Video Captioning with Tube Features

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 710072, P. R. China
[2]Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, P. R. China
binzhao111@gmail.com, xuelong_li@opt.ac.cn, luxiaoqiang@opt.ac.cn

Abstract

Visual feature plays an important role in the video captioning task. Considering that the video content is mainly composed of the activities of salient objects, it has restricted the caption quality of current approaches which just focus on global frame features while paying less attention to the salient objects. To tackle this problem, in this paper, we design an object-aware feature for video captioning, denoted as tube feature. Firstly, Faster-RCNN is employed to extract object regions in frames, and a tube generation method is developed to connect the regions from different frames but belonging to the same object. After that, an encoder-decoder architecture is constructed for video caption generation. Specifically, the encoder is a bi-directional LSTM, which is utilized to capture the dynamic information of each tube. The decoder is a single LSTM extended with an attention model, which enables our approach to adaptively attend to the most correlated tubes when generating the caption. We evaluate our approach on two benchmark datasets: MSVD and Charades. The experimental results have demonstrated the effectiveness of tube feature in the video captioning task.

1 Introduction

Vision and language are two most important ways for human beings to perceive the world. Video captioning is one of the techniques that bridge them together [Pan *et al.*, 2016a; Li *et al.*, 2017]. Specifically, the goal of video captioning is to describe the video content with natural language text. It is an important task with wide applications, such as text-based video retrieval, human-robot interaction and so on.

Earlier works on video captioning are mostly template-based, which first detect the semantic content (i.e., subjective, verb, objective), and then generate the caption based on a sentence template [Guadarrama *et al.*, 2013; Thomason *et al.*, 2014]. However, these template-based approaches are inferior in modeling the rich information of natural language. Recently, inspired by the great success of *Recurrent Neural Network* (RNN) in language modeling [Luong *et al.*, 2015; Kalchbrenner and Blunsom, 2013], RNN-based approaches are introduced to video captioning and have achieved exciting results [Venugopalan *et al.*, 2015b]. Generally, given the visual feature as input, RNN is utilized to generate the caption word by word.

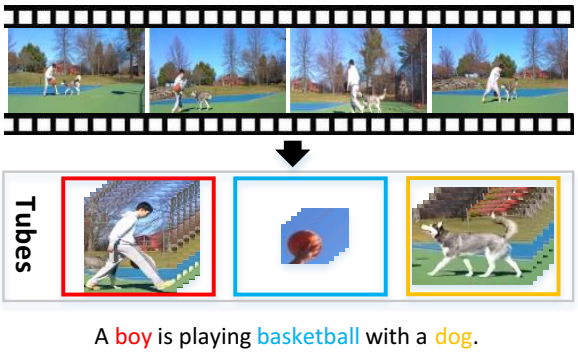

Figure 1: Tube features capture the dynamic information of objects. Moreover, the proposed approach can adaptively attend to the most correlated tubes when generating a certain word. The tubes bounded in different colors denote the most emphasized tube when generating the word in the same color.

As the caption is automatically generated based on the video content, visual feature is quite important to the caption quality [Baraldi *et al.*, 2017; Pan *et al.*, 2016a]. Initially, the visual feature for video captioning is directly extended from the image captioning task, i.e., average pooling the deep frame features [Venugopalan *et al.*, 2015b]. Later, to extract more correlated visual features, attention models are developed to selectively focus on a subset of video frames [Long *et al.*, 2016; Hori *et al.*, 2017; Li *et al.*, 2017]. However, these feature extraction methods focus on the global information, but the importance of salient objects is overlooked, which may restrict the accuracy of generated captions. Based on this point, in this paper, we design tube features to extract object information for the task of video captioning.

Actually, as depicted in the bottom of Figure 1, a tube reflects the trajectory of corresponding object. Firstly, Faster-RCNN [Ren *et al.*, 2015] is employed to detect the objects

in each frame. Then, these object regions are sequentially linked to be tubes. Besides, to avoid the loss of global frame information, all frames are linked as an additional tube. After tubes are generated, an encode-decoder architecture is designed to construct our network, where the encoder is a bi-directional LSTM, which is utilized to capture the dynamic information of objects by encoding each tube feature into a fix-sized visual vector, and the decoder LSTM is a single LSTM, used to generate the caption word by word. Specifically, at each time step, all the visual vectors and the text feature of previous word are input to the decoder LSTM. Besides, an attention model is designed to adaptively emphasize different tubes when generating a certain word.

