

M³: Multimodal Memory Modelling for Video Captioning

Junbo Wang^{1,3} Wei Wang^{1,3,*} Yan Huang^{1,3} Liang Wang^{1,2,3} Tieniu Tan^{1,2,3}

¹Center for Research on Intelligent Perception and Computing (CRIPAC),
National Laboratory of Pattern Recognition (NLPR)

²Center for Excellence in Brain Science and Intelligence Technology (CEBSIT),
Institute of Automation, Chinese Academy of Sciences (CASIA)

³University of Chinese Academy of Sciences (UCAS)

{junbo.wang, wangwei, yhuang, wangliang, tnt}@nlpr.ia.ac.cn

Abstract

Video captioning which automatically translates video clips into natural language sentences is a very important task in computer vision. By virtue of recent deep learning technologies, video captioning has made great progress. However, learning an effective mapping from the visual sequence space to the language space is still a challenging problem due to the long-term multimodal dependency modelling and semantic misalignment. Inspired by the facts that memory modelling poses potential advantages to long-term sequential problems [35] and working memory is the key factor of visual attention [33], we propose a Multimodal Memory Model (M³) to describe videos, which builds a visual and textual shared memory to model the long-term visual-textual dependency and further guide visual attention on described visual targets to solve visual-textual alignments. Specifically, similar to [10], the proposed M³ attaches an external memory to store and retrieve both visual and textual contents by interacting with video and sentence with multiple read and write operations. To evaluate the proposed model, we perform experiments on two public datasets: MSVD and MSR-VTT. The experimental results demonstrate that our method outperforms most of the state-of-the-art methods in terms of BLEU and METEOR.

1. Introduction

Describing videos with natural sentences automatically also called video captioning is very important for bridging vision and language, which is also a very challenging problem in computer vision. It has plenty of practical applications, e.g., human-robot interaction, video indexing and describing videos for the visually impaired.

Video captioning involves in understanding both vision

*Corresponding Author: Wei Wang

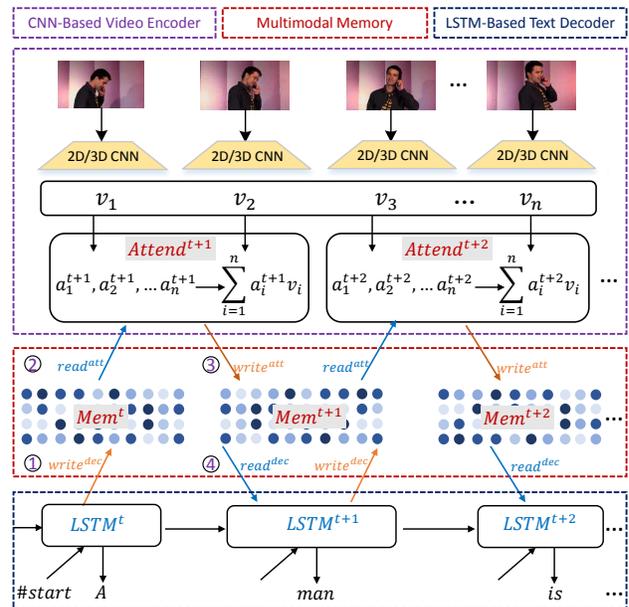


Figure 1. The overall framework of M³ for video captioning. It contains a CNN-based video encoder, a multimodal memory and a LSTM-based text decoder which are denoted by dashed box in different colors. The multimodal memory *Mem* stores and retrieves both visual and textual information by interacting with video and sentence with multiple read and write operations. The proposed M³ with explicit memory modelling can not only model the long-term visual-textual dependency, but also guide visual attention for effective video representation. (Best viewed in color)

and language, and then builds the mapping from visual contents to words. As we know, video as image sequence contains rich information about actor, object, action, scene and their interactions. It is very difficult for the existing methods to use a single visual representation [31] to capture all these information over a long period. Yao et al. [37] at-

tempt to dynamically select multiple visual representations based on temporal attention mechanism which is driven by the hidden representations from a Long Short-Term Memory (LSTM) text decoder. The LSTM text decoder, which integrates the information from both words and selected visual contents, models the sentence generation and guides visual selection. Recently, neural memory models have been proposed and successfully applied to question answering [35], which show greater advantages than LSTM to model long-term dependency in sequential problems. Furthermore, working memory is one of the key factors to guide eye movement in visual attention for efficient visual search, which has been computationally modelled in [14] and [33]. Explicitly modelling memory for video and sentence in visual captioning can not only model the long-term visual-textual dependency, but also guide visual attention to solve multimodal semantic misalignment. As we know, few memory models have been proposed for multimodal sequences.

