Automatic Video Captioning via Multi-channel Sequential Encoding

Chenyang Zhang and Yingli Tian(✉)

Department of Electrical Engineering,
The City College of New York, New York, NY 10031, USA
czhang10@citymail.cuny.edu, ytian@ccny.cuny.edu

Abstract. In this paper, we propose a novel two-stage video captioning framework composed of (1) a multi-channel video encoder and (2) a sentence-generating language decoder. Both of the encoder and decoder are based on recurrent neural networks with long-short-term-memory cells. Our system can take videos of arbitrary lengths as input. Compared with the previous sequence-to-sequence video captioning frameworks, the proposed model is able to handle multiple channels of video representations and jointly learn how to combine them. The proposed model is evaluated on two large-scale movie datasets (MPII Corpus and Montreal Video Description) and one YouTube dataset (Microsoft Video Description Corpus) and achieves the state-of-the-art performances. Furthermore, we extend the proposed model towards automatic American Sign Language recognition. To evaluate the performance of our model on this novel application, a new dataset for ASL video description is collected based on YouTube videos. Results on this dataset indicate that the proposed framework on ASL recognition is promising and will significantly benefit the independent communication between ASL users and others.

Keywords: Video captioning · Long-short-term-memory · Sequential encoding · American Sign Language

1 Introduction

Automatic visual content understanding and describing have become a fast-growing research area in computer vision for the recent decade. Effective understanding visual medias can significantly improve the performance of computer programs to automatically analyze and organize the online media. With the recent ground-breaking progress in large-scale visual recognition and deep neural networks, an explosive amount of techniques have been proposed in object recognition [1,2], scene understanding [3,4] and action recognition [5,6]. These findings successfully broaden the horizon of visual recognition research. Combining with the rapid progress of natural language processing, visual content describing has drawn more and more attention in the field of computer vision and machine

G. Hua and H. Jégou (Eds.): ECCV 2016 Workshops, Part II, LNCS 9914, pp. 146–161, 2016.
DOI: 10.1007/978-3-319-48881-3_11

learning. How to bridge the gap between visual content and natural human language has become the motivation of many research topics, such as image and video captioning.

Automatic image captioning deals with both images and textual data and generates natural sentences to summarize input image content. Generating descriptive sentences for images requires knowledge from multiple domains such as computer vision, natural language processing, and machine learning. Inspired by the recent renewed interests in deep learning techniques, there are many image captioning frameworks proposed [7–12]. The paradigm for generating captions for images takes two steps: (1) **Encoding stage:** the visual input (an image) is processed by a feature extraction layer (encoder). (2) **Decoding stage:** a language model is applied to decode the input feature encoding to a pre-defined vocabulary. The output sentence is generated based on the probabilistic distribution over the vocabulary using the language model. Recurrent neural network (RNN) has been proven to be an effective choice for the decoder because RNN is capable to address the temporal dynamics in output sentences.

Video captioning is a similar problem with image captioning and the encoder-decoder framework is also applicable for this problem. However, different from static images, videos contain much more semantic information related to temporal dynamics. Therefore, the video captioning framework should be able to model not only the static visual content inside each video frame, but also the temporal order of the frames. To address this problem, researchers have proposed several methods to adapt the encoder-decoder system to handle sequential inputs, such as mean-pooling over frames [13], temporal attention model [14], and directly employing sequence-to-sequence RNNs [15].

In this paper, we propose a novel framework for video captioning task. The main idea is illustrated in Fig. 1. To include more temporal motion-related information from the input video sequences, two channels (motion history images and raw video frames) are employed as video inputs. Our proposed framework integrates three different types of neural networks to perform automatic video captioning: (1) **3D-CNN:** instead of using object-detection-oriented feature extraction networks (such as VGG and AlexNet), we employ 3D convolutional neural networks (3D CNNs) to extract spatial-temporal features from video clips. (2) **RNN Encoder:** since the length of each video is arbitrary, the generated 3D CNN features are also of arbitrary lengths. A recurrent neural network (RNN) with long-short-term-memory (LSTM) cells is employed to map the sequential inputs to a fixed-dimensional encoding space. To jointly learn the encoding from two input channels, one LSTM encoder is assigned to each channel and the two encoders forms a parallel system. The fusion layer is a fully-connected layer which maps the LSTM internal states to the encoding space and the encoded vectors are concatenated. (3) **RNN language model:** the RNN language model defines a probability distribution of the next word in a sequence based on both the context and the current word. In our model, the context encoded in the form of LSTM internal state and initialized by the learned encoding vector.