To our knowledge, this is the first approach that extracts object-aware features for the task of video captioning. It is capable of reducing the interference caused by irrelevant background information and improving the accuracy of generated captions. Moreover, the experimental results on the MSVD dataset [Guadarrama *et al.*, 2013] and Charades dataset [Sigurdsson *et al.*, 2016] have verified the effectiveness of the proposed approach.

2 Our Approach

In this paper, we develop a video caption generator conditioning on tube features. In the following, we first present the procedure of tube construction, and then introduce the generation of video caption.

2.1 Tube Generation

In general, tubes are formed by the trajectories of objects. Firstly, the Faster-RCNN [Ren *et al.*, 2015] is employed to detect objects in the video. After that, tubes are generated by sequentially linking the detected object regions (i.e., bounding box). In other words, each object tube is composed of those regions in different frames but belonging to the same object.

After bounding boxes are detected, a similarity graph is constructed among them. As depicted in Figure 2, the edge between each pair of bounding boxes in adjacent frames are labeled with a similarity score which indicates the confidence that they belong to the same object. Without loss of generality, suppose that r_j and r_{j+1} are two regions in adjacent frames j and $j+1$, the similarity score s is computed as:

$$s\left(r_j, r_{j+1}\right) = \lambda\left(Dis\_ind + Area\_ind + Iou\_ind\right), \quad (1)$$

where s is a comprehensive index that jointly considers the relationships of distance, area, and *Intersection Overlap Union* (IOU) between region r_j and r_{j+1}, and λ is a scalar to normalize s to $(0, 1)$. Specifically, the three terms on the right of Equ. (1) are defined as follows:

1) Dis_ind. It is designed to exploit the location relationship between two regions, a higher Dis_ind indicates that two regions are closer to each other. The distance of two regions is determined by the Euclidean distance of their geometrical center, denoted as $dis\left(r_j, r_{j+1}\right)$. Then, Dis_ind is defined as:

$$Dis\_ind = \exp\left\{-\frac{dis\left(r_j, r_{j+1}\right)}{\sigma}\right\}, \quad (2)$$

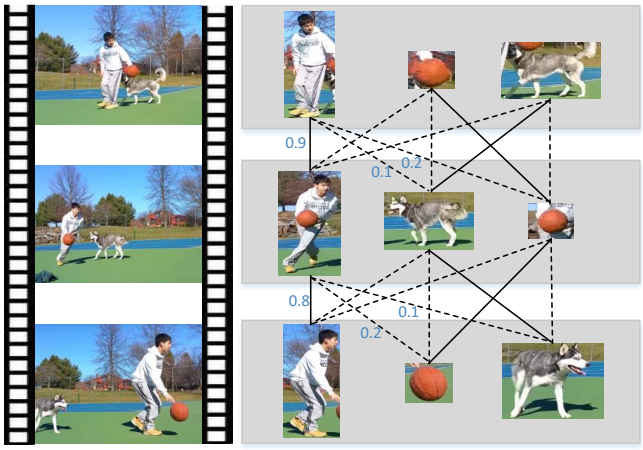

Figure 2: The similarity graph constructed by regions in different frames. As depicted, there is an edge between any two regions in adjacent frames, and the scores of each edge (partially displayed for brevity) denote the confidence of the two regions belong to the same object tube. The solid edge means two regions are assigned to the same tube.

where σ denotes the maximum Euclidean distance among regions of frame j and $j+1$.

2) $Area\_ind$. It represents the difference between the area sizes of two regions, a higher $Area\_ind$ means two regions have similar size. It is formulated as:

$$Area\_ind = \exp\left\{-\left|\frac{\min\left(area\left(r_j\right), area\left(r_{j+1}\right)\right)}{\max\left(area\left(r_j\right), area\left(r_{j+1}\right)\right)} - 1\right|\right\}, \quad (3)$$

where $area\left(\cdot\right)$ denotes the region size.

