Hierarchical LSTM with Adjusted Temporal Attention for Video Captioning

Jingkuan Song[1], **Zhao Guo**[1], **Lianli Gao**[1], **Wu Liu**[2], **Dongxiang Zhang**[1], **Heng Tao Shen**[1]
[1]Center for Future Media and School of Computer Science and Engineering,
University of Electronic Science and Technology of China, Chengdu 611731, China.
[2]Beijing University of Posts and Telecommunications, Beijing 100876, China.
jingkuan.song@gmail.com, {zhao.guo, lianli.gao, zhangdo}@uestc.edu.cn,
liuwu@bupt.edu.cn, shenhengtao@hotmail.com

Abstract

Recent progress has been made in using attention based encoder-decoder framework for video captioning. However, most existing decoders apply the attention mechanism to every generated word including both visual words (e.g., "gun" and "shooting") and non-visual words (e.g. "the", "a"). However, these non-visual words can be easily predicted using natural language model without considering visual signals or attention. Imposing attention mechanism on non-visual words could mislead and decrease the overall performance of video captioning. To address this issue, we propose a hierarchical LSTM with adjusted temporal attention (hLSTMat) approach for video captioning. Specifically, the proposed framework utilizes the temporal attention for selecting specific frames to predict the related words, while the adjusted temporal attention is for deciding whether to depend on the visual information or the language context information. Also, a hierarchical LSTMs is designed to simultaneously consider both low-level visual information and high-level language context information to support the video caption generation. To demonstrate the effectiveness of our proposed framework, we test our method on two prevalent datasets: MSVD and MSR-VTT, and experimental results show that our approach outperforms the state-of-the-art methods on both two datasets.

1 Introduction

Previously, visual content understanding [Song *et al.*, 2016; Gao *et al.*, 2017] and natural language processing (NLP) are not correlative with each other. Integrating visual content with natural language learning to generate descriptions for images, especially for videos, has been regarded as a challenging task. Video captioning is a critical step towards machine intelligence and many applications in daily scenarios, such as video retrieval [Wang *et al.*, 2017; Song *et al.*, 2017], video understanding, blind navigation and automatic video subtitling.

Thanks to the rapid development of deep Convolutional Neural Network (CNN), recent works have made significant progress for image captioning [Vinyals *et al.*, 2015; Xu *et al.*, 2015; Lu *et al.*, 2016; Karpathy *et al.*, 2014; Fang *et al.*, 2015; Chen and Zitnick, 2014; Chen *et al.*, 2016]. However, compared with image captioning, video captioning is more difficult due to the diverse sets of objects, scenes, actions, attributes and salient contents. Despite the difficulty there have been a few attempts for video description generation [Venugopalan *et al.*, 2014; Venugopalan *et al.*, 2015; Yao *et al.*, 2015; Li *et al.*, 2015; Gan *et al.*, 2016a], which are mainly inspired by recent advances in translating with Long Short-Term Memory (LSTM). The LSTM is proposed to overcome the vanishing gradients problem by enabling the network to learn when to forget previous hidden states and when to update hidden states by integrating memory units. LSTM has been successfully adopted to several tasks, e.g., speech recognition, language translation and image captioning [Cho *et al.*, 2015; Venugopalan *et al.*, 2014]. Thus, we follow this elegant recipe and choose to extend LSTM to generate the video sentence with semantic content.

Early attempts were proposed [Venugopalan *et al.*, 2014; Venugopalan *et al.*, 2015; Yao *et al.*, 2015; Li *et al.*, 2015] to directly connect a visual convolution model to a deep LSTM networks. For example, Venugopalan *et al.* [Venugopalan *et al.*, 2014] translate videos to sentences by directly concatenating a deep neural network with a recurrent neural network. More recently, attention mechanism [Gu *et al.*, 2016] is a standard part of the deep learning toolkit, contributing to impressive results in neural machine translation [Luong *et al.*, 2015], visual captioning [Xu *et al.*, 2015; Yao *et al.*, 2015] and question answering [Yang *et al.*, 2016]. Visual attention models for video captioning make use of video frames at every time step, without explicitly considering the semantic attributes of the predicted words. For example, in Fig. 1, some words (i.e., "man", "shooting" and "gun") belong to visual words which have corresponding canonical visual signals, while other words (i.e., "the", "a" and "is") are non-visual words, which require no visual information but language context information [Lu *et al.*, 2016]. In other words, current visual attention models make use of visual information for generating each work, which is unnecessary or even misleading. Ideally, video description not only requires modeling and integrating their sequence dynamic temporal attention information into a natural language but also needs to take into account the relationship between sentence semantics