In this paper, we propose a Multimodal Memory Model (M^3) to describe videos, which builds a visual and textual shared memory to guide visual attention on described targets and enhance the long-term visual-textual dependency modelling. Inspired by Neural Turing Machines [10], the proposed M^3 attaches an external memory to store and retrieve both visual and textual information by interacting with video and sentence with multiple read and write operations. Fig. 1 shows the overall framework of multimodal memory modelling for video captioning, which consists of three key components: convolutional neural networks (CNN) based video encoder, multimodal memory and LSTM-based text decoder. (1) CNN-based video encoder first extracts video frame/clip features using pretrained 2D/3D CNNs which are often used for image/video classification. The extracted features $\{v_i\}_{i=1}^n$ form the original video representation. Similar to [37], temporal soft-attention *Attend* is used to select visual information most related to each word. But very different from [37] using the hidden states from a LSTM decoder, we guide the soft-attention based on the content from a multimodal memory ($read^{att}$ in Fig. 1 denotes the content read from memory for attention). Then the selected visual information will be written into the memory ($write^{att}$ denotes the content written to memory from selective attention). (2) LSTM-based text decoder models the sentence generation with a LSTM-RNN architecture, which predicts the $\{t+1\}^{th}$ word conditioned on not only previous hidden representation $LSTM^t$ but also the content read from the multimodal memory ($read^{dec}$ denotes the content read from memory for decoder). Besides word prediction, the text decoder also writes the updated representation to the memory ($write^{dec}$ denotes the content written to memory from the decoder). (3) Multimodal memory contains a memory matrix Mem to interact with video and

sentence, e.g., write hidden representation from the LSTM decoder to memory $write^{dec}$, and read memory contents for the decoder $read^{dec}$. Each write operation will update the multimodal memory, e.g., from Mem^t to Mem^{t+1} . In Fig. 1, we illustrate the procedure of memory-video/sentence interactions: ① write hidden states to update memory, ② read the updated memory content to perform soft-attention, ③ write selected visual information to update memory again, ④ read the updated memory content for next word prediction. The main contributions of our work are summarized as follows:

- To our knowledge, we are the first to model multimodal data by selective reading/writing both visual contents and sentences with a shared memory structure, and apply it to video captioning.
- The proposed model performs better than most of the state-of-the-art methods on two public datasets: MSVD and MSR-VTT, which demonstrates its effectiveness.

2. Related Work

In this section, we briefly introduce some existing work that closely related to our proposed model.

Video Captioning Video captioning has been investigated for a long period due to its importance in bridging vision and language. Various methods have been proposed to solve this problem, which can be mainly categorized into two classes. The first class [12, 17, 28] detect the attributes of given videos and derive the sentence structure with predefined sentence templates. Then probabilistic graphical models are used to align the phases to the attributes. Similar to image captioning, these methods always generate grammatically correct sentences, but lose the novelty and flexibility of the sentence. The second class of methods inspired by Neural Machine Translation (NMT) [16, 6] map video sequence to sentence by virtue of deep neural networks, e.g., CNNs and RNNs. Venugopalan et al. [31] apply average pooling to extract the features of multiple video frames and use a two-layer LSTM network on these features to generate descriptions. In order to enhance video representation, Ballas et al. [1] exploit the intermediate visual representation extracted from pre-trained image classification models, and Pan et al. [20] propose a hierarchical recurrent neural encoder to explore the temporal transitions with different granularities. In order to generate more sentences for each video, Yu et al. [38] exploit a hierarchical recurrent neural network decoder which contains a sentence generator and a paragraph generator. To emphasize the mapping from video to sentence, Yao et al. [37] propose a temporal attention model to align the most relevant visual segments to the generated captions, and Pan et al. [21] propose a long short-term memory with a visual-semantic embedding

model. Recently, the second class of deep learning based methods have made much progress in video captioning. We augment the existing deep learning based models with an external memory to guide visual attention and enhance the long-term visual-textual dependency modelling in this paper.

Memory Modelling To extend the memory ability of traditional neural networks, Graves et al. [10] propose a Neural Turing Machine (NTM) which holds an external memory to interact with the internal state of neural networks by attention mechanism. NTM has shown the potential of storage and access of information over long time periods which has always been problematic for RNNs, e.g., copying, sorting and associative recall. Besides memory matrix in NTM, memory is also modelled as continuous and differentiable doubly-linked lists and stacks [15], queues and dequeues [11]. Different from exploring various forms of dynamic storages, Weston et al. [34] model large long-term static memory. The internal information stored in the static memory is not modified by external controllers, which is specially used for reading comprehension. These memory networks have been successfully applied to the tasks which need dynamic reasoning, e.g., textual question answering [3] and visual question answering [35]. As we know, few memory models have been proposed for video captioning. In this paper, we will propose an external multimodal memory to interact with video and sentence simultaneously.

3. Our Model

In this section, we will first introduce three key components of our model including: 1) convolutional neural networks (CNN) based video encoder, 2) Long Short-Term Memory (LSTM) based text decoder, and 3) multimodal memory. Then we will explain the procedure of model training and inference in details.