3) Iou_ind. It stands for the rate of the overlapping area between two regions, which is defined as:

$$Iou\_ind = \frac{area\left(r_j\right) \cap area\left(r_{j+1}\right)}{area\left(r_j\right) \cup area\left(r_{j+1}\right)}. \quad (4)$$

After the similarity graph is constructed, the tubes are generated by finding the optimal path in the graph, which is formulated as:

$$tube^i = \arg\max_{S^i} \sum_{j}^{J-1} s\left(r_j, r_{j+1}\right) \quad (5)$$

where $tube^i = [r_1, r_2, \ldots, r_J]$ is the linked region sequence for tube i. In Equ. (5), the tubes are generated iteratively by dynamic programming [Gkioxari and Malik, 2015]. When one tube is generated, the edges correlated to these regions are removed from the similarity graph. Specifically, S^i denotes the similarity graph at the t-th iteration. Equ. (5) is repeated until the region set is empty, and the object tubes are obtained finally. Besides, to include the global information of frames, the frame features are taken as an additional tube and added to the tube set.

After tubes generated, a bidirectional LSTM is employed to capture the dynamic information by encoding each tube into a fix-sized visual vector. In fact, the bidirectional LSTM consists of a forward LSTM and a backward LSTM, where

the main difference is that the backward LSTM operates reversely. In this case, the encoding can be formulated as:

$$h_j^f = LSTM\left(tube_j^i, h_{j-1}^f\right), \tag{6}$$

$$h_j^b = LSTM\left(tube_{J-j}^i, h_{j-1}^b\right), \tag{7}$$

where $LSTM(\cdot)$ is the short form of the calculations in each LSTM unit [Hochreiter and Schmidhuber, 1997]. h_j^f and h_j^b denote the hidden state of the forward LSTM and backward LSTM, respectively. Finally, the tube feature τ^i is defined as the concatenation of the final forward and backward hidden state, i.e.,

$$\tau^i = \left[h_J^f, h_J^b\right], \tag{8}$$

where $[\cdot,\cdot]$ stands for the vector concatenation operation.

2.2 Video Caption Generation

After the encoding of tube features, another LSTM is employed as the decoder to generate the video caption. At each time step, the hidden layer of the decoder LSTM is generated by:

$$h_t = LSTM\left([v_t, e_{t-1}], h_{t-1}\right), \tag{9}$$

where v_t denotes the input visual feature at time step t, and e_{t-1} is the text feature of previous word. Given the hidden state h_t , a probability distribution over the vocabulary is calculated by:

$$p_t = softmax\left(U_p \tanh\left(W_p\left[v_t, e_{t-1}, h_t\right] + b_p\right)\right), \tag{10}$$

where W_p, U_p and b_p are the training parameters. p_t denotes the predicted probability of each word in the vocabulary to be selected, its dimensionality is equal to the vocabulary size. Once Equ. (10) is computed, the t-th word of the caption is selected from the vocabulary by the maximum value in p_t.

It can be observed from Equ. (10) that p_t is determined jointly by current visual feature v_t, previous text feature e_{t-1}, together with the history information captured in h_t. In this paper, the text feature is extracted for each word by embedding its one-hot feature into the same space as the visual feature, and the visual feature is generated by applying attention model to the tube visual vectors $\left\{\tau^i\right\}_{i=1}^n$, here n denotes the number of tubes.

Essentially, the attention model is designed to make the decoder LSTM adaptively attend to the most correlated visual features. At each time step, the visual feature input to the LSTM (i.e., v_t in Equ. (9)) is a weighted combination of the tube visual vectors,

$$v_t = \sum_{i=1}^{n} \alpha_t^i \tau^i, \tag{11}$$

where α_t^i stands for the attention weight of the i-th tube when generating the t-th word of the caption. It can be observed from Equ. (11) that a higher attention weight means the corresponding tube is more emphasized in composing the visual feature.

According to [Yao *et al.*, 2015], α_t^i represents the relevance of the i-th tube given all the words generated previously, i.e., $e_1, \ldots, e_{t-1}$. Fortunately, except e_{t-1}, they are recorded by the previous hidden state h_{t-1} of the decoder LSTM. Thus, the relevance score is calculated by:

$$l_t^i = w^{\mathrm{T}} \tanh\left(W_\alpha\left[\tau^i, e_{t-1}, h_{t-1}\right] + b_\alpha\right), \tag{12}$$

where w, W_α and b_α are the parameters to be learned, and

$$h_{t-1} = LSTM\left([v_{t-1}, e_{t-2}], h_{t-2}\right). \tag{13}$$

Then, the attention weight α_t^i is obtained after normalizing l_t^i by:

$$\alpha_t^i = \exp\left(l_t^i\right) / \sum_j^n \exp\left(l_t^j\right). \tag{14}$$

Note that α_t^i is a dynamic attention weight that changes in different time steps, so that our attention model can adaptively focus on different tubes when generating each word.