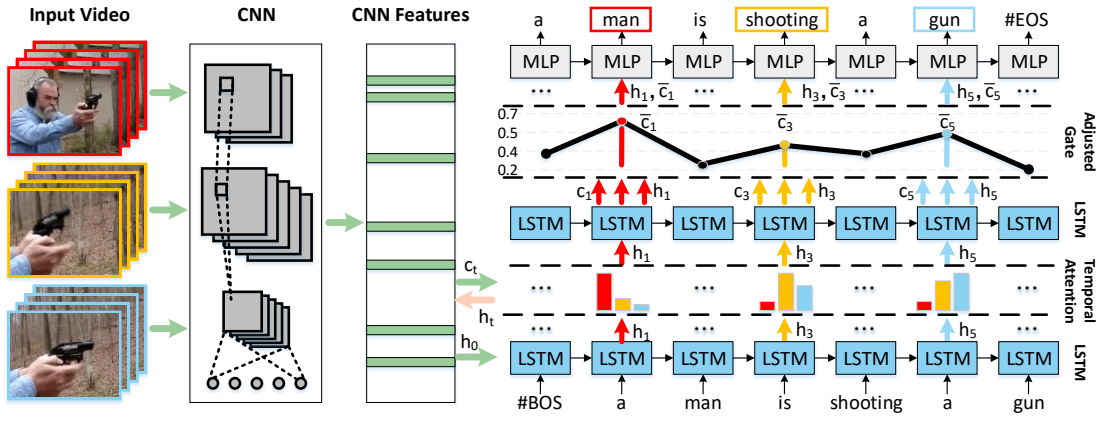

Figure 1: The framework of our proposed method hLSTMat. To illustrate the effectiveness of our hLSTMat, each generated visual words (i.e., "man", "shooting" or "gun") is generated with visual information extracting from a set of specific frames. For instance, "man" is marked with "red", this indicates it is generated by using the frames marked with red bounding boxes, "shooting" is generated replying on the frames marked with "orange". Other non-visual words such as "a" and "is" are relying on the language model.

and visual content [Gan *et al.*, 2016b], which to our knowledge has not been simultaneously considered.

To tackle these issues, inspired by the attention mechanism for image captioning [Lu *et al.*, 2016], in this paper we propose a unified encoder-decoder framework (see Fig. 1), named hLSTMat, a Hierarchical LSTMs with adjusted temporal attention model for video captioning. Specifically, first, in order to extract more meaningful spatial features, we adopt a deep neural network to extract a 2D CNN feature vector for each frame. Next, we integrate a hierarchical LSTMs consisting of two layers of LSTMs, temporal attention and adjusted temporal attention to decode visual information and language context information to support the generation of sentences for videos description. Moreover, the proposed novel adjusted temporal attention mechanism automatically decides whether to rely on visual information or not. When relying on visual information, the model enforces the gradients from visual information to support video captioning, and decides where to attend. Otherwise, the model predicts the words using natural language model without considering visual signals.

It is worthwhile to highlight the main contributions of this proposed approach: 1) We introduce a novel hLSTMat framework which automatically decides when and where to use video visual information, and when and how to adopt the language model to generate the next word for video captioning. 2) We propose a novel adjusted temporal attention mechanism which is based on temporal attention. Specifically, the temporal attention is used to decide where to look at visual information, while the adjusted temporal model is designed to decide when to make use of visual information and when to rely on language model. A hierarchical LSTMs is designed to obtain low-level visual information and high-level language context information. 3) Experiments on two benchmark datasets demonstrate that our method outperforms the state-of-the-art methods in both BLEU and METEOR.

2 The Proposed Approach

In this section, first we briefly describe how to directly use the basic Long Short-Term Memory (LSTM) as the decoder for video captioning task. Then we introduce our novel encoder-decoder framework, named hLSTMat (see Fig. 1).

2.1 A Basic LSTM for Video Captioning

To date, modeling sequence data with Recurrent Neural Networks (RNNs) has shown great success in the process of machine translation, speech recognition, image and video captioning [Chen and Zitnick, 2014; Fang *et al.*, 2015; Venugopalan *et al.*, 2014; Venugopalan *et al.*, 2015] etc. Long Short-Term Memory (LSTM) is a variant of RNN to avoid the vanishing gradient problem [Bengio *et al.*, 1994].