3.1. CNN-Based Video Encoder

Convolutional neural networks (CNNs) have achieved great success in many computer vision tasks recently, e.g., image classification [18] and object detection [9]. Due to the power of representation learning, CNNs pre-trained by these tasks can be directly transferred to other computer vision tasks as generic feature extractors. To make better video representations, we consider using pre-trained 2D CNNs to extract appearance features of videos, and pre-trained 3D CNNs to obtain motion features of videos since the temporal dynamics is very important for video understanding. In particular for an input video, we first sample it with fixed number of frames/clips n , and then exploit the pre-trained 2D CNNs/3D CNNs to extract features of each frame/clip. We denote the obtained video representation as $V = \{v_1, v_2, v_3, \dots, v_n\}$, where n is the number of sampled frames/clips. In the following, we define the proposed

temporal attention model. Given the visual representations of the video, and the content r read from the multimodal memory. By virtue of a single layer neural network followed by a softmax function, the attention weights over all locations of the input video can be formulated as follows:

$$\alpha_i^t = \text{softmax}(w^T \tanh(W_r r_t^{vr} + U_\alpha v_i + b_\alpha)) \quad (1)$$

where W_r, U_α, b_α , and w are the parameters to be learned. Different from [37], here we incorporate the content read from multimodal memory instead of the previous hidden state from LSTM network. We argue that the hidden state from LSTM network can not fully represent all the information of previous words, while our multimodal memory can well keep them. Based on the attention weights, the final representation of input video can be gained by:

$$V_t = \sum_{i=1}^n \alpha_i^t v_i \quad (2)$$

To simplify the following description, the above procedure can be abbreviated as follows:

$$V_t = \beta(V, r) \quad (3)$$

3.2. LSTM-Based Text Decoder

Different from the commonly used unimodal LSTM [39], we incorporate the fused multimodal information r_t as another input, which is read from our multimodal memory during caption generation as demonstrated in next section. For given sentences, we use one-hot vector encoding to represent each word. By denoting the input word sequence as $\{y_t | t = 0, 1, 2, \dots, T\}$, and the corresponding embedding vector of word y_t as E_t , the hidden activations h_t at time t can be computed as follows.

$$i_t = \sigma(W_i E_{t-1} + U_i h_{t-1} + M_i r_t + b_i) \quad (4)$$

$$f_t = \sigma(W_f E_{t-1} + U_f h_{t-1} + M_f r_t + b_f) \quad (5)$$

$$o_t = \sigma(W_o E_{t-1} + U_o h_{t-1} + M_o r_t + b_o) \quad (6)$$

$$\tilde{c}_t = \phi(W_c E_{t-1} + U_c h_{t-1} + M_c r_t + b_c) \quad (7)$$

$$c_t = i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \quad (8)$$

$$h_t = o_t \odot \phi(c_t) \quad (9)$$

where the default operation between matrices is matrix multiplication, \odot denotes an element-wise multiplication, W , U , and M denote the shared weight matrices to be learned, and b denotes the bias term. \tilde{c}_t is the input to the memory cell c_t , which is gated by the input gate i_t . σ denotes the element-wise logistic sigmoid function, and ϕ denotes hyperbolic tangent function \tanh .

For clear illustration, the process of language modelling mentioned above can be abbreviated as follows.

$$h_t = \psi(h_{t-1}, c_{t-1}, y_{t-1}, r_t) \quad (10)$$

3.3. Multimodal Memory

Our multimodal memory at time t is a $N \times M$ matrix M_t , where N denotes the number of memory locations and M denotes the vector length of each location. The memory interacts with the LSTM-based language model and CNN-based visual model via selective read and write operations. Since there exists bimodal information, i.e., video and language, we employ two independent read/write operations to guide the information interaction.

3.3.1 Memory Interaction

The interaction of visual information and textual elements is performed in the following order.

Writing hidden representations to update memory

Before predicting the next word during the process of caption generation, our LSTM-based language model will write previous hidden representations into the multimodal memory, to summarize the previous textual information. We denote the current textual weighting vector, textual erase vector and textual add vector as w_t^{sw} , e_t^{sw} and a_t^{sw} , respectively, all of which are emitted by the LSTM-based language model. The elements of textual erase vector e_t^{sw} lie in the range of (0,1). The lengths of textual erase vector e_t^{sw} and textual add vector a_t^{sw} are both M . Since both the textual erase vector and textual add vector have M independent elements, the elements in every memory location can be erased or added in a fine-grained way. Then the textual information can be written into the memory as follows.

$$M_t(i) = M_{t-1}(i) [1 - w_t^{sw}(i) e_t^{sw}] + w_t^{sw}(i) a_t^{sw} \quad (11)$$

where $i \in [1, N]$ denotes i -th memory location.