3 Experiments

3.1 Setup

Datasets

The MSVD [Guadarrama *et al.*, 2013] dataset contains 1970 video clips collected from YouTube. These clips are open-domain, including different scenes, activities and scenarios, *etc*. In total, there are about 80839 video captions annotated by Mechanical Turkers. On average, each video contains 41 captions, and each caption has about 8 words. In this paper, the dataset is split into three sets, 1200 videos for training, 100 videos for validation, and the remaining 670 videos for testing.

The Charades dataset [Sigurdsson *et al.*, 2016] is much more challenging, which is composed of 9848 videos, and each video records a sequence of daily living activities in indoor environment, including cooking, cleaning, watching TV, *etc*. Exactly, this dataset provides 16130 video captions, and each caption contains 23 words on average. In this paper, the dataset is split into a training set of 7569 videos, a validation set of 400 videos, and a testing set of 1863 videos.

Feature Extraction

In this paper, the Faster-RCNN is utilized to detect objects and extract their features. Practically, to analyze the influence of different features, the Faster-RCNN is trained on PASCAL VOC 2007 and MicroSoft COCO 2014 on the basis of VGG-16 [Simonyan and Zisserman, 2014] and Resnet-101 [He *et al.*, 2016], respectively.

To extract text features, we first build the vocabulary for each dataset. In this paper, all the captions are processed by tokenizing the sentences, lowercasing the words, and removing rare words. After that, we obtain a vocabulary of 2743 words for the MSVD dataset, and 1525 words for the Charades dataset, respectively. Finally, the words are extracted with one-hot features (1-of-N coding, N denotes the vocabulary size) and then embedded into 300-dimensional GloVe vectors [Pennington *et al.*, 2014].

Training

Given the reference captions, the proposed approach is trained by maximizing the log-likelihood function:

$$\Theta = \arg\max_{\Theta} \sum_{k=1}^{N} \sum_{t=1}^{T} \log P\left(g_t^k | g_{1:t-1}^k, v_t^k, \Theta\right), \tag{15}$$

Metrics	CIDEr	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
FGM [Thomason *et al.*, 2014]	–	–	–	–	0.137	–	0.239
Mean Pool [Venugopalan *et al.*, 2015b]	–	–	–	–	0.372	–	0.281
S2VT [Venugopalan *et al.*, 2015a]	0.486	0.735	0.593	0.482	0.369	0.652	0.289
SA [Yao *et al.*, 2015]	0.481	0.741	0.589	0.482	0.366	0.647	0.294
LSTM-E [Pan *et al.*, 2016b]	–	0.749	0.609	0.506	0.402	–	0.295
p-RNN [Yu *et al.*, 2016]	–	0.773	0.645	0.546	**0.443**	–	0.311
HRNE [Pan *et al.*, 2016a]	–	0.784	0.661	0.551	0.436	–	0.321
HBA [Baraldi *et al.*, 2017]	–	0.772	0.663	0.541	0.418	–	0.318
MAM-RNN [Li *et al.*, 2017]	**0.539**	**0.801**	0.661	0.547	0.413	0.688	0.322
Ours	0.522	0.776	**0.671**	**0.554**	0.438	**0.693**	**0.326**

Table 1: The results of various approaches on MSVD dataset. (The best results are **bold**, the second-best are underlined)

here Θ stands for all the training parameters in our approach. Besides, N is the size of training set, g_t^k stands for the t-th word in the reference caption of training video k, and T denotes the total time steps of the decoder LSTM. Following existing approaches [Venugopalan *et al.*, 2015a], T is fixed as 80 in this paper, which is bigger than the longest caption.