LSTM Unit. A basic LSTM unit consists of three gates (input $\mathbf{i}_t$, forget $\mathbf{f}_t$ and output $\mathbf{o}_t$), a single memory cell $\mathbf{m}_t$. Specifically, $\mathbf{i}_t$ allows incoming signals to alter the state of the memory cell or block it. $\mathbf{f}_t$ controls what to be remembered or be forgotten by the cell, and somehow can avoid the gradient from vanishing or exploding when back propagating through time. Finally, $\mathbf{o}_t$ allows the state of the memory cell to have an effect on other neurons or prevent it. Basically, the memory cell and gates in a LSTM block are defined as follows:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{y}_t + \mathbf{U}_i h_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{y}_t + \mathbf{U}_f h_{t-1} + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{y}_t + \mathbf{U}_o h_{t-1} + \mathbf{b}_o) \\
\mathbf{g}_t &= \phi(\mathbf{W}_g\mathbf{y}_t + \mathbf{U}_g h_{t-1} + \mathbf{b}_g) \\
\mathbf{m}_t &= \mathbf{f}_t \odot \mathbf{m}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \phi(\mathbf{m}_t)
\end{aligned} \tag{1}$$

where the weight matrices $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{b}$ are parameters to be learned. $\mathbf{y}_t$ represents the input vector for the LSTM unit at each time t. σ represents the logistic sigmoid nonlinear activation function mapping real numbers to $(0, 1)$, and it can be thought as knobs that LSTM learns to selec-

tively forget its memory or accept the current input. ϕ denotes the hyperbolic tangent function tanh. $\odot$ is the element-wise product with the gate value. For convenience, we denote $\mathbf{h}_t, \mathbf{m}_t = \text{LSTM}(\mathbf{y}_t, \mathbf{h}_{t-1}, \mathbf{m}_{t-1})$ as the computation function for updating the LSTM internal state.

Video Captioning. Given a video input $\mathbf{x}$, an encoder network ϕ_E encodes it into a continuous representation space:

$$\mathbf{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_n\} = \phi_E(\mathbf{x}). \tag{2}$$

where ϕ_E usually denotes a CNN neural network, n denotes the number of frames in $\mathbf{x}$, $\mathbf{v}_i \in \mathbb{R}^d$ is the frame-level feature of the i-th frame, and it is d-dimensional. Here, LSTM is chosen as a decoder network ϕ_D to model $\mathbf{V}$ to generate a description $\mathbf{z} = \{z_1, \cdots, z_T\}$ for $\mathbf{x}$, where T is the description length. In addition, the LSTM unit updates its internal state $\mathbf{h}_t$ and the t-th word z_t based on its previous internal state $\mathbf{h}_{t-1}$, the previous output y_t and the representation $\mathbf{V}$:

$$(\mathbf{h}_t, z_t) = \phi_D(\mathbf{h}_{t-1}, y_t, \mathbf{V})$$

In addition, the LSTM updates its internal state recursively until the end-of-sentence tag is generated. For simplicity, we named this simple method as basic-LSTM.

2.2 Hierarchical LSTMs with Adjusted Temporal Attention for Video Captioning

In this subsection, we introduce our hLSTMat framework, which consists of two components: 1) a CNN Encoder and 2) an attention based hierarchical LSTM decoder.

CNN Encoders

The goal of an encoder is to compute feature vectors that are compact and representative and can capture the most related visual information for the decoder. Thanks to the rapid development of deep convolutional neural networks (CNNs), which have made a great success in large scale image recognition task [He *et al.*, 2016], object detection [Ren *et al.*, 2015] and visual captioning [Venugopalan *et al.*, 2014]. High-level features can been extracted from upper or intermediate layers of a deep CNN network. Therefore, a set of well-tested CNN networks, such as the ResNet-152 model [He *et al.*, 2016] which has achieved the best performance in ImageNet Large Scale Visual Recognition Challenge, can be used as a candidate encoder for our framework.

Attention based Hierarchical Decoder

Our decoder (see Fig. 2) integrates two LSTMs. The bottom LSTM layer is used to efficiently decode visual features, and the top LSTM is focusing on mining deep language context information for video captioning. We also incorporate two attention mechanisms into our framework. A temporal attention is to guide which frame to look, while the adjusted temporal attention is proposed to decide when to use visual information and when to use sentence context information. The top MLP layer is to predict the probability distribution of each word in the vocabulary.

Unlike vanilla LSTM decoder, which performs mean pooling over 2D features across each video to form a fixed-dimension representation, attention based LSTM decoder is focusing on a subset of consecutive frames to form a fixed-dimensional representation at each time t.

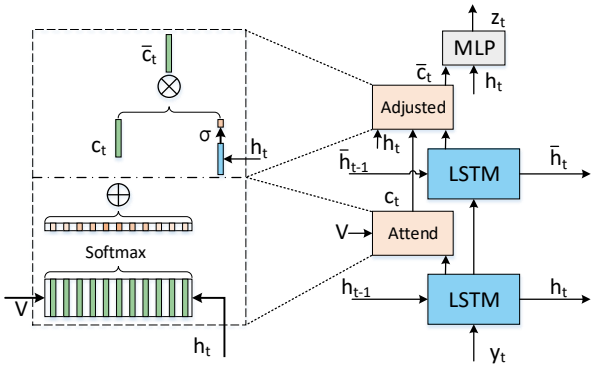

Figure 2: An illustration of the proposed method generating the t-th target word z_t given a video.