Reading the updated memory for temporal attention

After writing textual information into the memory, the updated memory content is read out to guide a visual attention model to select prediction-related visual information. Assuming that the current visual weighting vector over the N locations at time t is w_t^{vr} , which needs to be normalized as follows.

$$\sum_{i=1}^N w_t^{vr}(i) = 1, 0 \leq w_t^{vr}(i) \leq 1, \forall i \in [1, N] \quad (12)$$

Then the visual read vector r_t^{vr} returned by the visual attention model is computed as a linear weighting of the row-vectors $M_t(i)$:

$$r_t^{vr} = \sum_{i=1}^N w_t^{vr}(i) M_t(i) \quad (13)$$

Temporal attention selection for video representation

After reading the updated memory content, we apply the proposed temporal attention model to select most relevant

video representations by increasing corresponding weights, which is very effective when there exist explicit visual-semantic mappings.

$$c_t = \beta(V, r_t^{vr}) \quad (14)$$

Writing selected visual information to update memory

After selecting visual information via the attention model above, the information will be written into the memory for updating. Similar to the operation of writing hidden representations into the memory, the current visual weighting vector w_t^{vw} , visual erase vector e_t^{vw} and visual add vector a_t^{vw} are all emitted by the visual attention model. The elements of visual erase vector e_t^{vw} lie in the range of (0,1). The lengths of visual erase vector e_t^{vw} and visual add vector a_t^{vw} are both M . Then the visual information can be written into the memory as follows.

$$M_t(i) = M_t(i) [1 - w_t^{vw}(i) e_t^{vw}] + w_t^{vw}(i) a_t^{vw} \quad (15)$$

Reading the updated memory for LSTM-based language model

When finishing the above writing operation, the updated memory is read out for language modelling. Similarly, assuming that the textual weighting vector over the N locations at the current time is w_t^{sr} , which also has to be normalized as follows.

$$\sum_{i=1}^N w_t^{sr}(i) = 1, 0 \leq w_t^{sr}(i) \leq 1, \forall i \in [1, N] \quad (16)$$

Then the textual read vector r_t^{sr} returned by the LSTM-based language model is computed as a linear weighting of the row-vectors $M_t(i)$:

$$r_t^{sr} = \sum_{i=1}^N w_t^{sr}(i) M_t(i) \quad (17)$$

Computing of RNN-based language model After getting the reading information from the updated memory, we can compute the current hidden state of LSTM-based language model by calling the following function.

$$h_t = \psi(h_{t-1}, c_{t-1}, y_{t-1}, r_t^{sr}) \quad (18)$$

3.3.2 Memory Addressing Mechanisms

As stated in [25, 10], the objective function is hard to converge when using a location-based addressing strategy. Therefore, we use a content-based addressing strategy to update the above read/write weighting vector. During the process of content-based addressing, each read/write head (e.g., the LSTM-based text decoder) first produces a key vector k_t and a sharpening factor β_t . The key vector k_t is mainly used for comparing with each memory vector $M_t(i)$ by a similarity measure function K , and the sharpening factor β_t is employed for regulating the precision of the focus. Then all of them can be computed as follows.

$$K(x, y) = \frac{x \cdot y}{\|x\| \cdot \|y\| + \epsilon} \quad (19)$$

$$d_t(i) = \beta_t K(k_t, M_t(i)) \quad (20)$$

$$w_t(i) = \text{softmax}(d_t(i)) \quad (21)$$

3.4. Training and Inference

Assuming that there are totally L training video-description pairs (x^i, y^i) in the entire training dataset, where the description y^i has a length of t_i . The overall objective function used in our model is the averaged log-likelihood over the whole training dataset plus a regularization term.

$$L(\theta) = \frac{1}{L} \sum_{i=1}^L \sum_{j=1}^{t_i} \log \rho(y_j^i | y_{1:j-1}^i, x^i, \theta) + \lambda \|\theta\|_2^2 \quad (22)$$

where y_j^i is a one-hot vector used to denote the input word, θ is all parameters to be optimized in the model, and λ denotes the regularization coefficient. As all components in our model including multimodal memory components are differential, we can use Stochastic Gradient Descent (SGD) to learn the parameters.

Similar to most LSTM language models, we use a softmax layer to model the next word’s probability distribution over the whole vocabulary.

$$z_t = \tanh(W_v V_t + W_h h_t + W_e y_{t-1} + b_h) \quad (23)$$

$$\rho_t = \text{softmax}(U_\rho z_t + b_\rho) \quad (24)$$

where $W_v, W_h, W_e, b_h, U_\rho$, and b_ρ are the parameters to be estimated. Based on the probability distribution ρ_t , we can recursively sample y_t until obtaining the end of symbol in the vocabulary.

4. Experiments

To validate the effectiveness of the proposed model, we perform extensive experiments on two public video captioning datasets. The one is Microsoft Video Description Dataset (MSVD) [4] which has been used by most of the state-of-the-art methods. The other is recently released Microsoft Research-Video to Text (MSR-VTT) [36] which is the largest dataset in terms of number of sentence and vocabulary.

4.1. Datasets

Microsoft Video Description Dataset Microsoft Video Description Dataset (MSVD) [4] consists of 1970 videos which range from 10 seconds to 25 seconds. Each video has multi-lingual descriptions which are labelled by the Amazon’s Mechanical Turk workers. For each video, the descriptions depict a single activity scene with about

40 sentences. So there are about 80,000 video-description pairs. Following the standard split [37, 21], we divide the original dataset into a training set of 1200 videos, a validation set of 100 videos, and a test set of 670 videos, respectively.