Evaluation Metrices

In this paper, four metrices are employed to evaluate the quality of generated captions, they are BLEU [Papineni *et al.*, 2002], ROUGE-L [Lin and Och, 2004], CIDEr [Vedantam *et al.*, 2015], METEOR [Denkowski and Lavie, 2014], where BLEU has four versions, i.e., BLEU 1-4. In fact, most of them are originally proposed for machine translation. Recently, they are also widely used in video captioning, since the evaluation of the two tasks are both the comparison between generated sentence and reference sentence. Practically, in video captioning, the higher scores of these metrices indicate the higher similarity between the generated caption and the reference caption, i.e., the higher accuracy of the generated caption.

3.2 Results and Discussion

Results on the MSVD Dataset

Table 1 shows the performance of various approaches on the MSVD dataset. It should be noted that, for a fair comparison, the results listed here are all generated with the baseline of static frame feature extractors. Concretely, HRNE are with GoogLeNet [Szegedy *et al.*, 2015], and others are with VGG-16 [Simonyan and Zisserman, 2014]. Besides, to our approach, the Faster-RCNN model employed in this part is on the basis of VGG-16, trained on MicroSoft COCO dataset.

In Table 1, all the compared approaches except FGM are based on RNN. Specifically, FGM is a template-based approach, which generates video captions based on a sentence template. It is a successful combination of visual feature and language statics, and has won the state-of-the-art performance in non-RNN based approaches. However, benefiting from the great ability of RNN in modeling sequence, it can be observed from Table 1 that RNN-based approaches get much better performance. Actually, the biggest difference between our approach and other RNN-based approaches lies in the encoding of visual features. Detailedly, Mean Pool encodes the visual feature by simply average pooling the frame features.

Metrics	METEOR
S2VT(RGB+FLOW VGG-16)	0.297
SA(GoogLeNet+C3D)	0.296
LSTM-E(C3D)	0.299
LSTM-E(VGG-16+C3D)	0.299
p-RNN (VGG-16+C3D)	0.303
HRNE (C3D)	0.310
HBA (GoogLeNet+C3D)	0.324
MAM-RNN (C3D)	0.325
Ours (VGG-16 on VOC)	0.328
Ours (VGG-16 on COCO)	0.326
Ours (Resnet-101 on VOC)	0.331
Ours (Resnet-101 on COCO)	**0.334**

Table 2: The results with different features on MSVD dataset.

SA and p-RNN are attention based approaches, which generate visual features by the weighted sum of frame features or region features. However, in the above three approaches, the temporal information of video features is missed. Our approach performs better than them because the temporal information is captured by the encoder LSTM. Similarly, S2VT also employs LSTM to encode the frame features. But they are conditioned on global frame features. The better results of our approach have verified the effectiveness of object-aware tube features. Moreover, HRNE, HBA and MAM-RNN design hierarchical architectures to model the frame sequence, which increase the model complexity. Fortunately, our approach gets comparable results with them. Besides, our approach also performs better than LSTM-E which jointly exploits the visual and semantic information by integrating a visual-semantic embedding space into LSTM. But its visual input to LSTM is just the average pooling of frame features, which restricts the performance.

In Table 2, the influence of different features is analyzed. Here, all the compared approaches are with C3D (3D convolutional neural network) [Tran *et al.*, 2015] or flow frames (generated by optical flow) [Venugopalan *et al.*, 2015a] to exploit the dynamic information of the video. Moreover, some of them are even fused with static frame features. Meanwhile, to our approach, four versions of feature extractor is provided, i.e., Faster-RCNN based on different CNNs and trained on different datasets. It can be observed that even though our