- Bottom LSTM Layer. For the bottom LSTM layer, the updated internal hidden state depends on the current word y_t, previous hidden state $\mathbf{h}_{t-1}$ and memory state $\mathbf{m}_{t-1}$:

$$\begin{aligned} \mathbf{h}_0, \mathbf{m}_0 &= \left[\mathbf{W^{ih}}; \mathbf{W^{ic}}\right] Mean(\{\mathbf{v_i}\}) \\ \mathbf{h}_t, \mathbf{m}_t &= LSTM(\mathbf{y}_t, \mathbf{h}_{t-1}, \mathbf{m}_{t-1}) \end{aligned} \tag{3}$$

  where $\mathbf{y}_t = \mathbf{E}[y_t]$ denotes a word feature of a single word y_t. $Mean(\cdot)$ denotes a mean pooling of the given feature set $\mathbf{v_i}$. $\mathbf{W^{ih}}$ and $\mathbf{W^{ic}}$ are parameters that need to be learned.

- Top LSTM Layer. For the top LSTM, it takes the output of the bottom LSTM unit output $\mathbf{h}_t$, previous hidden state $\bar{\mathbf{h}}_{t-1}$ and the memory state $\bar{\mathbf{m}}_{t-1}$ as inputs to obtain the hidden state $\bar{\mathbf{h}}_t$ at time t, and it can be defined as below:

$$\bar{\mathbf{h}}_t, \bar{\mathbf{m}}_t = LSTM(\mathbf{h}_t, \bar{\mathbf{h}}_{t-1}, \bar{\mathbf{m}}_{t-1}) \tag{4}$$

- Attention Layers. In addition, for attention based LSTM, context vector is in general an important factor, since it provides meaningful visual evidence for caption generation [Yao *et al.*, 2015]. In order to efficiently adjust the choose of visual information or sentence context information for caption generation, we defined an adjusted temporal context vector $\bar{\mathbf{c}}_t$ and a temporal context vector $\mathbf{c}_t$ at time t. See below:

$$\bar{\mathbf{c}}_t = \psi(\mathbf{h}_t, \bar{\mathbf{h}}_t, \mathbf{c}_t), \quad \mathbf{c}_t \quad = \varphi(\mathbf{h}_t, \mathbf{V}) \tag{5}$$

  where ψ denotes the function of our adjust gate, while φ denotes the function of our temporal attention model. Moreover, $\bar{\mathbf{c}}_t$ denotes the final context vector through our adjusted gate, and $\mathbf{c}_t$ represents intermediate vectors calculated by our temporal attention model. These two attention layers will be described in details in Sec. 2.3 and Sec. 2.4.

- MLP layer. To output a symbol z_t, a probability distribution over a set of possible words is obtained using $\mathbf{h}_t$ and our adjusted temporal attention vector $\bar{\mathbf{c}}_t$:

$$\mathbf{p}_t = softmax\left(\mathbf{U}_p \phi(\mathbf{W}_p[\mathbf{h}_t; \bar{\mathbf{c}}_t] + \mathbf{b}_p) + \mathbf{d}\right) \tag{6}$$

where $\mathbf{U}_p$, $\mathbf{W}_p$, $\mathbf{b}_p$ and $\mathbf{d}$ are parameters to be learned. Next, we can interpret the output of the softmax layer $\mathbf{p}_t$ as a probability distribution over words:

$$P(z_t|z_{<t}, \mathbf{V}, \Theta) \tag{7}$$

where $\mathbf{V}$ denotes the features of the corresponding input video, and Θ are model parameters.

To learn Θ in our modal, we minimize the negative logarithm of the likelihood:

$$min_{\Theta} - \sum_{t=1}^{T} logP(z_t|z_{<t}, \mathbf{V}, \Theta) \tag{8}$$

where T denotes the total number of words in sentence. Therefore, Eq.8 is regarded as our loss function to optimize our model.

After the parameters are learned, we choose BeamSearch [Vinyals *et al.*, 2015] method to generate sentences for videos, which iteratively considers the set of the k best sentences up to time t as candidates to generate sentence of time $t+1$, and keeps only best k results of them. Finally, we approximates $D = argmax_{D'} Pr(D^{'}|X)$ as our best generated description. In our entire experiment, we set the beam size of BeamSearch as 5.