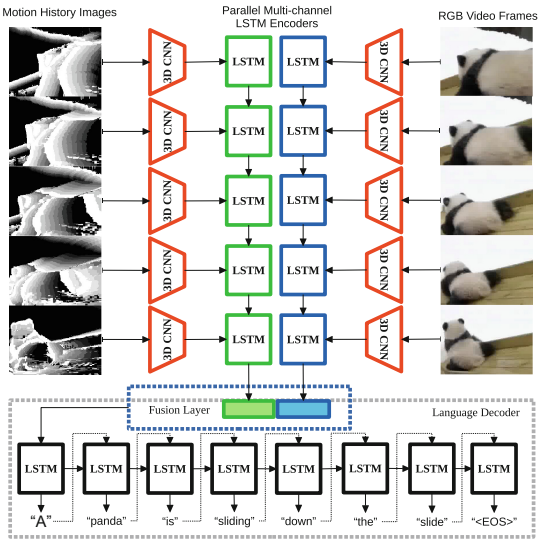

Fig. 1. Illustration of our proposed video captioning framework. Two channels of input frames are utilized: motion history images (MHIs) and RGB video frames. Firstly, raw features are extracted from each input channel frames using 3D convolutional neural networks. The feature extraction phase generates sequential features of arbitrary lengths. Secondly, the sequence of features is encoded using RNNs with LSTM cells for each channel. Then a fusion layer is employed to combine the encoded features from both LSTM encoders. Finally the fused features are fed into a LSTM-based language decoder to be decoded into a sequence of words. "<EOS>" represents the "*end of sentence*" token.

In addition, we also explore the potential utilization of the proposed video captioning framework in automatic video-based American Sign Language (ASL) translation. ASL is a visual gestural language which is used by many people who are deaf or hard-of-hearing. Automatically generating textual descriptions from ASL videos can significantly benefit the ASL-using population to communicate with non-ASL users. To the best of our knowledge, there has been no such effort to link ASL translation with video captioning before. We have collected a large-scale dataset from YouTube uploaded by ASL signers and gained annotations by aligning the video clips with subtitles. The proposed network is able to gain ASL-oriented knowledge from the dataset and to generate meaningful sentences from ASL videos.

The contributions of this work have three aspects:

- A sequential LSTM encoder framework is proposed to learn to embed video sequences addressing both spatial and temporal information.
- Our framework can handle multiple streams of input sequences and automatically learn how to combine.
- We are the first to explore video captioning in the area of ASL translation and provide a novel dataset in this area.

The rest of this paper is organized as the following. Section 2 reviews the related research work. Section 3 elaborates the architecture of the proposed framework. Then the datasets used and proposed by this paper are described in Sect. 4. Section 5 discusses the experiments. Finally the paper is concluded in Sect. 6.

2 Related Work

In this section, we briefly review the related research work in two aspects as below.

Video Captioning. Similar to image captioning, video captioning is also based on building connections between visual signals and textual data. Automatic video captioning is a recent branch of automatic video annotation, which starts with automatic video tagging. In [16], the authors explored to automatically assign conceptual tags to YouTube videos by learning from both visual and audio features. The authors of [17] treated the problem as an activity-recognition problem. They built hierarchical semantic trees to organize detected entities such as actors, actions, and objects. Zero-shot-learning-based language models were applied on the learned hierarchies to assign a short sentence to summary the detected potentials. Similarly, semantic triplets (*subject-verb-object*) were also used in [18] to organize detections of objects and activities for sentence inferencing. Quadruples were utilized in [19] to include more information from the context and scene for more accurate descriptions. Other efforts made to improve the performance of automatic tagging include video tag augmentation [20], video clustering [21], and video re-ranking [22]. Inspired by the successful utilization of LSTM-based RNNs in image captioning, there has been a lot of work using RNNs for video captioning. In [13], Venugopalan *et al.* proposed to apply average pooling over image features extracted from each video frame to obtain a video feature. Then the video feature was encoded to feed into a LSTM-based RNN language model for sentence decoding. To capture more temporal dynamics, attention models were applied in [23] to learn a weighting function over sampled key-frames. In [15], the authors explicitly modeled the sequential input (video) and sequential output (sentence) by exploiting a sequence-to-sequence LSTM architecture. Our work is most related to [15] because we also model the input encoding part with sequential input LSTMs. However, we separate the video encoding and sentence decoding parts to avoid feature entanglement. Additionally, applying such a separate model can enable us to conveniently combine multiple channels of input instead of raw-feature concatenation [23] or late score fusion [15].

American Sign Language Recognition. ASL is used by deaf people across U.S. and Canada. Some researchers have estimated that the population using ASL as a primary language was about 500,000 [24]. In automatic ASL recognition, early attempts have been made to explore the use of Hidden Markov Models (HMMs) in sequence modeling [25,26]. In [27,28], the authors proposed to track varies facial landmarks for ASL recognition. In recent years, since the

progress in commercial multi-modality sensors, researchers have been focusing on exploring the utilizations of multiple sensors. For example, in [29,30], the authors proposed to employ Kinect and Leap Motion sensors, respectively, for real-time hand-gesture-based ASL recognition. In this work, we propose to study ASL recognition from the perspective of data-driven video captioning. To the best of our knowledge, this is the first time ASL recognition is combined with video captioning.