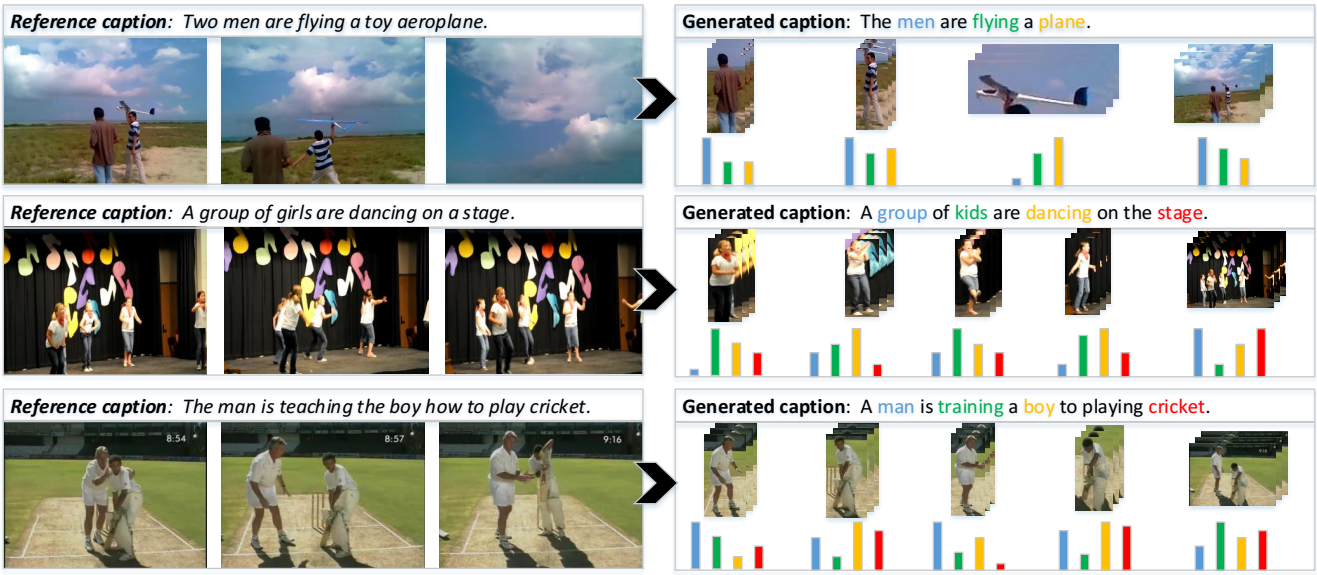

Figure 3: Example results of our approach on the MSVD dataset, where the video content is represented by three frames. The histograms below each tube denote the attention weight of that tube when generating different words. Actually, only the words that are heavily dependent on the visual information are considered here, and they are distinguished by different colors.

Metrics	METEOR
Ours (without frame-tube)	0.298
Ours (without attention model)	0.305
Ours (single LSTM encoder)	0.312
Ours (full)	0.326

Table 3: The results of different variants on MSVD dataset.

approach is not equipped with 3D convolutional network, all the four results are better than compared approaches. It is because the dynamic information is already captured by the encoder LSTM. Therefore, the results have shown the great advantages of our tube features. Besides, our approach has shown stable performance on the four feature extractors. Precisely, the results on Resnet-101 is slightly higher than VGG-16. It is because Resnet-101 is much deeper, which indicates stronger capability to exploit the visual information. Additionally, there is little difference between the results on PASCAL VOC and Microsoft COCO. Actually, Microsoft COCO contains 80 categories of objects, where most of the objects appearing in the video are included. However, PASCAL VOC just has 20 categories, which means the Faster-RCNN trained on this dataset may not capable of covering the diversity of the objects in the video. But, it can be observed that there is little decline in the results on PASCAL VOC. It is because that the detected object regions not only capture the information of an exact object, but also the common objectness. Besides, only the region feature is utilized to form the tube, their category label is not considered. Therefore, as long as the object regions are detected, whether they are classified accurately is not important to our approach.

In Table 3, to analyze the influence of different components, we test several variants of our approach. 1) The frame tube is deleted from the tube set, i.e., just object tubes are considered. It can be observed that the performance declined significantly. It indicates that the background information also makes contributions to the accuracy of video captions, and the global frame feature is as important as object-aware features in video captioning. 2) The attention model is dropped, such that the visual feature at each step is generated by average pooling the tube features. The worse performance verified the necessity of emphasizing the most correlated tube in the caption generation. 3) The bi-directional LSTM encoder is replaced with a single LSTM, so that only the forward dynamic information is considered. The worse results compared with our full model illustrate the advantages of bi-directional LSTM in dynamic information capturing. Overall, the presented results in Table 3 have verified the necessity of frame-tube, attention model and the bi-directional LSTM encoder of our approach.

Besides, in Figure 3, we provide some examples of our results on the MSVD dataset. Generally, most of the captions generated by our approach can describe the video content accurately. Moreover, when generating the key words, our approach can automatically attend to the most correlated tubes. Precisely, it can be observed from Figure 3 that there are duplicate tubes in the third row. It is caused by the wrong split of our tube generation method. This phenomenon occurs sometimes, especially when frames change rapidly. Fortunately, our attention model can reduce the interference caused by this mistake. Overall, in Figure 3, we can draw the conclusion that with the tube features, our approach shows great advantages in video captioning.

