Multi-Feature Fusion for Video Captioning

Yubo Jiang
Beijing Institute of Technology Zhuhai
No.6, Jinfeng Road, Tangjia Bay, Xiangzhou District, Zhuhai, Guangdong

ABSTRACT
Video captioning is the task that integrates natural language processing and computer together. For typical approaches, they are based on CNN-RNN, which use the pre-trained Convolutional Neural Network (CNN) to extract image feature and the Recurrent Neural Network (RNN) to generate captions word by word. However, most of the approaches only use the global video feature and loss the spatial and motion information. To address the aforementioned problem, a novel video captioning method based on multi-feature fusion is proposed. This method extracts the spatial features, motion features and video features of each frame, and all the features are fused to generate video captions. The fused features are input into long-term and short-term memory (LSTM) which is used as the natural language generating module. Multiple natural language modules are trained by different feature combinations, and then fused in later stages. First, one model is selected to obtain multiple possible outputs of the current input, and then the probabilities of the current output are calculated by other models. Then the probabilities of these outputs are weighted and the highest probabilities are taken as outputs. In this method, feature fusion methods include pre-fusion and post-fusion. Experiments on the standard test set MSVD show that the fusion of different types of feature methods can achieve higher evaluation scores; the same type of feature fusion evaluation results will not be higher than a single feature score; the use of features to fine-tune the pre-trained model is not effective. The METEOR score is 0.302, which is 1.34% higher than the current maximum value. It shows that this method can improve the accuracy of automatic video description.

General Terms
Pattern Recognition, Deep Learning

Keywords
Feature Fusion, Video Captioning, Deep Learning, LSTM

1. INTRODUCTION
With the development of mobile Internet industry and big data technology, computer vision has become a hot research field. In the past, It has become an impossible task to label and describe multimedia data entirely relying on manual work. This paper focuses on the automatic description of video content with natural language sentence. Video captioning has high application value and practical significance. It can be widely used in intelligent security, video retrieval, human-computer interaction, virtual reality and help the blind understand movie video. The task of describing video in natural language is very simple for normal people, but it is a difficult task for computer. It requires the proposed method to cross the semantic gap between low-level pixel features and high-level semantics. The existence of semantic gap leads to the video captioning is a challenging task.

Recently, researchers have realized video captioning as a sequence to sequence learning task, which is similar to the Machine Translation (MT). Inspired by the great success of Recurrent Neural Network (RNN) in MT, RNN have been introduced into video captioning and obtained many encouraging results [1], [2], [3]. Figure 1 shows the most of existing CNN-RNN based approaches which follow the encoder-decoder diagram. Firstly, every frame is represented as a fixed-length feature vector by a pre-trained Convolutional Neural Network (CNN), and all the vectors are pooled (input into a max-pooling or average-pooling layer) into one vector which is used as the whole video feature. Then the video feature is encoded into a fixed size vector and taken as the input to the RNN. Last, RNN is utilized as the decoder to generate the corresponding sentence word by word. In other words, each word is generated conditioned on the visual feature and previous words. However, this kind method has an obvious drawback, which regards the whole video as an image and this may destroy the motion and sequential information of the video.

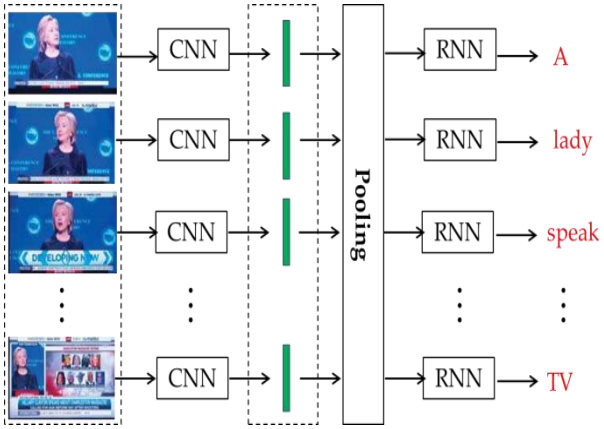

Figure 1. A generative "CNN-RNN" architecture for video captioning

To address the aforementioned problem, an end-to-end method for video captioning based on Long-Short Term Memory (LSTM) is proposed. Firstly, the features of video frame sequences were extracted. Different from other methods which only use the global video feature output via pooling manuscript, three kinds of features used in our method: 1) frame features output from the pre-trained CNN, 2) motion features such as Optical Flow (OF) and 3) video features such as Dense Trajectory (DT). In this paper, all these features are fused and then input into the LSTM to generate video captions.

