SibNet: Sibling Convolutional Encoder for Video Captioning

Sheng Liu
State University of New York at Buffalo
Buffalo, New York
sliu66@buffalo.edu

Zhou Ren
Snap Research
Los Angeles, California
zhou.ren@snapchat.com

Junsong Yuan
State University of New York at Buffalo
Buffalo, New York
jsyuan@buffalo.edu

ABSTRACT

Video captioning is a challenging task owing to the complexity of understanding the copious visual information in videos and describing it using natural language. Different from previous work that encodes video information using a single flow, in this work, we introduce a novel Sibling Convolutional Encoder (SibNet) for video captioning, which utilizes a two-branch architecture to collaboratively encode videos. The first content branch encodes the visual content information of the video via autoencoder, and the second semantic branch encodes the semantic information by visual-semantic joint embedding. Then both branches are effectively combined with soft-attention mechanism and finally fed into a RNN decoder to generate captions. With our SibNet explicitly capturing both content and semantic information, the proposed method can better represent the rich information in videos. Extensive experiments on YouTube2Text and MSR-VTT datasets validate that the proposed architecture outperforms existing methods by a large margin across different evaluation metrics.

KEYWORDS

video captioning, visual-semantic joint embedding, autoencoder

ACM Reference Format:
Sheng Liu, Zhou Ren, and Junsong Yuan. 2018. SibNet: Sibling Convolutional Encoder for Video Captioning. In *MM '18: 2018 ACM Multimedia Conference, Oct. 22–26, 2018, Seoul, Republic of Korea.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3240508.3240667

1 INTRODUCTION

Video captioning aims to understand videos and summarize them concisely using natural language sentences [9, 25, 27, 44–46, 51]. Such an ability, which is a key element of machine intelligence, is crucial to many multimedia applications such as video retrieval, human-computer interaction, and video surveillance. By understanding the semantics of videos, video captioning characterizes visual information into languages and provides concise summarization of video data, which facilitates the effectiveness and efficiency of indexing, searching, and querying large video corpus.

MM '18, October 22–26, 2018, Seoul, Republic of Korea

ACM ISBN 978-1-4503-5665-7/18/10…$15.00
https://doi.org/10.1145/3240508.3240667

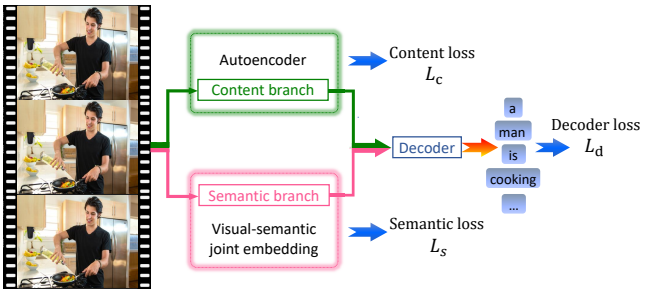


Figure 1: Overview of the proposed SibNet, which employs a two-branch architecture to collaboratively encode videos. The proposed loss function contains three components: a content loss L_c, a semantic loss L_s, and a decoder loss L_d. We leverage autoencoder and visual-semantic joint embedding to impose fine-grained regularization that pushes content branch to capture visual contents and pushes semantic branch to encode video semantics.

Motivated by the success of neural machine translation (NMT) [1] and neural image captioning [34, 54], deep neural network models with encoder-decoder pipeline have been applied to video captioning recently and achieved excellent performance [9, 29, 45, 51]. To transfer a sequence of images into a sequence of words, the encoder, e.g., an LSTM or CNN, compresses a video into a vector representation, and then a decoder, e.g., a RNN, helps to further transfer it into a sentence, i.e., a sequence of words following the syntax. Such a sequence-to-sequence learning pipeline has shown promising capacity of "translating" videos into sentences. However, the translation performance often relies on (1) the encoder that captures the visual information of the video, and also (2) the decoder that generates the sentence. Although the decoder can help ensure that the generated sentence is meaningful, we argue that the encoder would be even more important, because the information lost in the encoding phase could not be fully recovered by the decoder, thus resulting in imprecise or incomplete translation.