2.3 Temporal Attention Model

As mentioned above, context vector $\mathbf{c}_t$ is an important factor in encoder-decoder framework. To deal with the variability of the length of videos, a simple strategy [Venugopalan *et al.*, 2014] is used to compute the average of features across a video, and this generated feature is used as input to the model at each time step:

$$\mathbf{c}_t = \frac{1}{n} \sum_{i=1}^{n} \mathbf{v}_i \tag{9}$$

However, this strategy effectively collapses all frame-level features into a single vector, neglecting the inherent temporal structure and leading to the loss of information. Instead of using a simple average strategy (see Eq. 9), we wish to take the dynamic weight sum of the temporal feature vectors according to attention weights α_t^i, which are calculated by a soft attention. For each $\mathbf{v}_i$ at t time step, we use the follow function to calculate $\mathbf{c}_t$:

$$\mathbf{c}_t = \frac{1}{n} \sum_{i=1}^{n} \alpha_t^i \mathbf{v}_i \tag{10}$$

where at t time step $\sum_{i=1}^{n} \alpha_t^i = 1$.

In this paper, we integrate two LSTM layers, a novel temporal attention model for computing the context vector $\mathbf{c}_t$ in Eq. 5 proposed in our framework. Given a set of video features $\mathbf{V}$ and the current hidden state of the bottom layer LSTM $\mathbf{h}_t$, we feed them into a single neural network layer, and it returns an unnormalized relevant scores ε_t. Finally, a softmax function is applied to generate the attention distribute over the n frames of the video:

$$\begin{aligned} \varepsilon_t &= \mathbf{w}^T tanh\left(\mathbf{W}_a \mathbf{h}_t + \mathbf{U}_a \mathbf{V} + \mathbf{b}_a\right) \\ \alpha_t &= softmax(\varepsilon_t) \end{aligned} \tag{11}$$

where $\mathbf{w}^T$, $\mathbf{W}_a$, $\mathbf{U}_a$ and $\mathbf{b}_a$ are parameters to be learned. $\alpha_t \in \mathbb{R}^n$ is the attention weight which quantifies the relevance of features in $\mathbf{V}$.

Different from [Yao *et al.*, 2015], we utilize the current hidden state instead of previous hidden state $\mathbf{h}_t$ generated by the first LSTM layer to obtain the context vector $\mathbf{c}_t$, which focuses on salient feature in the video.

2.4 Adjusted Temporal Attention Model

In this paper we propose an adjusted temporal attention model to compute a context vector $\bar{\mathbf{c}}_t$ in Eq. 5, shown in Fig. 2, to make sure that a decoder uses nearly no visual information from video frames to predict the non-visual words, and use the most related visual information to predict visual words. In our hierarchical LSTM network, the hidden state in the bottom LSTM layer is a latent representation of what the decoder already knows. With the hidden state $\mathbf{h}_t$, we extend our temporal attention model, and propose an adjusted model that is able to determine whether it needs to attend the video to predict the next word. In addition, a sigmoid function is applied to the hidden state $\mathbf{h}_t$ to further filter visual information.

$$\begin{aligned} \bar{\mathbf{c}}_t &= \beta_t \mathbf{c}_t + (1 - \beta_t) \bar{\mathbf{h}}_t \\ \beta_t &= sigmoid(\mathbf{W}_s \mathbf{h}_t) \end{aligned} \tag{12}$$

where $\mathbf{W}_s$ denotes the parameters to be learned and β_t is adjusted gate at each time t. In our adjusted temporal attention model, β_t is projected into the range of $[0, 1]$. When $\beta_t = 1$, it indicates that full visual information is considered, while when $\beta_t = 0$ it indicates that none visual information is considered to generate the next word.

3 Experiments

We evaluate our algorithm on the task of video captioning. Specifically, we firstly study the influence of CNN encoders. Secondly, we explore the effectiveness of the proposed components. Next, we compare our results with the state-of-the-art methods.

3.1 Datasets

We consider two publicly available datasets that have been widely used in previous work.

The Microsoft Video Description Corpus (MSVD). This video corpus consists of 1,970 short video clips, approximately 80,000 description pairs and about 16,000 vocabulary words [Chen and Dolan, 2011]. Following [Yao *et al.*, 2015; Venugopalan *et al.*, 2015], we split the dataset into training, validation and testing set with 1,200, 100 and 670 videos, respectively.

MSR Video to Text (MSR-VTT). In 2016, Xu *et al.* [Xu *et al.*, 2016] proposed a currently largest video benchmark for video understanding, and especially for video captioning. Specifically, this dataset contains 10,000 web video clips, and each clip is annotated with approximately 20 natural language sentences. In addition, it covers the most comprehensive categorizes (i.e., 20 categories) and a wide variety of visual content, and contains 200,000 clip-sentence pairs.

Table 1: Experiment results on the MSVD dataset. We use different features to verify our hLSTMat method.