In this paper, Caffe [4] is used to complete the CNN feature extraction the training of LSTM language generating module. In order to improve the speed of feature extraction and model training, the Graphics Processing Unit (GPU) is used for parallel computing to enhance the performance. The experimental results on Microsoft Video Description (MSVD) [5] dataset show that the proposed method achieves the state-of-the-art results among the most the popular methods, which shows the effectiveness on video captioning task.

The rest of this paper is organized as follows. Section 2 introduces some related works which are corresponding to our video captioning task. In Section 3, our method for video captioning is described in detail. The experimental results and analysis are shown in Section 4. Finally, we draw a conclusion for this in Section 5.

2. RELATED WORK

Recently, using natural language to describe the content of an image or a video has attracted much attention. Many challenges, such the Common Object in Context Explanation Contest (COCO), the Mass Film Description Challenge (MDC) and the Large Scale Movie Description Challenge (LSMDC), have attracted many researchers' attention and many corresponding papers have been published. In this section, some typical methods for image captioning and video captioning.

2.1 Image Captioning

Many researches have been done on image captioning and gained some progress. Methods for image captioning task can be roughly categorized three categories: 1) template-based methods; 2) retrieval-based methods; and 3) multi-modal neural network-based (MMNN-based) methods. Template-based methods are mainly composed two important parts: hard-coded language templates and visual concepts [6]. This kind method generates a fixed-length sentence for the input image. These methods retrieve similar images and the corresponding captions from the training dataset [7]. They project image features and sentence representations into a common space, which is used for ranking image captions or for image search. Recently, with the development of deep learning, CNN and RNN have gotten a lot exploits in CV and NLP fields [8], [9], [10]. Under such circumstances, MMNN-based methods become the most effective and popular methods. Their core idea is that regarding image captioning task as a machine translation (MT) problem. In other words, when an image is input into their models, it will be "translated" to a natural language sentence. They use CNN to extract image features and RNN to generate sentences. In machine translation, it is divided into encoding and decoding stages. In encoding stage, RNN reads the sentences of the source language and transforms them into a vector representation of a fixed length. In decoding stage, RNN takes the vector representation as the initial value of the hidden layer to generate sentences of the target language.

2.2 Video Captioning

There are two kinds of methods for generating natural language description by video:

One is the pipeline method which contains two stages [11]. In the first stage, semantic content (such as subject, verb, object, scene and so on) is located from video. In the second stage, semantic content is based on fixed templates (such as SVO (Subject, Verb, Object), SOV (Subject, Object, Verb), VSO (Verb, Subject, Object) and so on). Different language types have different syntax structures, so semantic content is based on different templates. These template-based image captioning methods have some limitations. They can only make a simple statement of the video, but cannot describe other rich information for the video.

The other is end-to-end method [12]. This kind method combines the video content with the original corpus of the corresponding natural language sentences as the input of the module. The method generates the natural language sentence for the input video in two steps: the first step is using the CNN to extract video features and the RNN to encode the corresponding sentences. Both the video features and the sentence representation are input into the multi-modal to train the whole model. In the second step, when a video input into the trained model, the video is represented into a fixed length feature vector and decoded into a sentence word by word.

In this paper, a novel video captioning method based on multi-feature fusion is proposed. Based on the mainstream pre-fusion and post-fusion methods, multi-feature fusion methods are studied. We extract different features for both the training and test videos. And all the features are fused by the pre- fusion module. In the same time, the corresponding sentences are represented into vectors. Then the sentence representation and the different video feature are used to train the different language module, respectively. When generating the describing sentence, the different features are fused into one vector via weighted summing, which we called the post-fusion. Finally, the effectiveness of this method is evaluated by experiments.