3 Method

The framework of our proposed method is illustrated in Fig. 1. The whole framework is composed of four core modules: (1) 3D CNN-based feature extractor. (2) Sequential feature encoder. (3) Parallel fusion layer. (4) Sentence-generation language module. Both the feature encoder and the language module are based on RNNs with LSTM cells.

3.1 LSTM-based RNNs

Recurrent neural network (RNN) is a category of neural network containing an internal state. RNN is able to encode a dynamic temporal behavior due to its connections between units form directed cycles. The internal state of RNN can be treated as a state of memory, which contains information of both current input and the previous memory. Therefore, RNN has the capability to "remember" the history of both previous inputs and outputs. RNN is widely applied in prediction frameworks which is dependent on context, such as machine-translation [31]. A RNN cell can be formatted as:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t), \tag{1}$$

where h_t and x_t denote the hidden state and input encoding at time step t, respectively; W_h and W_x denote the parameters assigned to each state vector. $\sigma(\cdot)$ denotes the sigmoid function.

However, RNN often suffers from modeling long-term temporal dependencies [32]. A modification called *long-term-short-memory* (LSTM) is proposed for better long-term temporal dependency modeling with more sophisticated internal states and connections. A typical LSTM cell can be formatted as:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ \hat{C}_t &= tanh(W_c x_t + U_c h_{t-1} + b_c) \\ C_t &= f_t \odot C_{t-1} + i_t \odot \hat{C}_t \\ h_t &= o_t \odot tanh(C_t), \end{aligned} \tag{2}$$

where $\odot$ is element-wise product; $\sigma(\cdot)$ denotes the sigmoid nonlinearity-introduce function; x_t is the input encoding at each time step t to the LSTM cell; W_i, W_f, W_c, W_o, U_i, U_f, U_c, and U_o are weight matrices assigned to parameters of input gate, forget gate, cell state and output gate, respectively; b_i, b_f, b_c and b_o are bias vectors for corresponding gates and states; i_t, o_t, f_t, C_t and h_t denote the state values of input gate, output gate, forget gate, cell state and hidden state, respectively. $\hat{C}_t$ represents the candidate cell state before combining with the previous cell state (C_{t-1}) and the forget gate.

In our work, the LSTM cells are the building blocks of two types of RNNs: (1) feature encoding RNN and (2) sentence decoding RNN (language model). The illustrations of both RNNs are shown in Fig. 2. The two types of RNN cells are connected as illustrated in Fig. 1. The feature encoding RNN is responsible to encode the sequential inputs from video features; and the sentence decoding RNN is responsible to decode the output from encoding RNN to a sequence of words.

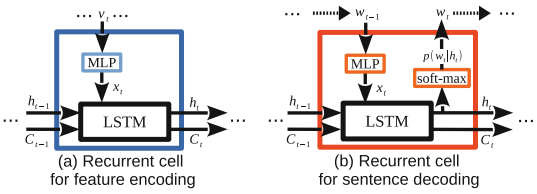

Fig. 2. Illustration of two types of recurrent cells for feature encoding and sentence decoding, respectively. Both cells contain an internal LSTM cell. At each time step, feature encoding recurrent cell takes an input video feature (v_t) and sentence decoding cell takes an input as the word-prediction (w_t) from the previous time step. Note that the MLPs in both cells act as look-up tables which map the input vector to the internal input vector (x_t).

3.2 Feature Encoder

Suppose the input video sequence $V = \{c_1, c_2, ..., c_T\}$ is composed of T short video clips. Without loss of generality, the length of each video clip $\|c_i\|$ could equal to 1 to represent individual frames. The video sequence can be encoded with a feature extractor ϕ (such as C3D [6] and VGG-net [33]), thus the video can be represented as: $\phi(V) = \{v_1, v_2, ..., v_T\}$, where $v_t = \phi(c_t)$ denotes a video feature vector for a video clip.