Model	B@1	B@2	B@3	B@4	METEOR
C3D	79.9	68.2	58.3	47.5	30.5
GoogleNet	80.8	68.6	58.9	48.5	31.9
Inception-v3	82.7	72.0	62.5	51.9	33.5
ResNet-50	80.9	69.1	59.5	49.0	32.3
ResNet-101	82.2	70.9	61.4	50.8	32.7
ResNet-152	**82.9**	**72.2**	**63.0**	**53.0**	**33.6**

3.2 Implementation Details

Preprocessing

For MSVD dataset, we convert all descriptions to lower cases, and then use wordpunct_tokenizer method from NLTK toolbox to tokenize sentences and remove punctuations. Therefore, it yields a vocabulary of 13,010 in size for the training split. For MSR-VTT dataset, captions have been tokenized, thus we directly split descriptions using blank space, thus it yields a vocabulary of 23,662 in size for training split. Inspired by [Yao *et al.*, 2015], we preprocess each video clip by selecting equally-spaced 28 frames out of the first 360 frames and then feeding them into a CNN network proposed in [He *et al.*, 2016]. Thus, for each selected frame we obtain a 2,048-D feature vector, which are extracted from the $pool5$ layer.

Training details

In the training phase, in order to deal with sentences with arbitrary length, we add a begin-of-sentence tag <BOS> to start each sentence and an end-of-sentence tag <EOS> to end each sentence. In the testing phase, we input <BOS> tag into our attention-based hierarchical LSTM to trigger video description generation process. For each word generation, we choose the word with the maximum probability and stop until we reach <EOS>.

In addition, all the LSTM unit sizes are set as 512 and the word embedding size is set as 512, empirically. Our objective function Eq. 8 is optimized over the whole training video-sentence pairs with mini-batch 64 in size of MSVD and 256 in size of MSR-VTT. We adopt adadelta [Zeiler, 2012], which is an adaptive learning rate approach, to optimize our loss function. In addition, we utilize dropout regularization with the rate of 0.5 in all layers and clip gradients element wise at 10. We stop training our model until 500 epochs are reached, or until the evaluation metric does not improve on the validation set at the patience of 20.

Evaluation metrics

To evaluate the performance, we employ two different standard evaluation metrics: BLUE [Papineni *et al.*, 2002] and METEOR [Banerjee and Lavie, 2005].

3.3 The Effect of Different CNN Encoders

To date, there are 6 widely used CNN encoders including C3D, GoogleNet, Inception-V3, ResNet-50, ResNet-101 and ResNet-152 to extract visual features. In this sub-experiment, we study the influence of different versions of CNN encoders on our framework. The experiments are conducted on the MSVD dataset, and the results are shown in Tab. 1. By observing Tab. 1, we find that by taking ResNet-152 as the visual decoder, our method performs best with 82.9% B@1, 72.2% B@2, 63.0% B@3, 53.0% B@4 and 33.6% METEOR, while Inception-v3 is a strong competitor, with 82.7% B@1, 72.0% B@2, 62.5% B@3, 51.9% B@4 and 33.5% METEOR. However, the gap between ResNet-152 and Inception-v3 is very small.

3.4 Architecture Exploration and Comparison

In this sub-experiment, we explore the impact of three proposed components, including basic LSTM proposed in Sec.2.1 (basic LSTM), hLSTMt which removes the adjusted mechanism from the hLSTMat, and hLSTMat, as well as comparing them with the state of the art methods: MP-LSTM [Venugopalan *et al.*, 2014] and SA [Yao *et al.*, 2015]. In order to conduct a fair comparison, all the methods take ResNet-152 as the encoder. We conduct the same experiments on the MSVD dataset. The experimental results are shown in Tab. 2. It shows that our hLSTMat achieves the best results in all metrics with 82.9% B@1, 72.2% B@2, 63.0% B@3, 53.0% B@4 and 33.6% METEOR. Also, by comparing with SA which take previous hidden state to calculate temporal attention weight, our hLSTMt performs better for video captioning. Moreover, by comparing with hLSTMt, we find that adjusted attention mechanism can improve the performance of video captioning. We also add one-layer LSTM and adjusted attention as an additional baseline. Results show that the adjusted attention mechanism can improve the performance.

3.5 Compare with the-state-of-the-art Methods

Results on MSVD dataset

In this subsection, we show the comparison of our approach with the baselines on the MSVD dataset. Some of the above baselines only utilize video features generated by a single deep network, while others (i.e., S2VT, LSTM-E and p-RNN) make uses of both single network and multiple network generated features. Therefore, we first compare our method with approaches using static frame-level features extracted by a single network. In addition, we compare our method with methods utilized different deep features or their combinations. The results are shown in Tab.3. When using static frame-level features, we have the following observations:

1) Compared with the best counterpart (i.e., p-RNN) which only takes spatial information, our method has 8.7% improvement on B@4 and 2.5% on METEOR.