3. OUR METHOD

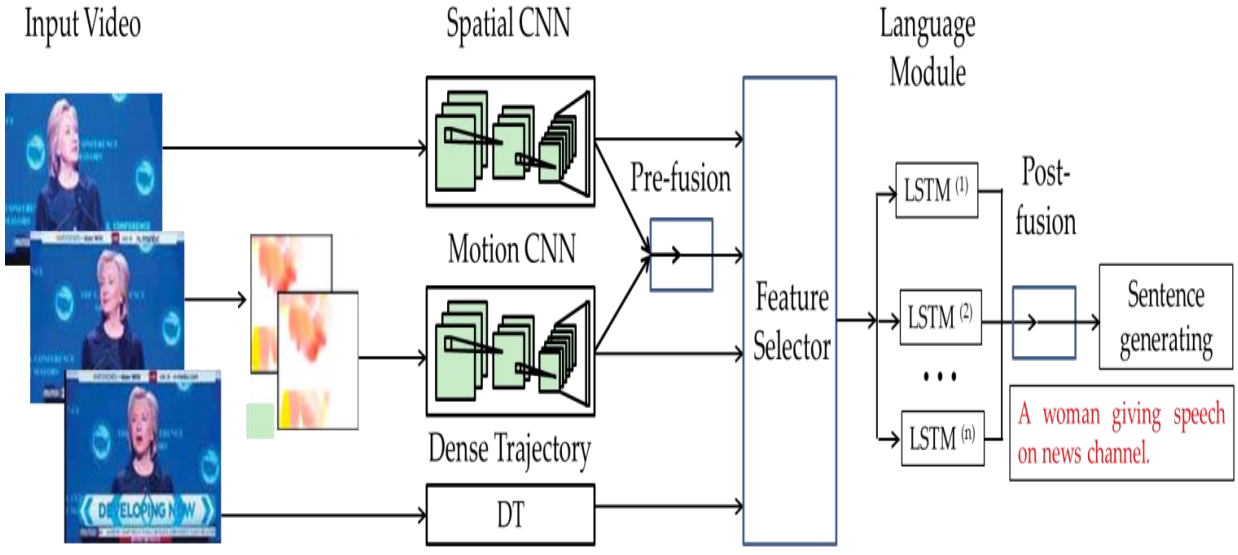

Figure 2. Overview of our scheme for video captioning

Figure 2 show the diagram of our model. The proposed method mainly contains five parts: 1) video features extracting, 2) multi-feature pre-fusion, 3) feature selecting, 4) language module and 5) feature post-fusion. Through the five parts, the describing sentences are generated. In this section, the proposed method is introduced in detail.

3.1 Caption Generating Model

This paper presents a model for video captioning as shown in Fig. 2. Input video frame sequences $V=(x_1,x_2,\mathrm{L}\ ,x_n)$, and finally get the output of natural language word sequences $S=(y_1,y_2,\mathrm{L}\ ,y_m)$. In normal cases, the number of input frames and the number of words sequence are variable. In this paper, when give an input video sequence V, the model estimated the output sentence S 's probability conditioned on the input V. This can be expressed as the following formula:

$$p(S|V)=p(y_1,y_2,\mathrm{L}\ ,y_m|x_1,x_2,\mathrm{L}\ ,x_n) \tag{1}$$

When given a video sample, we expect the probability conditioned on the given video to reach the maximum, i.e., in Eq. (1) gets the maximum value.

3.2 Feature Representation

This method firstly extracts three types of video features from the input video frame sequence, including spatial features (based on VGG16, AlexNet fc7 layer features), motion features (extracting video optical flow and inputting the optical flow maps into the pre-trained CNN to generate the feature map), dense trajectory features (DT features). And then both the features are fused at the pre-fusion layer. After the pre-fusion layer, a feature selector is followed.

The features used in this paper are extracted from the implementation of pre-trained models or open methods. Features used in this paper mainly contain three categories: 1) spatial feature, 2) motion feature and 3) trajectory feature.