Microsoft Research-Video to Text Dataset Microsoft Research-Video to Text Dataset (MSR-VTT) is the recently released largest dataset in terms of number of sentence and vocabulary, which consists of 10,000 video clips and 200,000 sentences. Each video clip is labelled with about 20 sentences. Similar to MSVD, the sentences are annotated by Amazon’s Mechanical Turk workers. With the split in [36], we divide the original dataset into a training set of 6513 videos, a validation set of 497 videos and a testing set of 2990 videos, respectively.

4.2. Data Preprocessing

Video Preprocessing Instead of extracting features for each video frame, we uniformly sample K frames from original video for feature extraction. When the video length is less than K , we pad zero frames at the end of original frames. Empirically, we set K to 28 for 98 frames per video in MSVD, and set K to 40 for 149 frames per video in MSR-VTT. For the extensive comparisons, we extract features from both pretrained 2D CNN networks, e.g., GoogleNet [26], VGG-19 [23], Inception-V3 [27], ResNet-50 [13], and 3D CNN networks, e.g., C3D [29]. Specifically, we extract the features of the pool5/7x7_s1 layer in GoogleNet, the fc7 layer in VGG-19, the pool3 layer in Inception-V3, the pool5 layer in ResNet-50 and the fc6 layer in C3D.

Description Preprocessing The descriptions in MSVD and MSR-VTT are all converted into lower case. To reduce unrelated symbols, we tokenize all sentences by NLTK toolbox¹ and remove punctuations. The vocabulary in MSVD is about 13,000 while the vocabulary in MSR-VTT is about 29,000. For convenience, we set the vocabulary size to 20,000 for both datasets. So the rare words in MSR-VTT are eliminated to further reduce the vocabulary.

4.3. Evaluation Metrics

In this paper, we adopt two standard evaluation metrics: BLEU [22] and METEOR [7], which are widely used in machine translation and image/video captioning. The BLEU metric measures the n-grams precision between generated sentence and original description, which correlates highly with human evaluation results. The METEOR metric measures the word correspondences between generated sentences and reference sentences by producing an alignment [5]. METEOR is often used as a supplement to BLEU. To guarantee a fair comparison with previous methods, we utilize the Microsoft COCO Caption Evaluation tool [5] to gain all experimental results.

¹<http://www.nltk.org/index.html>

4.4. Experimental Settings

During model training, we add a start tag and an end tag to the sentence in order to deal with variable-length sentences. We also add masks to both sentences and visual features for the convenience of batch training. Similar to [37], the sentences with length larger than 30 in MSVD and the sentences with length larger than 50 in MSR-VTT are removed. For the unseen words in the vocabulary, we set them to unknown flags. Several other parameters, e.g., word embedding dimension (468), beam size (5) and the size of multimodal memory matrix (128,512), are set using the validation set. To reduce the overfitting during training, we apply dropout [24] with a rate of 0.5 on the output of fully connected layers and the output of LSTMs but not on the recurrent transitions. To further prevent gradient explosion, we clip the gradients to [-10,10]. The optimization algorithm is ADADELTA [40] which we find fast in convergence.

4.5. Quantitative Analysis

Method	B@1	B@2	B@3	B@4	METEOR
FGM	-	-	-	13.68	23.90
LSTM-YT	-	-	-	33.29	29.07
SA	-	-	-	40.28	29.00
S2VT	-	-	-	-	29.2
LSTM-E	74.9	60.9	50.6	40.2	29.5
p-RNN	77.3	64.5	54.6	44.3	31.1
HRNE	79.2	66.3	55.1	43.8	33.1
BGRCN	-	-	-	48.42	31.70
MAA	79.40	67.10	56.80	46.10	31.80
RMA	-	-	-	45.7	31.9
M ³ -c3d	77.30	68.20	56.30	45.50	29.91
M ³ -vgg19	77.70	67.50	58.90	49.60	30.09
M ³ -google	79.05	68.74	60.00	51.17	31.47
M ³ -res	80.80	69.90	60.40	49.32	31.10
M ³ -inv3	81.56	71.39	62.34	52.02	32.18

Table 1. The performance comparison with the other ten state-of-the-art methods using single visual feature on MSVD. The results of the proposed M³ with five single features are shown at the bottom of the table. We compare the best single feature results of the other ten methods at the top of the table.

Method	B@1	B@2	B@3	B@4	METEOR
SA-G-3C	-	-	-	41.92	29.60
S2VT-rgb-flow	-	-	-	-	29.8
LSTM-E-VC	78.8	66.0	55.4	45.3	31.0
p-RNN-VC	81.5	70.4	60.4	49.9	32.6
HBA	-	-	-	42.5	32.4
M ³ -VC	81.90	71.26	62.08	51.78	32.49
M ³ -IC	82.45	72.43	62.78	52.82	33.31

Table 2. The performance comparison with the other five state-of-the-art methods using multiple visual feature fusion on MSVD. Here V, C, I and G denote VGG-19 [23], C3D [29], Inception-V3 [27] and GoogleNet [26], respectively.

