are based on clustering and dictionary learning, respectively. *CSUV* and *LSMO* utilize property models to measure the importance, representativeness and diversity of each shot, and then summarize the video according to the shot scores. *Summary Transfer* generates the summary with category labels of videos, and gets the best results among non-RNN-based approaches. However, our approach get comparable results, even without the category information of videos.

In Table 2, the last four approaches are RNN-based. *vsLSTM* is the first approach that introduces LSTM to the video summarization task. It applies a long plain bidirectional LSTM to the whole video and predicts the shot importance with a *Multi-Layer Perception* (MLP). However, videos for summarization usually contain thousands of frames. It is really hard to train such a long LSTM, let

Table 3. The summarization results of various approaches on the VTW dataset. (The scores in bold indicate the best values.)

Feature	shallow feature			deep feature		
Metrics	Precision	Recall	F-measure	Precision	Recall	F-measure
CSUV [12]	0.367	0.423	0.393	–	–	–
HD-VS [33]	–	–	–	0.392	0.483	0.433
vsLSTM [37]	0.388	0.490	0.433	0.397	0.495	0.441
Hierarchical RNN [38]	0.408	0.516	0.456	0.417	0.525	0.465
HSA-RNN	**0.434**	**0.537**	**0.480**	**0.443**	**0.548**	**0.491**

alone with MLP. Worse still, it has been reported in [26] that such a long LSTM weakens its capability in temporal dependency exploitation, which is essential to video summarization. *dppLSTM* is derived from *vsLSTM* by a *Determinatal Point Process* model to measure the diversity among key shots. Although it improves the performance, the generalization of *vsLSTM* is reduced, since some of the video summaries don't meet the diversity constraints, such as the VTW dataset. Besides, both *vsLSTM* and *dppLSTM* ignore the latent hierarchical structure of the video data.

Actually, *Hierarchical RNN* is the one most similar to our approach. It develops a two-layer architecture to capture the intra-shot and inter-shot temporal dependency separately. The better performance than *vsLSTM* reflects the effectiveness of this hierarchical structure. But *Hierarchical RNN* fails to exploit the video structure before summarizing the video. Specifically, the shots in *Hierarchical RNN* are generated by fixed length segmentation, which mainly has two drawbacks: 1) it will destroy the inherent intra-shot temporal dependency and obstruct the first layer LSTM to understand the video structure. 2) it will break the integrity of activities captured in each shot, and then reduce the summary quality. Fortunately, the proposed HSA-RNN can make up these drawbacks, which has been verified by the results in Table 2.

Finally, Figure 4 presents some exemplar summaries generated by RNN-based approaches, i.e., *vsLSTM*, *dppLSTM*, *Hierarchical RNN*, and *HSA-RNN*. It can be observed that the summaries generated by *HSA-RNN* show the highest similarity with the human generated summaries. Moreover, the durations of selected key shots are close to those of manually generated shots, which means less information loss or mixture in the key shots. Overall, it demonstrates the superiority of *HSA-RNN* in video summarization.

4.2.3 Video Summarization on the VTW Dataset

Table 3 presents the results of different approaches on the VTW dataset. Actually, quite a lot of existing approaches are dataset dependent. For a fair comparison, only the results of those approaches most suitable for VTW are listed in this part.

CSUV, *vsLSTM* and *Hierarchical RNN* have been introduced in Table 1. *HD-VS* adopts two-stream CNNs to summarize the video, where AlexNet [16] and C3D [30] are employed to extract the appearance and temporal information, respectively. Benefiting from the super ability of CNN in visual feature extraction, it gets the best performance among non-RNN-based approaches on VTW.

In Table 3, it can be observed that *vsLSTM* and *Hierarchical RNN* perform better than *HD-VS*. It indicates that, benefiting from the capability of sequence modeling, LSTM is more suitable for the video summarization task. Furthermore, the even better performance of the proposed HSA-RNN exhibits its improvements on video summarization by exploiting the video structure.

Overall, the experimental results in Table 1, 2 and 3 have verified the effectiveness of our approach in three aspects: 1) the structure-adaptive architecture. It makes up the gap between structure exploitation and video summarization, and improves the summary quality effectively. 2) the hierarchical structure. It increases the long temporal dependency capture ability of LSTM, meanwhile, reduces the computation operations significantly. 3) the bidirectional LSTM. Both forward and backward information are important to shot boundary detection and video summarization.

5. Conclusion

In this paper, we propose a Hierarchical Structure-Adaptive RNN (HSA-RNN) for the video summarization task, which can adaptively exploit the video structure and generate the summary simultaneously. Specifically, it contains two layers, constructed by bidirectional LSTMs. The first layer is utilized to segment the video into several shots and extract shot features. The second layer is designed to capture the forward and backward temporal dependencies among shots, whose outputs are employed to predict the probability of each shot to be selected into the summary. To our knowledge, it is the first approach that integrates structure exploitation and video summarization into one end-to-end architecture. The results on four popular datasets have verified the effectiveness of our approach in both shot boundary detection and video summarization.

References

[1] A. Aner and J. R. Kender. Video summaries through mosaic-based shot and scene clustering. In *Computer Vision - ECCV 2002, 7th European Conference on Computer Vision, Copenhagen, Denmark, May 28-31, 2002, Proceedings, Part IV*, pages 388–402, 2002.

[2] E. E. Apostolidis and V. Mezaris. Fast shot segmentation combining global and local visual descriptors. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pages 6583–6587, 2014.

[3] L. Baraldi, C. Grana, and R. Cucchiara. Shot and scene detection via hierarchical clustering for re-using broadcast video. In *Computer Analysis of Images and Patterns - 16th International Conference, CAIP 2015, Valletta, Malta, September 2-4, 2015 Proceedings, Part I*, pages 801–811, 2015.