Spatial Feature: In this paper, the pre-trained CNN is used to extract the spatial features of video frame sequence. In recent years, CNN has made a series of breakthroughs in image classification, object detection, image semantic segmentation and other fields. Features extracted by CNN can well express images or videos. So this paper chooses CNN (VGG16 and AlexNet) pre-trained on the ImageNet classification task dataset. We use the output from fc7 layer of the CNN as the frame feature. Then we calculate the mean value of the frame feature. Finally, a 4,096-dimensional feature vector is obtained to represent the whole video.

Motion feature: A characteristic of video is that it consists of many consecutive video frames. There are motion changes between frames. Therefore, it is necessary to analyze the motion features in video analysis. In this paper, the method described in reference [13] is used to extract the optical flow of adjacent frames. The extracted optical flow data are normalized to 0-255 and stored as a picture file. When the number of video frames is N, the number of optical flow images is N-1. The pre-training model [14] is used to extract the fc7 layer features of the optical flow image as motion features, and the mean value of the optical flow sequence features is also calculated. Finally, a 4,096-dimensional feature vector is obtained to represent the motion feature of the whole video.

Dense trajectory feature: Unlike image captioning task, video frames have temporal relationship, so it is necessary to extract temporal features of video when analyzing the video problem. In this paper, the DT feature is extracted by the method proposed in [15]. When extracting the DT feature, non-overlapping rectangular blocks are used to cover the region on the frames. Finally, the DT feature of each region obtained by stitching is used as the feature of the whole video.

3.3 Feature Fusion

The frames of different frames extracted by different models are characterized by $F_i=(F_{M_i}^1,F_{M_i}^2,\mathrm{L}\ ,F_{M_i}^n)$, where M_i represents the i-th module, they are used to verify the effectiveness of the proposed feature fusion strategy.

Pre-fusion. It is necessary to fuse features before model training, and the fused feature is the input of the model. This article has verified two kinds of pre-fusion methods:

1) Feature stitching. In the feature extraction stage, each module uses a vector F_i to represent the entire video, and i represents the i-th feature. F_{fusion} is obtained by stitching these features. This process can be expressed as the following formula:

$$F_{fusion}=(F_1,F_2,\mathrm{L}\ ,F_m), \tag{2}$$

where m represents the number of features.

2) Weighted summation. The length of the feature extracted from different modules is aligned, and the weight vector $W=(w_1,w_2,\mathrm{L}\ ,w_m)$ is used to weighted sum of the features. The formula is as follows:

$$\begin{aligned} F_{fusion} &= WF^T \\ &= w_1F_1+w_2F_2+\mathrm{L}\ +w_mF_m \end{aligned}, \tag{3}$$

where F_{fusion} is the final fused feature, m represents the number of features, and W satisfies the formula $\sum_{i=1}^{m} w_i=1$

Post-fusion. Following the multiple language modules, a post-fusion layer is used to re-fuse the video features. And two main methods are used to realize the post-fusion.

1) Fine-tuning mode. One feature is used as input to train the video natural language description model M_i. After training, another feature is used as input. The weights of the previously trained are used as initial values. The same network of video text generation models is used to fine-tune the model. Finally, M_{i+1}, which can be used to generate video natural language description model, is obtained.

2) Weighted summation. When estimating the output, one model M_i and the previous word are used as input to estimate the possible output of the next word, and the ten words with highest probability conditioned the current

input are obtained. The probabilities of these alternatives are $p_1 = p_{1k}(y')$ ($k = 1,2,\mathrm{L}\ ,10$). The probabilities of these alternatives are calculated by using other module M_i , and then re-calculated by weight vector. The probability of output word is calculated as the following formula:

$$p(y') = WP^T = w_1 p_1 + w_2 p_2 + \mathrm{L}\ + w_n p_n \ ,(4)$$

where $\sum_{i=1}^{n} w_i = 1$, n is the number of modules. We select the word with the highest probability as the output word..