2) The hierarchical structure in HRNE reduces the length of input flow and composites multiple consecutive input at a higher level, which increases the learning capability and enables the model encode richer temporal information of multiple granularities. Our approach (53.0% B@4, 33.6% METEOR) performs better than HRNE (43.6% B@4, 32.1% METEOR) and HRNE-SA (43.8% B@4, 33.1% METEOR). This shows the effectiveness of our model.

3) Our hLSTMat (53.0% B@4, 33.6% METEOR) can achieve better results than our hLSTMt (52.1% B@4, 33.3% METEOR). This indicates that it is beneficial to incorporate the adjusted temporal attention into our framework.

On the other hand, utilizing both spatial and temporal video information can enhance the video caption performance. VGGNet and GoogleNet are used to generate spatial information,

Table 2: The effect of different components and the comparison with the state-of-the-art methods on the MSVD dataset. The default encoder for all methods is ResNet-152.

Model	B@1	B@2	B@3	B@4	METEOR	CIDEr
basic LSTM	80.6	69.3	59.7	49.6	32.7	69.9
MP-LSTM [Venugopalan *et al.*, 2014]	81.1	70.2	61.0	50.4	32.5	71.0
SA [Yao *et al.*, 2015]	81.6	70.3	61.6	51.3	33.3	72.0
basic+adjusted attention	80.9	69.7	61.1	50.2	31.6	71.5
hLSTMt	82.5	71.9	62.0	52.1	33.3	73.5
hLSTMat	**82.9**	**72.2**	**63.0**	**53.0**	**33.6**	**73.8**

Table 3: The performance comparison with the state-of-the-art methods on MSVD dataset. (V) denotes VGGNet, (O) denotes optical flow, (G) denotes GoogleNet, (C) denotes C3D and (R) denotes ResNet-152.

Model	B@1	B@2	B@3	B@4	METEOR	CIDEr
S2VT(V) [Venugopalan *et al.*, 2015]	-	-	-	-	29.2	-
S2VT(V+O)	-	-	-	-	29.8	-
HRNE(G) [Pan *et al.*, 2016]	78.4	66.1	55.1	43.6	32.1	-
HRNE-SA (G)	79.2	66.3	55.1	43.8	33.1	-
LSTM-E(V)[Pan *et al.*, 2015]	74.9	60.9	50.6	40.2	29.5	-
LSTM-E(C)	75.7	62.3	52.0	41.7	29.9	-
LSTM-E(V+C)	78.8	66.0	55.4	45.3	31.0	-
p-RNN(V) [Yu *et al.*, 2016]	77.3	64.5	54.6	44.3	31.1	62.1
p-RNN(C)	79.7	67.9	57.9	47.4	30.3	53.6
p-RNN(V+C)	81.5	70.4	60.4	49.9	32.6	65.8
hLSTMt (R)	82.5	71.9	62.0	52.1	33.3	73.5
hLSTMat (R)	**82.9**	**72.2**	**63.0**	**53.0**	**33.6**	**73.8**

while optical flow and C3D are used for capturing temporal information. For example, compared with LSTM-E(V) and LSTM-E (C), LSTM-E(V+C) achieves higher 45.3% B@4 and 31.0% METEOR. In addition, for p-RNN, p-RNN(V+C) (49.9% B@4 and 32.6% METEOR) performs better than both p-RNN(V) (44.3% B@4 and 31.3% METEOR) and p-RNN(C) (47.4% B@4 and 30.3% METEOR).

Our approach achieves the best results (53.0% B@4 and 33.6% METEOR) using static frame-level features compared with approaches combining multiple deep features. For S2VT(V+O), LSTM-E(V+C) and p-RNN(V+C), they use two networks VGGNet/GoogleNet and optical flow/C3D to capture video's spatial and temporal information, respectively. Compared with them, our approach only utilizes ResNet-152 to capture frame-level features, which proves the effectiveness of our hierarchical LSTM with adjusted temporal attention model.

We adopt questionnaires collected from ten users with different academic backgrounds. Given a video caption, users are asked to score the following aspects: 1) Caption Accuracy, 2) Caption Information Coverage, 3) Overall Quality. Results show that our method outperforms others at 'Overall Quality', and 'Caption Accuracy' with small margin. But it has lower value for 'Information coverage' than p-RNN.

Results on MSR-VTT dataset

We compare our model with the state-of-the-art methods on the MSR-VTT dataset, and the results are shown in Tab. 4. Our model performs the best on all metrics, with 38.3% @B4 and 26.3% METEOR. Compared with our methods using only temporal attention, the performance is improved by 1.1% for @B4, and 0.2% for METEOR. This verifies the effectiveness of our method.