Therefore, the input video can be encoded into a sequence of feature vectors $\{v_t\}$. For the feature encoding RNN as illustrated in Fig. 2(a), one video feature v_t is fed into the RNN cell with a multiple-layer-perceptron (MLP). The MLP can represent any multi-layer neural network, and in our case the MLP indicates a fully-connected layer followed by a ReLU layer. Note that the MLP acts like a look-up table, mapping the input feature vector into a continuous RNN embedding space. At each time step t, the RNN cell takes input from both the previous

cell and the video sequence; it encodes the input vectors using an internal LSTM cell and output hidden state h_t and cell state C_t to the next cell. The behavior of the internal feature encoding LSTM cell ($LSTM_{FE}$) can be formatted as:

$$[h_t, C_t] = LSTM_{FE}(h_{t-1}, C_{t-1}, MLP(v_t)). \tag{3}$$

Parallel fusion layer. Our framework is designed to handle video encodings from multiple channels of the input video, such as RGB frames and motion history images (MHI) as shown in Fig. 1. Because different channel of video encoding contains different information, each channel should have its own feature encoding so that the intrinsic characteristics can be encoded. In our framework, to connect the output encoding vectors from feature encoding RNNs and the input of sentence decoding RNN, a parallel paradigm to conduct the mapping is employed:

$$ENC(V) = MLP(h_T) \oplus MLP(h_T^{'}), \tag{4}$$

where $ENC(V)$ denotes the final video encoding of the input video V and $\oplus$ denotes vector concatenation; h_T and $h_T^{'}$ denote the final state vector of two streams of RNN encoders. Note that the dimension of $ENC(V)$ matches with the dimension of RNN encoding space in the language model decoder.

3.3 Language Model

A general language model is usually designed to compute the probability of a sequence of words:

$$p(w_1, w_2, ..., w_K) = p(w_K|w_{K-1}, , , w_1) \cdot ... \cdot p(w_2|w_1) \cdot p(w_1), \tag{5}$$

where w_i is the i^{th} word in the output sentence.

In video captioning scenario, the language model is designed to compute the modified probability:

$$p(w_1, w_2, ..., w_K, Y) = p(w_K|w_{K-1}, , , w_1, Y) \cdot ... \cdot p(w_2|w_1, Y) \cdot p(w_1, Y), \tag{6}$$

where $Y = ENC(V)$ represents the encoded video.

In our framework, the language model is implemented with a RNN-based sentence decoder, as shown in Fig. 2(b). More specifically, the RNN decoding cell at each time step computes the probability by providing the previous output words and the video encoding as following:

$$\begin{aligned} p(w_t|w_{t-1}, , , w_1, Y) &= p(w_t|h_t) = SM(h_t) \\ [h_t, C_t] &= LSTM_{LM}(x_t, h_{t-1}, C_{t-1}) \\ x_t &= \begin{cases} Y, & \text{if } t = 1 \\ MLP(\mathbf{1}(w_{t-1})), & \text{otherwise,} \end{cases} \end{aligned} \tag{7}$$

where $SM(\cdot)$ represents a soft-max layer and $\mathbf{1}(\cdot)$ denotes the 1-hot-vector representation of the word index. Note that the MLP learns the mapping from word-index to the RNN internal space. The output word w_t is sampled according to the probability distribution computed by the soft-max layer.

3.4 Video Representation

In this section, the procedure of obtaining video representations, *i.e.* $\phi(V)$, is discussed.

Spatial-temporal feature extraction. In [13], the video representation is obtained from mean-pooling of static image feature vectors of each frame. However, videos are more than combinations of individual frame. Only including static image features can capture the visual appearance such as objects and scenes, but discard the information of temporal motions. For instance, in the example of Fig. 1, information about "panda" could be included in visual appearance features, but information about "sliding" will more likely be included in motion features. To capture sufficient spatial-temporal features, our framework employs two strategies: (1) two channels of raw video representations are included: motion history images and RGB video frames. MHI focus on temporal motions and RGB frames focus on spatial appearances. (2) For each short clips in each channel (16 frames), temporal-spatial features are computed via a 3D convolutional neural network (C3D [6]). The C3D networks are pre-trained on action recognition dataset so that they are capable to capture discriminative spatial-temporal features.

Context embedded video representation. Before feeding the extracted C3D features into video encoding RNNs, an additional pooling layer is added to provide more context information to the video representation:

$$\begin{aligned} \phi(V) &= \{v_0, v_1, ..., v_T\} \\ v_t &= \begin{cases} max\_pool(v_1, ..., v_T), & \text{if } t = 0 \\ C3D(c_t), & \text{otherwise,} \end{cases} \end{aligned} \tag{8}$$

where v_t represents the input for video encoding RNN at each time step t and c_t represents the corresponding video clip.

Therefore, at time step $t = 0$, the encoding RNN will be fed with the "context" vector, which is the max pooling vector over all C3D feature vectors. In this way, the video encoding RNN starts with the holistic knowledge about the whole video before taking the sequential inputs representing each video clip.