3.4 Language Module

Inspired by the reference [16], the natural language generating module in this paper adopts two-layer LSTM network: one layer is used for encoding, and the input video features are transformed into vector representations; the other layer is used for decoding, and the video feature vectors are converted into word sequences. The LSTM model is chosen because it satisfies the following three basic conditions: 1) the model needs to be able to handle different lengths of video, and can generate different lengths of natural language description; 2) the model needs to be able to learn the temporal dependence of the frame before and after the video; in the training process, gradient descent method is used, and the error is caused; 3) Signals and gradients need to be transmitted back to the bottom for a long time.

1) In the training stage, the input data of the network includes the feature F_i directly extracted or the fused video feature vector F_{fusion} , and the corresponding sentences are projected to the same length as the feature vector through the embedding layer. Then the video feature vectors and the sentence representation are input into the language module to learn the relationship between video and sentence.

2) In the verification stage, the first layer encodes the video into feature vector F_{fusion} , the second layer decodes it, receives the hidden layer representation (h_t) and decodes it into words sequence. In the decoding phase, the model uses the maximum logarithmic likelihood function to estimate h_t and predict the next word. The model can be expressed by the following formula:

$$\theta^* = \arg\max_{\theta} \sum_{t=1}^{m} \log p\left(y_t \left| h_{n+t-1}, y_{t-1}; \theta \right.\right) \quad , \quad (5)$$

where θ denotes the model parameter set and $Y = (y_1, y_2, \mathrm{L}\ , y_m)$ represent the output words sequence.

In this paper, we use LSTM network as language module. Figure 3 shows the LSTM unit structure. Each LSTM unit contains a memory cell C whose output values are influenced by the current input x_t , previous hidden state h_{t-1}, and previous memory cell c_{t-1} . Every LSTM unit has four gates: input gate (i), input modulation gate ($\%$) and forget gate (f) to control the value of c_t , and output gate (o) to control the hidden state h_t . The forget gate allows the LSTM unit to forget the previous memory cell c_{t-1} , and the output gate determines how much memory is passed into the hidden layer (c_t). The gates and data update are defined as follows:

$$i_t = \sigma\left(W_{xi} x_t + W_{hi} h_{t-1} + b_i\right) \quad (6)$$

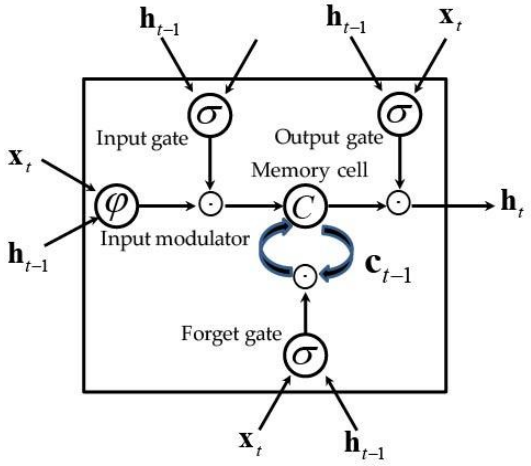

Figure 3. LSTM unit

$$f_t = \sigma\left(W_{xf} x_t + W_{hf} h_{t-1} + b_f\right) \quad (7)$$

$$o_t = \sigma\left(W_{xo} x_t + W_{ho} h_{t-1} + b_o\right) \quad (8)$$

$$\%_t = \sigma\left(W_{xc} x_t + W_{hc} h_{t-1} + b_c\right) \quad (9)$$

$$c_t = f_t \ \mathrm{e}\ c_{t-1} + i_t \ \mathrm{e}\ \% \quad (10)$$

$$h_t = o_t \ \mathrm{e}\ \phi(c_t) \quad (11)$$

3) In the test stage, when using the trained model to predict words sequence. When using the post-fusion strategy, we need to determine the specific value of the weight vector W . In this paper, we obtain W by fine-tuning the verification subset. In the fine-tuning process, we fix a weight w_i in turn, then assign $1 - w_i$ to $n-1$ other weight parameters. The minimum value of the weights is set to 0.01, and the range of each change is set to 0.01. The weight vector should satisfy the following formula:

$$\sum_{i=1}^{n} w_i = 1, \quad (12)$$

where n denotes the number of the language modules. When $n = 1$, it represent only one language module used.

4. EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Dataset

In this paper, MSVD dataset is used to verify the effectiveness of the proposed method. MSVD [6] dataset is a standard dataset for video captioning. The dataset contains 2089 videos in total. However, due to partial link failure, a total of 1,970 videos were saved. Every video have different language captions labeled by Amazon Mechanical Turk (AMT). Among them, a total of 80,827 descriptions are available in English, including 567,874 characters, 12,594 different words and symbols. Every video has the average length of 10.2 seconds with 41 corresponding sentences. We split the dataset as the [3] does. So the training set concludes 1,200 videos, the validation set contains 100 videos and the test set contains 160 videos.

4.2 Metric

We use four metrics to evaluate the proposed model. They are BLEU (BiLingual Evaluation Understudy) [17], ROUGE_L (Recall-Oriented Understudy for Gisting Evaluation, Longest Commonsubsequence) [18], and METEOR [19] and CIDEr (Consensus-based Image Description Evaluation) [20], respectively. The four metrics are calculated as a percentage. The higher value of the metrics, the higher quality of the sentences generated by the proposed model.

4.3 Experimental Steps

The basic procedures of this method are as follows:

1) Video preprocessing: extract the picture of each frame in the video by ffmpeg, and scale the picture a fixed size of 256 *256, and name the video frame from the beginning of the sequence.

2) Feature extraction: Based on video and frame sequences, the spatial features, motion features and video features described in Section 3.2 are extracted.

3) Preliminary feature fusion: Different spatial features, motion features and video features are selected to fuse through the feature fusion layer described in Section 3.3.

4) After fusing the previous features described in step 3, the new features are obtained, and the input data of the model are combined with the natural language description of the training data set to train multiple LSTM-based natural language description modules. The process is based on the deep learning framework Caffe, in which some of the hyper-parameters for training the natural language model are shown in Table 1.

Table 1. Part parameters of natural language description model based on LSTM

Hyper-parameter	Value	Meaning
test_interval	1,000	Test interval iterations
base_lr	0.01	Initial learning rate
lr_policy	step	Learning rate reduction strategy
gamma	0.5	The learning rate is reduced by multiplying the coefficient
stepsize	10,000	The number of iterations of reducing learning rate
momentum	0.9	momentum
clip_gradients	10	Gradient threshold

5) In the sentence generating step, several trained models are selected for post-fusion, and the weight vector W is obtained according to the weighting method in the post-fusion method described in Section 3.3.

6) Using the selected trained model and the weight vector W obtained in the previous step, a natural language description of the test set is generated.

7) Using the CocoCaption [21] tool, the similarity score is evaluated by combining the natural language description generated by this method with the manual video description given in the test set.

4.4 Results and Analysis

Table 2. Results of generated sentences on the MSVD

Feature	CIDEr	BLEU-4	ROUGE_L	METEOR
Spa (VGG)	0.415	0.327	0.676	0.290
Spa (Alex)	0.319	0.288	0.653	0.278
Mot	0.351	0.281	0.592	0.255
DT	0.417	0.318	0.636	0.291
[EC] Spa (VGG) + Spa (Alex)	0.414	0.315	0.632	0.287
[EC] Spa (VGG) + Mot	0.425	0.327	0.692	0.295
[EW] (Spa (VGG) + Spa (Alex))	0.407	0.321	0.659	0.282
[EC] (Spa(VGG) + Mot + DT)	0.429	0.331	0.697	0.299
Spa (VGG) + [LT] Alex	0.334	0.275	0.589	0.253

Spa (VGG) + [LT] Mot	0.374	0.302	0.661	0.271
[EC] (Spa (VGG) + Mot) + [LW] DT	0.436	0.337	0.698	0.301
Spa (VGG) + [LW] Mot + [LW] DT	0.451	0.352	0.703	0.302