Method	B@1	B@2	B@3	B@4	METEOR
SA-V	67.82	55.41	42.90	34.73	23.11
SA-C	68.90	57.50	47.00	37.40	24.80
SA-VC	72.20	58.90	46.80	35.90	24.90
M ³ -V	70.20	56.60	44.80	35.00	24.60
M ³ -C	77.20	61.30	47.20	35.10	25.70
M ³ -VC	73.60	59.30	48.26	38.13	26.58

Table 3. The performance comparison with SA [37] using different visual features on MSR-VTT. Here V and C denote VGG-19 [23] and C3D [29], respectively.

4.5.1 Experimental Results on MSVD

For comprehensive experiments, we evaluate and compare with the state-of-the-art methods using single visual feature and multiple visual feature fusion, respectively. Before the comparisons to these methods, we refer to ten state-of-the-art approaches([28], [31], [37], [30], [21], [38], [20], [1], [8], [19]) as these abbreviations (FGM, LSTM-YT, SA, S2VT, LSTM-E, p-RNN, HRNE, BGRCN, MAA, RMA).

When using single visual feature, we evaluate and compare our model with the above ten state-of-the-art approaches. The experimental results in terms of BLEU (n-gram) and METEOR are shown in Table 1. Here we give the best single feature results of the compared ten methods, and show the results of the proposed M³ together with five single features, e.g., VGG-19 [23], C3D [29], Inception-V3 [27], ResNet-50 [13] and GoogleNet [26]. Among these compared methods, SA [37] is the most similar method to ours, which also has an attention-driven video encoder and LSTM-based text decoder but no external memory. When both models use the same GoogleNet feature, our M³-google can make a great improvement over SA by $\frac{51.17-40.3}{40.3} = 26.9\%$ in the BLEU@4 score and by $\frac{31.47-29.0}{29.0} = 8.5\%$ in the METEOR score, respectively. It can be concluded that the better performance of our model benefits from multimodal memory modelling. In addition, our five M³ models outperform all the other methods except HRNE [20] in terms of METEOR. It is because HRNE [20] specially focuses on building a hierarchical video encoder for captioning. To be noted, both MAA [8] and RMA [19] apply a different memory modelling for video captioning, but our model apparently performs much better than them by a large margin in many evaluation metrics, which proves the superiority of our model in the video captioning. To further compare the results of the five M³ models using different visual features, we can see that M³-inv3 achieves the best performance, following by M³-res, M³-google and M³-vgg19. The performance rank is very similar to that of these methods' image classification accuracy on ImageNet [18], which proves that visual feature is very important for video captioning. Actually, the same conclusion has been drawn in image captioning where GoogleNet features ob-

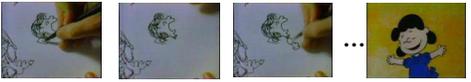
	Generated Sentence: SA: someone is playing M^3 : a man is drawing on a piece of paper	Reference Sentence: 1. a person is drawing a picture 2. a person is drawing a cartoon 3. the man is drawing a cartoon
	Generated Sentence: SA: a man is playing a guitar M^3 : a man is playing with a dog	Reference Sentence: 1. a man is petting two dogs 2. a man pets some dogs 3. a man is play with pets
	Generated Sentence: SA: men are playing soccer ball M^3 : people are playing basketball	Reference Sentence: 1. a basketball game is in play 2. two teams playing basket ball 3. people are playing basketball
	Generated Sentence: SA: a woman is mixing a bowl M^3 : a woman is mixing ingredients in a bowl	Reference Sentence: 1. a woman puts ingredients into a bowl 2. a woman is mixing flour and water 3. a woman is mixing ingredients
	Generated Sentence: SA: someone is slicing a carrot M^3 : a man is slicing a carrot with a knives	Reference Sentence: 1. a man cuts a carrot in half 2. the man is cutting carrots 3. a man is cutting carrots with a knife

Figure 2. Descriptions generated by SA-google, our M^3 -google and human-annotated ground truth on the test set of MSVD. We can see that, M^3 -google generates more relevant object terms than SA-google (“basketball” vs. “soccer ball”), and M^3 -google places more focus on the described targets than SA-google (“dog” vs. “guitar”). In particular, M^3 -google can generate longer sentences to describe more visual contents, e.g., “mixing ingredients in a bowl”, “slicing a carrot with a knives”.