4 Datasets

4.1 Microsoft Video Description Corpus

The Microsoft video description (MSVD) corpus is a video snippet-based dataset, which focuses on describing simple interactive events, such as driving, cooking, *etc.* Each video snippet is collected from YouTube. There are about $1,658$ video clips in this corpus which are available by the time of our experiments. Each video snippet lasts from multiple seconds to several minutes. Human annotators were asked to describe the video snippet using one sentence from any language. Since each video snippet was assigned to multiple annotators, there are multiple

sentences for one video snippet. Here, our paper only focuses on English descriptions. Among the 1,658 video snippets, 300 are used as testing and the rest are for training.

4.2 Movie Description Datasets

In this paper, two movie description datasets are employed: Max Planck Institute for Informatics Movie Description Dataset (MPII) [34] and Montreal video annotation dataset (MVAD) [35]. Both of the datasets are collected from Hollywood movies. MPII dataset contains over 68,000 video snippets from 94 High-definition movies and MVAD dataset contains 49,000 video snippets from 92 movies. The text annotation from the MVAD dataset is from Descriptive Video Service (DVS), a linguistic description that allows visually impaired people to follow the movie. Besides DVS, the MPII dataset also employs movie scripts to enrich the text annotations. Both datasets are very challenging compared to the MSVD dataset in several aspects: (1) movie videos have more complex scenes and varied backgrounds. (2) The text annotations are sourced from a combined corpus, therefore the linguistic complexity is much higher than well-structured sentences as in the MSVD dataset. The MVAD and the MPII datasets belong to the recent Large Scale Movie Description Challenge (LSMDC). We report evaluation on the public testing set, where the MPII dataset has 3,535 testing video/sentence pairs and the MVAD has 6,518.

4.3 American Sign Language Video Description Corpus

To the best of our knowledge, previous automatic ASL recognition frameworks only focus on hand gesture or facial expression recognition. We further explore the utilization of video captioning framework for ASL recognition. Since there is no proper public dataset for this task, we propose a new dataset, **ASL-TEXT**, collected from YouTube. This proposed dataset is focused on describing videos of ASL signing, and it contains about 20,000 video-sentence pairs. The ASL-TEXT dataset is very challenging in two aspects: (1) the scenes are complex but irrelevant, and the only relevant information is from human facial expressions and body gestures. (2) The sentences are extracted from YouTube subtitles, some of which are generated by automatic voice recognition. Therefore the language complexity and variation are even higher than the previous mentioned movie description datasets.

The resource of ASL on YouTube comes in several categories, such as *ASL lessons, ASL songs,* and *ASL instructions provided by public institutes*. We manually search on YouTube with multiple textual queries such as "ASL", "American Sign Language", and "ASL Lessons", *etc.* The search results are further manually filtered using several criteria: (1) the search results should be correct ASL signing. (2) The subtitles associated with the video snippets should be available. (3) There should be only one frontal-view signer in the video. To further rule out unnecessary background noises, face detection is applied on each video

frame and the video frames are then centered and cropped according to the face detection results. Some examples of the dataset are shown in Fig. 3(d).

Following the convention in MPII and MVAD datasets, each video is segmented into several short snippets. Since each video in our dataset has caption (or subtitle) available, we segment the videos so that each video clip corresponds to one sentence in the caption text. As a result, the ASL-TEXT dataset contains $22,527$ video/sentence pairs and the average length of video clips is $5.4s$. The sizes of vocabularies in the three datasets are comparable but the ASL-TEXT dataset has less words. The ASL-TEXT dataset is more challenging because the averaged word frequency is much lower than in the other two datasets. This dataset will be released to public (Table 1).

Table 1. Comparative statistics of the propose ASL-TEXT dataset with the MSVD and MPII datasets.

	#-sentences	#-words	Vocab. size	Avg. length
MPII	68,375	679,157	21,700	3.9 s
MVAD	56,634	568,408	18,092	6.2 s
ASL-TEXT	22,527	178,637	11,193	5.4 s

5 Experimental Results

5.1 Experimental Setup

Metric. In this paper, we mainly evaluate the proposed framework using the METEOR evaluation metric [36]. Compared to other n-gram-based metrics such as BLEU [37], METEOR is more appropriate to evaluate sequential predictions. METEOR scores the predictions by aligning them to more than one reference sentences, which are based on exact, stem, synonym, and paraphrase matches between words and phrases. Therefore METEOR takes more linguistic and semantic information into consideration.

Loss function. In each iteration during the training process, a batch of images is fed into the neural networks, and the language decoder generates a sequence of probability distributions. A log-likelihood function is applied for each probability vector and corresponding ground-truth vector (1-hot-vector). The losses and gradients are then computed by maximizing the likelihood function. The losses and gradients are averaged and back-propagated to the preceding network modules for parameter updates.