[4] L. Baraldi, C. Grana, and R. Cucchiara. Hierarchical boundary-aware neural encoder for video captioning. *CoRR*, abs/1611.09312, 2016.

[5] W. Chu, Y. Song, and A. Jaimes. Video co-summarization: Video summarization by visual co-occurrence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3584–3592, 2015.

[6] J. Chung, S. Ahn, and Y. Bengio. Hierarchical multiscale recurrent neural networks. *CoRR*, abs/1609.01704, 2016.

[7] Y. Cong, J. Yuan, and J. Luo. Towards scalable summarization of consumer videos via sparse dictionary selection. *IEEE Trans. Multimedia*, 14(1):66–75, 2012.

[8] S. E. F. de Avila, A. P. B. Lopes, A. da Luz Jr., and A. de Albuquerque Araújo. VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011.

[9] E. Elhamifar, G. Sapiro, and R. Vidal. See all by looking at a few: Sparse modeling for finding representative objects. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1600–1607, 2012.

[10] B. Gong, W. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2069–2077, 2014.

[11] M. Gygli. Ridiculously fast shot boundary detection with fully convolutional neural networks. *CoRR*, abs/1705.08214, 2017.

[12] M. Gygli, H. Grabner, H. Riemenschneider, and L. V. Gool. Creating summaries from user videos. In *European Conference on Computer Vision*, pages 505–520, 2014.

[13] M. Gygli, H. Grabner, and L. Van Gool. Video summarization by learning submodular mixtures of objectives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3090–3098, 2015.

[14] Y. Hadi, F. Essannouni, and R. O. H. Thami. Video summarization by k-medoid clustering. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 1400–1401. ACM, 2006.

[15] A. Khosla, R. Hamid, C. Lin, and N. Sundaresan. Large-scale video summarization using web-image priors. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 2698–2705, 2013.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114, 2012.

[17] Y. J. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1346–1353, 2012.

[18] X. Li, Z. Wang, and X. Lu. Surveillance video synopsis via scaling down objects. *IEEE Trans. Image Processing*, 25(2):740–755, 2016.

[19] T. Liu and J. R. Kender. Optimization algorithms for the selection of key frame sequences of variable length. In *Computer Vision - ECCV 2002, 7th European Conference on Computer Vision, Copenhagen, Denmark, May 28-31, 2002, Proceedings, Part IV*, pages 403–417, 2002.

[20] S. Lu, Z. Wang, T. Mei, G. Guan, and D. D. Feng. A bag-of-importance model with locality-constrained coding based feature learning for video summarization. *IEEE Trans. Multimedia*, 16(6):1497–1509, 2014.

[21] Z. Lu and K. Grauman. Story-driven summarization for egocentric video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2714–2721, 2013.

[22] Q. Luan, M. Song, C. Y. Liau, J. Bu, Z. Liu, and M. Sun. Video summarization based on nonnegative linear reconstruction. In *IEEE International Conference on Multimedia and Expo*, pages 1–6, 2014.

[23] B. Mahasseni, M. Lam, and S. Todorovic. Unsupervised video summarization with adversarial LSTM networks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, pages 1–10, 2017.

[24] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *Int. J. on Digital Libraries*, 6(2):219–232, 2006.

[25] C. Ngo, Y. Ma, and H. Zhang. Automatic video summarization by graph modeling. In *9th IEEE International Conference on Computer Vision (ICCV 2003), 14-17 October 2003, Nice, France*, pages 104–109, 2003.

[26] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition,*, pages 1029–1038, 2016.

[27] D. Potapov, M. Douze, Z. Harchaoui, and C. Schmid. Category-specific video summarization. In *European Conference on Computer Vision*, pages 540–555, 2014.

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[29] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes. Tvsum: Summarizing web videos using titles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5179–5187, 2015.

[30] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. C3D: generic features for video analysis. *CoRR*, abs/1412.0767, 2014.

[31] H. Yang, B. Wang, S. Lin, D. P. Wipf, M. Guo, and B. Guo. Unsupervised extraction of video highlights via robust recurrent auto-encoders. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 4633–4641, 2015.

[32] T. Yao, T. Mei, and Y. Rui. Highlight detection with pairwise deep ranking for first-person video summarization. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[33] T. Yao, T. Mei, and Y. Rui. Highlight detection with pairwise deep ranking for first-person video summarization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition,*, pages 982–990, 2016.

[34] W. Zaremba and I. Sutskever. Learning to execute. *CoRR*, abs/1410.4615, 2014.

[35] K. Zeng, T. Chen, J. C. Niebles, and M. Sun. Title generation for user generated videos. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, pages 609–625, 2016.

[36] K. Zhang, W. Chao, F. Sha, and K. Grauman. Summary transfer: Exemplar-based subset selection for video summarization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition,*, pages 1059–1067, 2016.

[37] K. Zhang, W. Chao, F. Sha, and K. Grauman. Video summarization with long short-term memory. In *Computer Vision - ECCV 2016 - 14th European Conference*, pages 766–782, 2016.

[38] B. Zhao, X. Li, and X. Lu. Hierarchical recurrent neural network for video summarization. In *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017*, pages 863–871, 2017.

[39] B. Zhao and E. P. Xing. Quasi real-time summarization for consumer videos. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 2513–2520, 2014.

[40] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. In *Proceedings of the 1998 IEEE International Conference on Image Processing, ICIP-98, Chicago, Illinois, October 4-7, 1998*, pages 866–870, 1998.