Table 4: The performance comparison with the state-of-the-art methods on MSR-VTT dataset.

Model	B@4	METEOR
MP-LSTM (V)	34.8	24.8
MP-LSTM (C)	35.4	24.8
MP-LSTM (V+C)	35.8	25.3
SA (V)	35.6	25.4
SA (C)	36.1	25.7
SA (V+C)	36.6	25.9
hLSTMt (R)	**37.4**	**26.1**
hLSTMat (R)	**38.3**	**26.3**

4 Conclusion and Future Work

In this paper, we introduce a novel hLSTMat encoder-decoder framework, which integrates a hierarchical LSTMs, temporal attention and adjusted temporal attention to automatically decide when to make good use of visual information or when to utilize sentence context information, as well as to simultaneously considering both low-level video visual features and language context information. Experiments show that hLSTMat achieves state-of-the-art performances on both MSVD and MSR-VTT datasets. In the future, we consider incorporating our method with both temporal and visual features to test the performance.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (Grant No. ZYGX2014J063, No. ZYGX2014Z007) and the National Natural Science Foundation of China (Grant No. 61502080, No. 61632007, No. 61602049).

References

[Banerjee and Lavie, 2005] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[Bengio *et al.*, 1994] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[Chen and Dolan, 2011] David L Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, pages 190–200, 2011.

[Chen and Zitnick, 2014] Xinlei Chen and C Lawrence Zitnick. Learning a recurrent visual representation for image caption generation. *arXiv preprint arXiv:1411.5654*, 2014.

[Chen *et al.*, 2016] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. *arXiv preprint arXiv:1611.05594*, 2016.

[Cho *et al.*, 2015] Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. Describing multimedia content using attention-based encoder-decoder networks. *Multimedia, IEEE Transactions on*, 17(11):1875–1886, 2015.

[Fang *et al.*, 2015] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *CVPR*, pages 1473–1482, 2015.

[Gan *et al.*, 2016a] Chuang Gan, Chen Sun, Lixin Duan, and Boqing Gong. Webly-supervised video recognition by mutually voting for relevant web images and web video frames. In *ECCV*, 2016.

[Gan *et al.*, 2016b] Zhe Gan, Chuang Gan, Xiaodong He, Yunchen Pu, Kenneth Tran, Jianfeng Gao, Lawrence Carin, and Li Deng. Semantic compositional networks for visual captioning. *CoRR*, abs/1611.08002, 2016.

[Gao *et al.*, 2017] Lianli Gao, Peng Wang, Jingkuan Song, Zi Huang, Jie Shao, and Heng Tao Shen. Event video mashup: From hundreds of videos to minutes of skeleton. In *AAAI*, pages 1323–1330, 2017.

[Gu *et al.*, 2016] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*, 2016.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[Karpathy *et al.*, 2014] Andrej Karpathy, Armand Joulin, and Fei Fei F Li. Deep fragment embeddings for bidirectional image sentence mapping. In *NIPS*, pages 1889–1897, 2014.

[Li *et al.*, 2015] Guang Li, Shubo Ma, and Yahong Han. Summarization-based video caption via deep neural networks. In *ACM Multimedia*, pages 1191–1194, 2015.

[Lu *et al.*, 2016] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. *arXiv preprint arXiv:1612.01887*, 2016.

[Luong *et al.*, 2015] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[Pan *et al.*, 2015] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. *arXiv preprint arXiv:1505.01861*, 2015.

[Pan *et al.*, 2016] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *CVPR*, pages 1029–1038, 2016.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318, 2002.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[Song *et al.*, 2016] Jingkuan Song, Lianli Gao, Feiping Nie, Heng Tao Shen, Yan Yan, and Nicu Sebe. Optimized graph learning using partial tags and multiple features for image and video annotation. *IEEE Transactions on Image Processing*, 25(11):4999–5011, 2016.

[Song *et al.*, 2017] Jingkuan Song, Lianli Gao, Li Liu, Xiaofeng Zhu, and Nicu Sebe. Quantization-based hashing: a general framework for scalable image and video retrieval. *Pattern Recognition*, 2017.

[Venugopalan *et al.*, 2014] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014.

[Venugopalan *et al.*, 2015] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *ICCV*, pages 4534–4542, 2015.

[Vinyals *et al.*, 2015] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[Wang *et al.*, 2017] Jingdong Wang, Ting Zhang, Jingkuan Song, Nicu Sebe, Heng Tao Shen, et al. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[Xu *et al.*, 2015] Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

[Xu *et al.*, 2016] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *CVPR*, 2016.

[Yang *et al.*, 2016] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *CVPR*, 2016.

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

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

[Zeiler, 2012] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.