Table 2 shows the results of generated sentences on the MSVD dataset. Spa represents spatial features, Mot represents motion features, DT represents video features, [EC] represents pre-fusion feature stitching, [EW] represents pre-fusion feature weighting sum, [LT] represents post-fusion fine-tuning, [LW] represents post-fusion weighting. The first to fourth rows of data are the evaluation scores for the experiment without feature fusion (only using a single feature); the fifth to eighth rows are the evaluation scores obtained only by pre-feature fusion; and the ninth to twelve rows are the evaluation scores obtained by the combination of pre-fusion and post-fusion. The results in Table 2 show that the results of the three feature fusion combinations proposed in this paper exceed the best value of 0.298 in reference [3], but the results of CIDEr and BLEU are lower than the best value of 0.5167 and 0.5167 in reference [12]. 0.4192. Through the data comparison and analysis, we can draw the following conclusions. 1) Fusion of different types of feature methods can improve the evaluation score, such as the 6-th, 8-th, 11-th and 12-th rows in Table 2 are higher than the single feature model before fusion evaluation score. 2) The results of the same type of feature fusion will not be higher than the score of single feature, for example, the results of rows 5-th and 7-th in Table 2 are lower than the model evaluation score generated by one of the individual features. 3) Using different features to fine-tune the existing model is less effective, such as rows 9 and 10 in Table 2, the score is much lower than the single feature model before fusion.

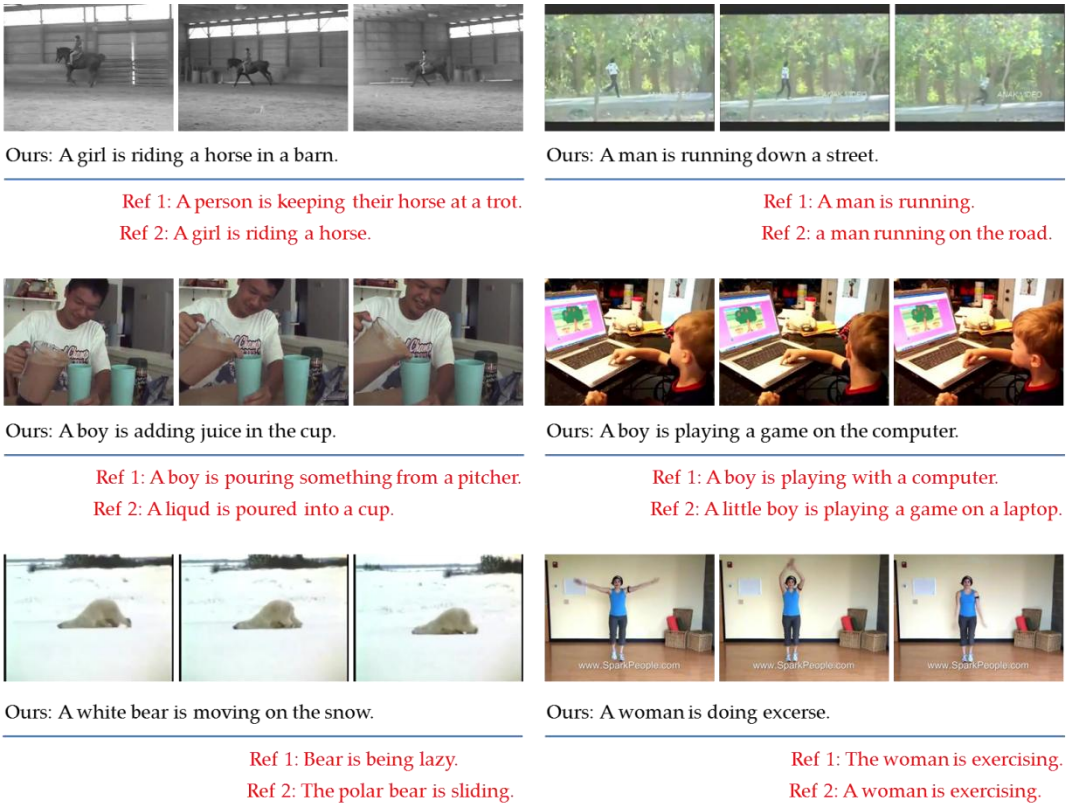

Figure 4. Example results of our method on the MSVD dataset

To provide a better understanding of the result, Figure 4 present some examples generated by our method. The sentences with black font are generated by the proposed method and the sentences with red font are reference in the dataset. It can be observed that most of the generated sentences can be basically describe the contents of the video. Through these examples, we can see that this method can generate the right captions effectively.