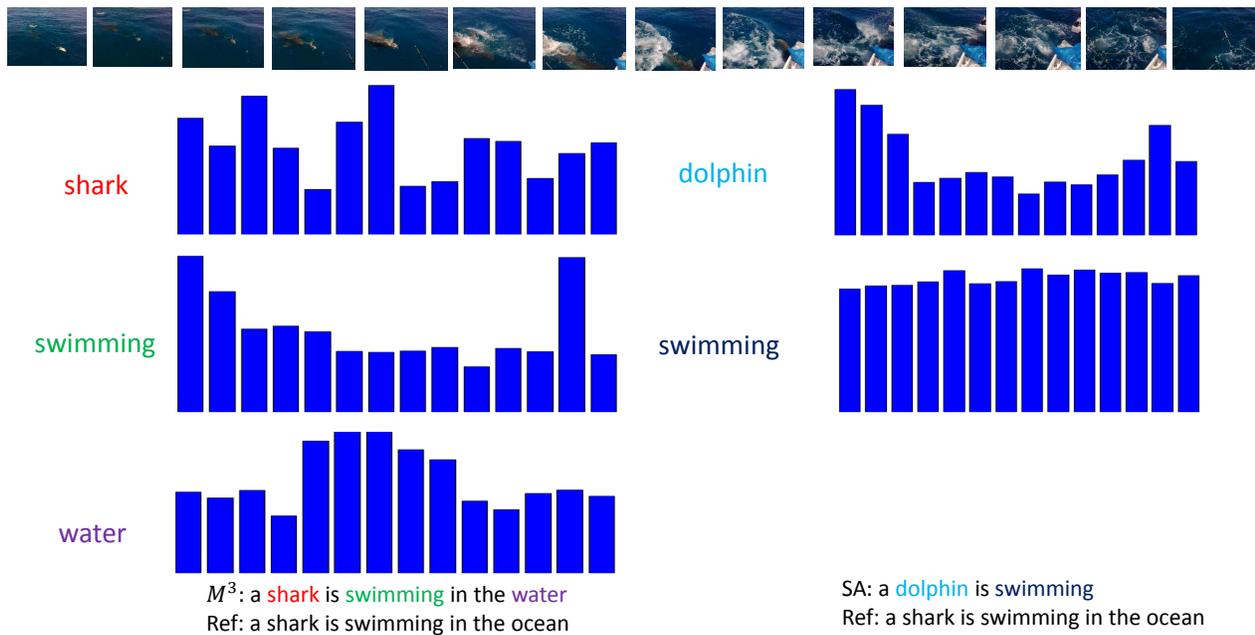


Figure 3. The attention shift of our M^3 -google and SA-google [37] across 14 sampled frames when generating the sentence. The attention weights of several generated key words corresponding to the 14 frames are shown as bar charts.

tain better results than VGG-19 features [32].

When using multiple visual feature fusion, we compare our model with the other five state-of-the-art

approaches([37], [30], [21], [38], [2]). The comparison results are shown in Table 2. SA-G-3C [37] uses the combination of GoogleNet feature and 3D-CNN feature. S2VT-rgb-

flow [30] uses the two-stream features consisting of RGB feature extracted from VGG-16 networks and optical flow feature extracted from AlexNet [18]. Both LSTM-E-VC [21] and p-RNN-VC [38] combine VGG-19 feature and C3D feature. We propose M^3 -VC and M^3 -IC for comparison. M^3 -VC also uses VGG-19 feature and C3D feature while M^3 -IC uses Inception-V3 feature and C3D feature. They all perform better than the other methods in terms of the two metrics, which proves the effectiveness of our model.

4.5.2 Experimental Results on MSR-VTT

MSR-VTT is a recently released benchmark dataset [36] which has the largest number of video-sentence pairs. Considering that there are few methods tested on this dataset, we compare our model with SA [37] which is the most similar work to ours. Similarly, we perform experiments with these two methods using single visual feature and multiple visual feature fusion simultaneously. The comparison results are reported in Table 3. SA-V and SA-C use the VGG-19 feature and C3D feature, respectively. SA-VC fuses these two kinds of features. Our M^3 -V, M^3 -C and M^3 -VC use the same features with the corresponding SA methods. It can be seen that our methods consistently outperform the corresponding SAs. The improved performance proves the importance of multimodal memory in our M^3 again. In addition, from either M^3 or SA, we can see that the results from C3D feature are generally better than those using VGG-19 feature. It may be that the motion information is very critical for the video representation in this dataset, because C3D feature encodes both visual appearance and motion information in video.

4.6. Qualitative Analysis

We evaluate our model with BLEU (n-gram) and METEOR above, which quantitatively reveal the relevance between generated sentence and human-annotated sentence. In this section, we will qualitatively analyze our model through visualizing the generated sentences and the corresponding attention shift across visual frames. Fig. 2 illustrates several descriptions generated by our M^3 -google, SA-google [37] and human-annotated ground truth on the test set of MSVD. We can see that our M^3 -google generates more relevant object terms than SA-google (“basketball” vs. “soccer ball” in the third video), and M^3 -google places more focus on the described targets than SA-google (“dog” vs. “guitar” in the second video). Particularly, M^3 -google can generate longer sentences to describe more visual contents, e.g., “mixing ingredients in a bowl” and “slicing a carrot with a knives” in the final two videos. All these results demonstrate the effectiveness of our method.

Fig. 3 shows the attention shift of our M^3 -google and

SA-google [37] across multiple frames when generating the sentence. There are 14 frames sampled from a testing video in MSVD, our M^3 -google generates “a shark is swimming in the water” while SA-google generates “a dolphin is swimming”. The attention weights of several generated key words corresponding to the 14 frames are shown as bar charts. We can see that the two methods show very different attention distributions for each word. It can be seen that the generated sentence is very relevant to the semantic object and action of the video, which further demonstrate the correctness that the proposed M^3 can guide the attention by multimodal memory modelling. Compared with SA, our model not only can identify the object (‘shark’ vs ‘dolphin’), but also can attend to specific frames relevant to the object.