Training and optimization. For computational efficiency, we assign the weights for the C3D networks with a pre-trained network and do not apply fine-tuning. The rest of the modules (LSTM feature encoder, fusion layer, and LSTM

language decoder) are trained end-to-end using stochastic gradient descent. The learning rates for all modules are set to 0.0001. Each iteration contains a batch of 16 samples. All RNN sizes are set to 1024. The drop-out rates for both encoder and decoder are set to 0.5. We implement the networks using Torch7 [38] and CuDNN. It takes about 1 to 3 days to converge on the training set using a GeForce TitanX core, depending on the sizes of datasets.

5.2 Video Description Results

MSVD dataset. The comparative METEOR scores of the proposed and other methods are shown in Table 2. The proposed method significantly outperforms the baseline factor graph model (FGM [19]) by 6.3 %. Comparing with *mean-pooling* methods [13], the improvements are 1.1 %–3.3 %, which demonstrate that including more temporal dynamic information is beneficial. Comparing with the current sequential modeling state-of-the-arts, temporal attention (TA) [14] and S2VT [15], our proposed method performs slightly better (30.2 % *vs.* 29.0–29.8 %). Some qualitative results are shown in Fig. 3(a).

MSVD dataset is more focused on describing static human-object interactions and scenes, such as "someone is doing something in somewhere". Comparing temporal-based methods (the proposed, TA [14] and S2VT [15]) and static-based methods (mean-pooling [13]), there are improvements but limited.

Table 2. METEOR scores on the MSVD dataset.

Method	METEOR (%)
FGM [19]	23.9
AlexNet [13]	26.9
VGG [13]	27.7
AlexNet-COCO [13]	29.1
GoogleNet [14]	28.7
GoogleNet + TA [14]	29.0
GoogleNet + 3D-CNN + TA [14]	29.6
AlexNet(Flow) + S2VT [15]	24.3
AlexNet + S2VT [15]	27.9
VGG + S2VT [15]	29.2
VGG + AlexNet(Flow) + S2VT [15]	29.8
Proposed	**30.2**

MPII and MVAD datasets. To further comparative evaluate our proposed method with the state-of-the-arts on more temporal-focused datasets, two movie-based datasets (MVAD and MPII) are employed for comparison. The proposed framework and other state-of-the-arts are compared in Table 3. Despite the scores

on each of the MPII and MVAD datasets, we also report the overall scores (weighed by the sizes of testing set). Our result (7.06) outperforms Visual-Labels (6.55) and VGG (6.31) by 0.51 and 0.75, respectively. It is beneficial to explicitly model the temporal dynamics of the input videos.

Compared to the previous state-of-the-art sequence-to-sequence model (S2VT [15]), our framework outperforms by 0.25. The experimental results demonstrate that our framework can avoid feature entanglement so that it can better model the temporal structures of videos.

Table 3. METEOR scores (%) on the Movie Description datasets, higher is better.

Method	MPII [34]	MVAD [35]	Overall
SMT [34]	5.6	–	–
Visual-Labels [39]	7.0	6.3	6.55
VGG [13]	6.7	6.1	6.31
Temporal Attention [14]	–	4.3	–
S2VT [15]	7.1	6.7	6.81
Proposed	7.0	7.1	**7.06**

5.3 ASL-TEXT

Since there is no other result available on our ASL-TEXT dataset, we evaluate the proposed framework on this new dataset comparing among different network configurations. There are two aspects to be investigated in this comparative evaluation. Firstly, since our fusion layer can assign different dimensions to each feature channel, the impact of assigning different portions to RGB and MHI will be discussed. Secondly, the impact of RNN sizes for both feature encoders and language decoders will be discussed. $20,527$ training samples and $2,000$ testing samples from ASL-TEXT are used and the METEOR scores of different configurations are shown in Table 4. In Table 4, $(RGB)\%$ denotes the parameter of how much percent of the encoding feature dimensions is assigned to RGB channel; (RNN_{ENC}, RNN_{DEC}) denotes the RNN sizes for encoder and decoder. There are two observations can be made from Table 4: (1) for each row, the METEOR score increases as the RNN sizes increases but after an optimal size setting, the performance starts to decrease. (2) Assigning different dimensions to different feature channels has little impact on the performance. Observation 1 shows that the ASL-TEXT dataset is more complex than other datasets because even moderate RNN sizes such as $(512, 512)$ is sufficient to over-fitting. Observation 2 demonstrates that our framework can automatically learn an optimal combination of multiple feature channels. Therefore there is no need to manually tune the weight of different feature channels.

Table 4. METEOR scores on the ASL-TEXT dataset of different configurations.