5. CONCLUSION

In this paper, a multi-feature video natural language description framework method is proposed. This method fuses features in two stages to improve the accuracy of natural language description. The features used in this paper include spatial features (such as VGG16, video frame sequence features extracted by AlexNet), motion features (CNN features of optical flow images), video features (DT features), and so on. Experiments on MSVD datasets show that the proposed method is feasible and effective, and the results show that multi-feature fusion can improve the accuracy of video natural language description, the same type of feature fusion cannot improve the accuracy, and the fusion between different types of features. There is a greater degree of accuracy. Based on the framework of feature fusion method proposed in this paper, appropriate selection of different types of features can further improve the accuracy of video natural language description.

6. REFERENCES

[1] J. Song, Z. Guo, L. Gao, W. Liu, D. Zhang and H. T. Shen, "Hierarchical LSTM with Adjusted Temporal Attention for Video Captioning", Proceedings of the International Joint Conferences on Artificial Intelligence, (2017) August.

[2] L. Baraldi, C. Grana and R. Cucchiara, "Hierarchical Boundary-Aware Neural Encoder for Video Captioning", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, (2017).

[3] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell and K. Saenko, "Sequence to Sequence – Video to Text", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, (2015).

[4] Y. Jia, Shelhamer , J. Donahue, et al., (2014). "Caffe: convolutional architecture for fast feature embedding", (2016).

[5] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. J. Mooney, T. Darrell and K. Saenko, "Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zeroshot recognition", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,, (2013).

[6] M. Everingham, L. Van Gool, C. K. I. Williams, and J. Winn, "The pascal visual object classes (voc) challenge", International Journal of Computer Vision (IJCV), vol. 88, no. 2, (2010), pp. 303–338.

[7] M. Hodosh, P. Young and J. Hockenmaier, "Framing image description as a ranking task: Data, models and evaluation metrics", Journal of Artificial Intelligence Research, vol. 47, (2013), pp. 853–899.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015).

[9] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", Proceedings of the Advances in Neural Information Processing Systems (NIPS), (2012).

[10] K. Cho, B. Van Merri¨enboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation", arXiv preprint arXiv:1406.1078, (2014).

[11] N. Krishnamoorthy, G. Malkarnenkar, R. J Mooney, and et al., "Generating Natural-Language Video Descriptions Using Text-mined Knowledge", Proceedings of the Twenty- Seventh AAAI Conference on Artificial Intelligence. Menlo Park, (2013).

[12] L. Yao, A. Torabi and K. Cho, "Describing videos by exploiting temporal structure", Proceedings of the 2015 IEEE International Conference on Computer Vision, Piscataway, (2015).

[13] G. Farneback, "Two-Frame Motion Estimation Based on Polynomial Expansion", Proceedings of the 13th Scandinavian Conference on Image Analysis, , Berlin (2003).

[14] G. Gkioxari and J. Malik, "Finding Action Tubes" Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Piscataway, (2015).

[15] H. Wang, A. Klaser, C. Schmid, "Action Recognition by Dense Trajectories", Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, Washington, (2011).

[16] O. Vinyals, A. Toshev and S. Bengio, "Show and Tell: A Neural Image Caption Generator", Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Piscataway, (2015).

[17] K. Papineni, "BLEU: A Method for Automatic Evaluation of Machine Translation", Wireless Networks, (2015).

[18] C. Lin and F. J. Och, "Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics" , Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, (2004).

[19] M. J. Denkowski and A. Lavie, "Meteor Universal: Language Specific Translation Evaluation For Any Target Language", Proceedings of the Ninth Workshop on Statistical Machine Translation, (2014).

[20] R. Vedantam, C. L. Zitnick and D. Parikh, "Cider: Consensus-Based Image Description Evaluation", in IEEE Conference on Computer Vision and Pattern Recognition, (2015).

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," Proceedings of the European Conference on Computer Vision (ECCV). Springer, (2014).