Existing video encoders choose to represent the whole video by merging conventional CNN features with average pooling or RNN that can capture the video's temporal structures. However, most of them only consider using one single branch to encode the video information. Different from a single image, a video is a sequence of images and conveys much richer information. Therefore, a single-branch video encoder may not provide sufficient representation of the video contents. To make a more holistic representation of videos, we propose Sibling Convolutional Encoder (SibNet), which is composed of two branches, i.e., the content branch and the semantic

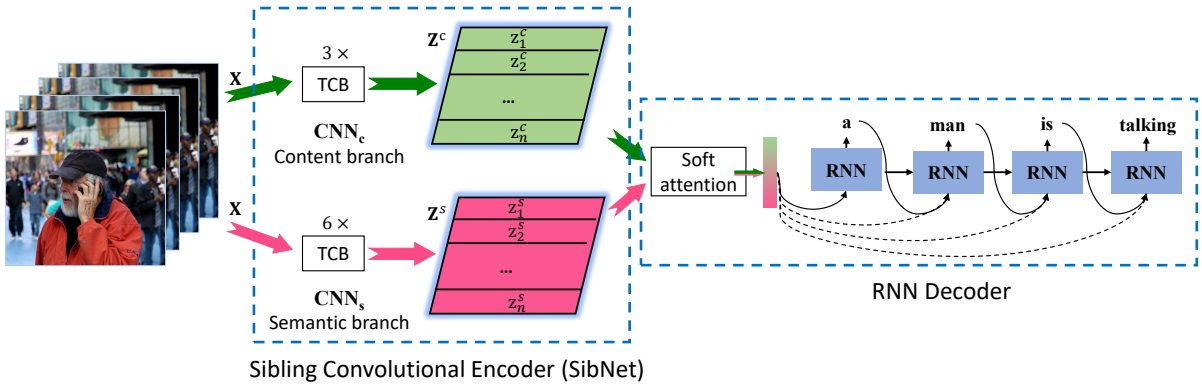


Figure 2: Illustration of the proposed Sibling Convolutional Encoder (SibNet), which is composed of the content branch and the semantic branch, denoted as CNN$_c$ and CNN$_s$, respectively. We construct both branches by stacking 3 and 6 identical temporal convolutional blocks (TCBs) (we will introduce TCB in Section 3.1.2). A soft-attention mechanism is utilized in our RNN decoder.

branch, to jointly encode the videos. The content branch explicitly learns visual representation of a video with an autoencoder, while the semantic branch encodes a video via visual-semantic joint embedding, which leverages the ground truth captions in the training data to generate semantic-specific representation. Finally, both branches are effectively combined with soft-attention mechanism and fed into the RNN decoder for caption generation. Our SibNet is specifically designed for video captioning task and it brings the following two advantages: (1) the content branch is able to faithfully capture the visual contents of the video. As it is a pure visual encoder, it can better capture the video details to provide more precise video captioning; (2) the semantic branch leverages visual-semantic joint embedding to produce semantic-specific representation. Such representation can capture how important certain frame is semantically, thus providing complementary information of the content branch.

To jointly train the encoder and decoder, we design a new loss function composed of three loss terms: (1) content loss from the content branch, (2) semantic loss from the semantic branch, and (3) decoder loss from the RNN decoder. In our joint optimization framework, these three loss terms regularize each other to ensure our SibNet generates an effective video representation that works well for video captioning. To model the temporal structures of the video and also make it compatible with our two-branch structure, our SibNet chooses to use temporal convolutional architecture, i.e., temporal convolutional block (TCB), to bring efficient video temporal encoding.

We evaluate the proposed SibNet on two standard video captioning benchmarks, YouTube2Text (MSVD) [3] and MSR-VTT [50]. The comparisons with previous results validate that although we only use a basic RNN decoder (LSTM), our SibNet significantly outperforms previous state-of-the-art methods across different evaluation metrics, thanks to the strong encoder that SibNet provides to capture richer and complementary information of video contents. We also analyze the contribution of each component and other design details of SibNet on the overall performance by performing comprehensive ablation studies, and the results further verify that the proposed two-branch architecture possesses unique merits for video captioning.