		(RNN_{ENC}, RNN_{DEC})				
		(128,128)	(256,128)	(256,256)	(512,256)	(512,512)
(RGB)%	10%	3.9	**4.7**	4.3	4.2	3.6
	30%	4.1	3.8	**4.7**	3.5	3.9
	50%	3.7	**4.7**	3.5	3.9	3.9
	90%	3.7	3.7	3.5	**4.5**	4.0

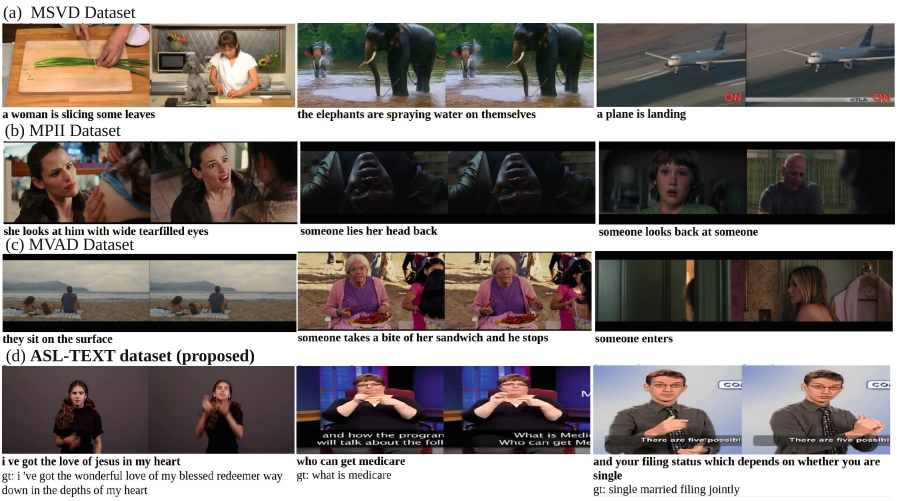

Fig. 3. Qualitative results of the proposed video captioning framework on four datasets: (a) MSVD, (b) MPII, (c) MVAD and (d) ASL-TEXT. The bold sentence under each pair of images is the predicted caption and for ASL-TEXT the ground-truth text is also attached.

Some qualitative results of the proposed framework have been shown in Fig. 3. For simple scenes and interactive actions in Fig. 3(a), our system can accurately generate descriptive sentences. For more complex scenarios as in movies (Fig. 3(b) and (c)), our system can predict well on the main actions (such as "sit", "eat" and "enter") but make errors in objects. For ASL recognition, it is promising to observe that the system has the potential to build relationships between key words (such as "love", "medicare", "WH-sign" and "single/married") and videos. The results demonstrate that exploring temporal structures and combining multiple feature channels are potentially beneficial for video captioning even in complex visual content and sentence structures.

6 Conclusion

In this paper, we have proposed a novel video captioning framework based on a two-stage encoder-decoder system. The encoding part is composed of a

multi-channel LSTM-based RNNs which can capture the temporal dynamics in video clips by allowing arbitrary-length input sequences. The decoding part is a LSTM-based language model which can decode the input video feature vector to a sequence of English words. A fusion layer is inserted between the encoder and decoder to automatically learn the optimized combination of multiple channels. To capture spatial-temporal information in the videos, we apply 3D convolutional neural networks pre-trained for action recognition (C3D) to extract features from both MHIs and raw RGB video frames. The whole network can be trained end-to-end using back-propagation. The proposed model is extensively evaluated on three public video description datasets comparing with the state-of-the-art methods and outperforms their performances. Furthermore, we collect an ASL recognition dataset and propose to apply video description framework in the area of automatic ASL recognition.

Acknowledgment. This work was supported in part by NSF grants EFRI-1137172, IIP-1343402, and IIS-1400802.

References

1. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: CVPR, pp. 248–255. IEEE (2009)
2. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229 (2013)
3. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. IEEE PAMI **35**(8), 1915–1929 (2013)
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR, pp. 580–587 (2014)
5. Le, Q.V., Zou, W.Y., Yeung, S.Y., Ng, A.Y.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: CVPR, pp. 3361–3368. IEEE (2011)
6. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV, pp. 4489–4497 (2015)
7. Yao, B.Z., Yang, X., Lin, L., Lee, M.W., Zhu, S.C.: I2t: Image parsing to text description. Proc. IEEE **98**(8), 1485–1508 (2010)
8. Chen, X., Fang, H., Lin, T.Y., Vedantam, R., Gupta, S., Dollar, P., Zitnick, C.L.: Microsoft coco captions: data collection and evaluation server. arXiv preprint arXiv:1504.00325 (2015)
9. Mao, J., Xu, W., Yang, Y., Wang, J., Yuille, A.L.: Explain images with multimodal recurrent neural networks. arXiv preprint arXiv:1410.1090 (2014)
10. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. arXiv preprint arXiv:1412.2306 (2014)
11. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. arXiv preprint arXiv:1411.4555 (2014)
12. Chen, X., Zitnick, C.L.: Learning a recurrent visual representation for image caption generation. arXiv preprint arXiv:1411.5654 (2014)
13. Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., Saenko, K.: Translating videos to natural language using deep recurrent neural networks. In: NAACL-HLT (2015)

14. Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., Courville, A.: Describing videos by exploiting temporal structure. In: ICCV, pp. 4507–4515 (2015)
15. Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., Saenko, K.: Sequence to sequence - video to text. In: ICCV (2015)
16. Aradhye, H., Toderici, G., Yagnik, J.: Video2text: learning to annotate video content. In: ICDM Workshop on Internet Multimedia Mining (2009)
17. Guadarrama, S., Krishnamoorthy, N., Malkarnenkar, G., Venugopalan, S., Mooney, R., Darrell, T., Saenko, K.: Youtube2text: recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In: ICCV, pp. 2712–2719. IEEE (2013)
18. Krishnamoorthy, N., Malkarnenkar, G., Mooney, R., Saenko, K., Guadarrama, S.: Generating natural-language video descriptions using text-mined knowledge. In: NAACL HLT 2013, p. 10 (2013)
19. Thomason, J., Venugopalan, S., Guadarrama, S., Saenko, K., Mooney, R.: Integrating language and vision to generate natural language descriptions of videos in the wild. In: COLING (2014)
20. Morsillo, N., Mann, G., Pal, C.: YouTube scale, large vocabulary video annotation. In: Schonfeld, D., Shan, C., Tao, D., Wang, L. (eds.) Video Search and Mining. SCI, vol. 287, pp. 357–386. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12900-1_14
21. Huang, H., Lu, Y., Zhang, F., Sun, S.: A multi-modal clustering method for web videos. In: Yuan, Y., Wu, X., Lu, Y. (eds.) ISCTCS 2012. CCIS, vol. 320, pp. 163–169. Springer, Heidelberg (2013). doi:10.1007/978-3-642-35795-4_21
22. Wei, S., Zhao, Y., Zhu, Z., Liu, N.: Multimodal fusion for video search reranking. IEEE Trans. Knowl. Data Eng. **22**(8), 1191–1199 (2010)
23. Xu, K., Ba, J., Kiros, R., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: neural image caption generation with visual attention. arXiv preprint arXiv:1502.03044 (2015)
24. Karchmer, M.A., Bachleda, B., Mitchell, R.E., Young, T.A.: How many people use asl in the united states? why estimates need updating. Sign Lang. Stud. **6**(3), 306–335 (2006)
25. Vogler, C., Metaxas, D.: Parallel hidden markov models for american sign language recognition. In: ICCV, vol. 1, pp. 116–122. IEEE (1999)
26. Starner, T., Pentland, A.: Real-time american sign language recognition from video using hidden markov models. In: Motion-Based Recognition, pp. 227–243. Springer (1997)
27. Metaxas, D.N., Liu, B., Yang, F., Yang, P., Michael, N., Neidle, C.: Recognition of nonmanual markers in american sign language (asl) using non-parametric adaptive 2d–3d face tracking. In: LREC, pp. 2414–2420. Citeseer (2012)
28. Liu, J., Liu, B., Zhang, S., Yang, F., Yang, P., Metaxas, D.N., Neidle, C.: Recognizing eyebrow and periodic head gestures using crfs for non-manual grammatical marker detection in asl. In: FGR, pp. 1–6. IEEE (2013)
29. Pugeault, N., Bowden, R.: Spelling it out: Real-time asl fingerspelling recognition. In: ICCV Workshops, pp. 1114–1119. IEEE (2011)
30. Fok, K.Y., Ganganath, N., Cheng, C.T., Tse, C.K.: A real-time asl recognition system using leap motion sensors. In: CyberC, pp. 411–414. IEEE (2015)
31. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
32. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)

33. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
34. Rohrbach, A., Rohrbach, M., Tandon, N., Schiele, B.: A dataset for movie description. In: CVPR (2015)
35. Torabi, A., Pal, C., Larochelle, H., Courville, A.: Using descriptive video services to create a large data source for video annotation research. arXiv preprint arXiv:1503.01070 (2015)
36. Denkowski, M., Lavie, A.: Meteor universal: language specific translation evaluation for any target language. In: Proceedings of the EACL 2014 Workshop on Statistical Machine Translation, vol. 6 (2014)
37. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics (2002)
38. Collobert, R., Kavukcuoglu, K., Farabet, C.: Torch7: a matlab-like environment for machine learning. In: BigLearn, NIPS Workshop. Number EPFL-CONF-192376 (2011)
39. Rohrbach, A., Rohrbach, M., Schiele, B.: The long-short story of movie description. In: Gall, J., Gehler, P., Leibe, B. (eds.) GCPR 2015. LNCS, vol. 9358, pp. 209–221. Springer, Heidelberg (2015). doi:10.1007/978-3-319-24947-6_17