Results on the Charades Dataset

Table 4 presents the results on the Charades dataset. As aforementioned, it is a more challenging dataset, so the perfor-

Metrics	CIDEr	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
S2VT [Venugopalan *et al.*, 2015a]	0.140	0.490	0.300	0.180	0.110	0.160
SA [Yao *et al.*, 2015]	**0.181**	0.403	0.247	0.155	0.076	0.143
MAAM [Fakoor *et al.*, 2016]	0.167	0.500	0.311	0.188	0.115	0.176
MAM-RNN [Li *et al.*, 2017]	0.177	**0.530**	0.307	0.185	0.130	0.179
Ours (VGG-16 on VOC)	0.172	0.501	**0.315**	0.190	0.125	0.181
Ours (VGG-16 on COCO)	0.174	0.502	0.310	0.193	0.130	0.179
Ours (Resnet-101 on VOC)	0.178	0.509	0.308	0.196	**0.134**	**0.191**
Ours (Resnet-101 on COCO)	0.180	0.507	0.313	**0.197**	0.133	0.190

Table 4: The results of various approaches on Charades dataset.

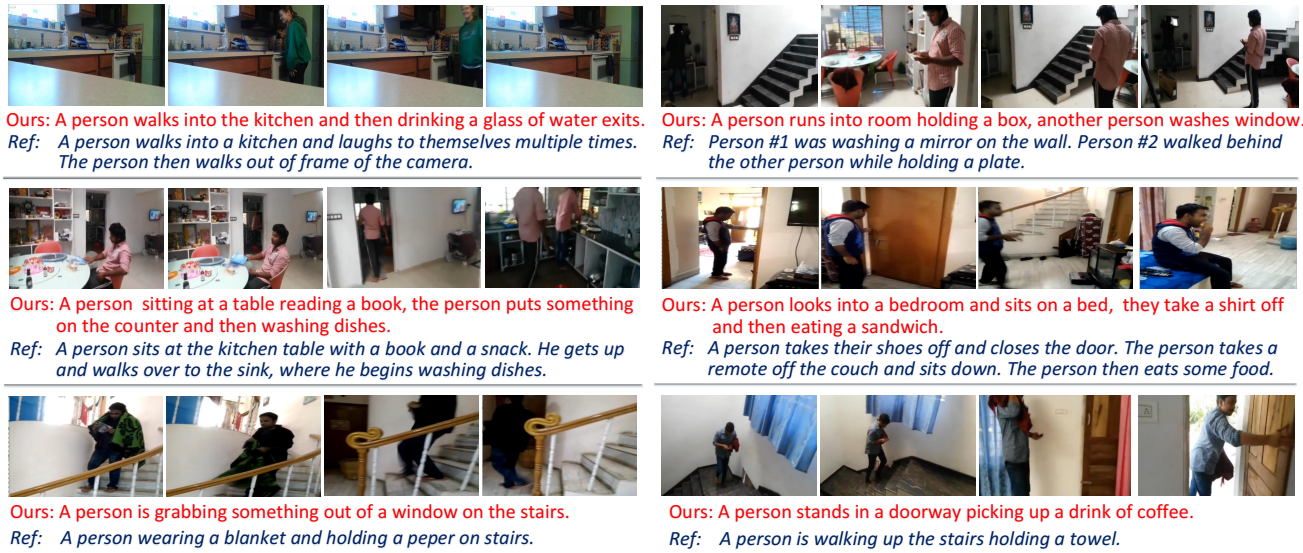

Figure 4: Example results of our approach on the Charades dataset, where each video is represented by four frames and the captions are displayed below. Specifically, the sentence in red denotes the caption generated by our approach, and the sentence in blue represents the reference caption annotated by human beings.

mance reflected on the evaluation metrics is much lower than that on the MSVD dataset. In Table 4, the results of four compared approaches are presented, where S2VT, SA and MAM-RNN have been introduced before, and MAAM designs a memory-augmented attention model to adaptively focus on a subset of frames when generating the caption. Compared to traditional attention model, like SA, the main novelty of MAAM is that the attention weight is determined jointly by the current visual feature and the memories of past attention. Practically, the better performance of MAAM than SA has shown its superiority. However, most of the four compared approaches neglect the object features. Therefore, our approach with tube features can get better performance than them. Besides, it can be observed from Table 4 that our approach with four different feature extractors gets comparable results, which indicates the stability of our approach.