5. Conclusions

This paper has proposed a Multimodal Memory Model to describe videos, which builds a visual and textual shared memory to model the long-term visual-textual dependency and further guide visual attention. The extensive experimental results on two publicly available benchmark datasets demonstrate that our method outperforms the state-of-the-art methods in terms of BLEU and METEOR metrics.

As we can see from the experimental results, video representation is very important for the performance of video captioning. In the future, we will consider to improve video representation learning algorithm, and integrate video feature extraction networks with multimodal memory networks to form an end-to-end deep learning system.

6. Acknowledgements

This work is jointly supported by National Key Research and Development Program of China (2016YFB1001000), National Natural Science Foundation of China (61525306, 61633021, 61721004, 61572504). In addition, this work is also supported by grants from NVIDIA and the NVIDIA DGX-1 AI Supercomputer.

References

- [1] N. Ballas, L. Yao, C. Pal, and A. Courville. Delving deeper into convolutional networks for learning video representations. *arXiv:1511.06432*, 2015.
- [2] L. Baraldi, C. Grana, and R. Cucchiara. Hierarchical boundary-aware neural encoder for video captioning. In *CVPR*, 2017.
- [3] A. Bordes, N. Usunier, S. Chopra, and J. Weston. Large-scale simple question answering with memory networks. *arXiv:1506.02075*, 2015.
- [4] D. L. Chen and W. B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, 2011.

- [5] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv:1504.00325*, 2015.
- [6] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv:1406.1078*, 2014.
- [7] M. Denkowski and A. Lavie. Meteor universal: Language specific translation evaluation for any target language. In *ACL*, 2014.
- [8] R. Fakoor, A.-r. Mohamed, M. Mitchell, S. B. Kang, and P. Kohli. Memory-augmented attention modelling for videos. *arXiv preprint arXiv:1611.02261*, 2016.
- [9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.
- [10] A. Graves, G. Wayne, and I. Danihelka. Neural Turing machines. *arXiv:1410.5401*, 2014.
- [11] E. Grefenstette, K. M. Hermann, M. Suleyman, and P. Blunsom. Learning to transduce with unbounded memory. In *NIPS*, 2015.
- [12] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *ICCV*, 2013.
- [13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016.
- [14] B. Jimmy, G. E. Hinton, V. Mnih, J. Leibo, and C. Ionescu. Using fast weights to attend to the recent past. In *NIPS*, 2016.
- [15] A. Joulin and T. Mikolov. Inferring algorithmic patterns with stack-augmented recurrent nets. In *NIPS*, 2015.
- [16] N. Kalchbrenner and P. Blunsom. Recurrent continuous translation models. In *EMNLP*, 2013.
- [17] N. Krishnamoorthy, G. Malkarnenkar, R. J. Mooney, K. Saenko, and S. Guadarrama. Generating natural-language video descriptions using text-mined knowledge. In *AAAI*, 2013.
- [18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [19] A. Kumar Jain, A. Agarwalla, K. Krishna Agrawal, and P. Mitra. Recurrent memory addressing for describing videos. In *CVPR Workshops*, 2017.
- [20] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. *arXiv:1511.03476*, 2015.
- [21] Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui. Jointly modeling embedding and translation to bridge video and language. *arXiv:1505.01861*, 2015.
- [22] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.
- [23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.
- [24] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.
- [25] S. Sukhbaatar, J. Weston, R. Fergus, et al. End-to-end memory networks. In *NIPS*, 2015.
- [26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
- [27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv:1512.00567*, 2015.
- [28] J. Thomason, S. Venugopalan, S. Guadarrama, K. Saenko, and R. J. Mooney. Integrating language and vision to generate natural language descriptions of videos in the wild. In *COLING*, 2014.
- [29] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.
- [30] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *ICCV*, 2015.
- [31] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv:1412.4729*, 2014.
- [32] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.
- [33] W. Wang, C. Chen, Y. Wang, T. Jiang, F. Fang, and Y. Yao. Simulating human saccadic scanpaths on natural images. In *CVPR*, 2011.
- [34] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv:1410.3916*, 2014.
- [35] C. Xiong, S. Merity, and R. Socher. Dynamic memory networks for visual and textual question answering. *arXiv:1603.01417*, 2016.
- [36] J. Xu, T. Mei, T. Yao, and Y. Rui. Msr-vtt: A large video description dataset for bridging video and language. In *CVPR*, 2016.
- [37] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *ICCV*, 2015.
- [38] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. Video paragraph captioning using hierarchical recurrent neural networks. *arXiv:1510.07712*, 2015.
- [39] W. Zaremba, I. Sutskever, and O. Vinyals. Recurrent neural network regularization. *arXiv:1409.2329*, 2014.
- [40] M. D. Zeiler. Adadelata: an adaptive learning rate method. *arXiv:1212.5701*, 2012.