In Figure 4, we present some examples of generated captions on Charades dataset. As aforementioned, Charades is a much challenging dataset. The primary reason is that its reference captions are much longer. From the reference captions and the frames displayed in Figure 4, it can be observed that some generated captions can basically describe the video content, but the accuracy is much lower than that on the MSVD dataset. Worse still, there are even some mistakes on the key words in the sentence, i.e., subjective, verb and objective. It is because there are a lot of small objects in the reference captions, such as phone, shoes, pillow and so on. Unfortunately, it is hard to capture the visual information of these small objects for both frame features and region features. In fact, it is a common problem for existing approaches. In the future, we plan to improve our approach to address this problem.

4 Conclusion

In this paper, we propose to generate video captions with tube features. Practically, it is an object-aware feature which captures the trajectories of objects in the video. In our approach, the Faster-RCNN is employed to first extract region proposals, and the regions belonging to the same objects are linked to be tubes. Then, a video caption generator is built following the encoder-decoder paradigm. Specifically, a bi-directional LSTM is employed to encoding both the forward and backward dynamic information of tubes, and a single LSTM extended with a tube-based attention model is utilized as the decoder to generate the caption word by word.

References

[Baraldi *et al.*, 2017] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. Hierarchical boundary-aware neural encoder for video captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3185–3194, 2017.

[Denkowski and Lavie, 2014] Michael J. Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Workshop on Statistical Machine Translation*, 2014.

[Fakoor *et al.*, 2016] Rasool Fakoor, Abdel-rahman Mohamed, Margaret Mitchell, Sing Bing Kang, and Pushmeet Kohli. Memory-augmented attention modelling for videos. *CoRR*, abs/1611.02261, 2016.

[Gkioxari and Malik, 2015] Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 759–768, 2015.

[Guadarrama *et al.*, 2013] Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Subhashini Venugopalan, Raymond J. Mooney, Trevor Darrell, and Kate Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *ICCV*, 2013.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8), 1997.

[Hori *et al.*, 2017] Chiori Hori, Takaaki Hori, Teng-Yok Lee, Ziming Zhang, Bret Harsham, John R. Hershey, Tim K. Marks, and Kazuhiko Sumi. Attention-based multimodal fusion for video description. In *ICCV*, pages 4203–4212, 2017.

[Kalchbrenner and Blunsom, 2013] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *EMNLP*, 2013.

[Li *et al.*, 2017] Xuelong Li, Bin Zhao, and Xiaoqiang Lu. MAM-RNN: multi-level attention model based RNN for video captioning. In *IJCAI*, 2017.

[Lin and Och, 2004] Chin-Yew Lin and Franz Josef Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *ACL*, 2004.

[Long *et al.*, 2016] Xiang Long, Chuang Gan, and Gerard de Melo. Video captioning with multi-faceted attention. *CoRR*, abs/1612.00234, 2016.

[Luong *et al.*, 2015] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *EMNLP*, 2015.

[Pan *et al.*, 2016a] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *CVPR*, 2016.

[Pan *et al.*, 2016b] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In *CVPR*, 2016.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[Sigurdsson *et al.*, 2016] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016.

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

[Szegedy *et al.*, 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[Thomason *et al.*, 2014] Jesse Thomason, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Raymond J. Mooney. Integrating language and vision to generate natural language descriptions of videos in the wild. In *ICCL*, 2014.

[Tran *et al.*, 2015] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[Vedantam *et al.*, 2015] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, 2015.

[Venugopalan *et al.*, 2015a] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond J. Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence - video to text. In *ICCV*, 2015.

[Venugopalan *et al.*, 2015b] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond J. Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. In *ACL*, 2015.

[Yao *et al.*, 2015] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher J. Pal, Hugo Larochelle, and Aaron C. Courville. Describing videos by exploiting temporal structure. In *ICCV*, 2015.

[Yu *et al.*, 2016] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *CVPR*, 2016.