2 RELATED WORK

Deep learning-based encoder-decoder architecture [4, 9, 17, 18, 21, 25–27, 31, 34, 41, 48, 52–54, 56] has shown its effectiveness in video captioning. Specifically, those methods first adopt an encoder to represent videos into feature vectors and then use a decoder to generate natural language captions.

Although for the task of image captioning, Convolutional Neural Networks (CNNs) [12, 14, 38, 39] have become a standard to encode image content in most state-of-the-art methods [34, 52, 54], for the task of video captioning, how to effectively encode video content is still an open problem. Venugopal et al. [46] proposed to map a sequence of video features to a fixed-length vector with an average pooling layer. Venugopal et al. [44, 45] also presented approaches to mine information from large natural language corpus or utilize temporal information of videos. With the success of attention mechanism in neural video classification [30], neural machine translation [1] and neural image captioning [52, 54]. Yao et al. [53] introduced it into video captioning. Pan et al. [25] proposed a video encoder composed of hierarchical RNN. Gan et al. [9] and Pan et al. [27] improved existing models by detecting manually defined semantic concepts. Comparing with the aforementioned methods, which mostly encode video information in a single flow, our proposed SibNet learns to explicitly and effectively encode the visual content and semantic information of videos using a two-branch architecture.

3 MODEL

In video captioning, the task is to generate a natural language description, a sentence $\mathbf{y}$ for a given input video V. Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ denote the ordered feature vectors of n frames in video V, $\mathbf{X} \in \mathbb{R}^{n \times d}$. Given $\mathbf{X}$ as input, an encoder generates a compact embedded representation $\mathbf{Z}$, which is either a fixed-length vector or a matrix composed of n vectors, to encode the visual information

in $\mathbf{X}$. Then, the decoder decodes the video representation $\mathbf{Z}$ into sentence $\mathbf{y} = [y_1, y_2, \ldots, y_m]$ as a sequence of m words.

Our method follows the encoder-decoder pipeline but proposes a novel Sibling Convolutional Encoder (SibNet) to encode videos.

3.1 Sibling Convolutional Encoder (SibNet)

As shown in Figure 2, SibNet is comprised of two branches, namely the content branch and the semantic branch, which are denoted as CNN_c and CNN_s, respectively. The content branch is designed to encode visual content information, while the semantic branch is designed to encode video semantic information. Unlike existing encoders, whose encoded feature $\mathbf{Z}$ is either a fixed-length vector or a matrix, the representation $\mathbf{Z}$ in SibNet is composed of *two* matrices $\mathbf{Z}^c$ and $\mathbf{Z}^s$. As we see in Figure 2, CNN_c and CNN_s share common properties: firstly, they have the same input $\mathbf{X}$ and the number of their output vectors n are the same. Besides, both branches are formed by a stack of temporal convolutional blocks (TCBs) (will be introduced in Section 3.1.2). Now let us introduce both branches in details.

3.1.1 Content Branch

The role of our content branch is to encode visual content information. Autoencoders have been widely used for unsupervised representation learning. It encodes visual content into a vector and then tries to recover the visual signal from such a vector.

In order to explicitly encode the video content, we propose to implement our content branch with an autoencoder, as shown in Figure 3. As we see, the autoencoder takes $\mathbf{X}$ as input, then passes it through the content branch CNN_c to encode the video into representation $\mathbf{Z}^c$. Our CNN_c is composed of 3 TCBs (will be introduced in Section 3.1.2). After that, CNN_a, which is composed of 3 temporal convolutional layers, reconstructs the original visual content from $\mathbf{Z}^c$. We use $\mathbf{X}^{'}$ to denote the reconstructed content generated by CNN_a. Here, $\mathbf{X}^{'} \in \mathbb{R}^{n \times d}$ is of the same size as $\mathbf{X}$. Euclidean distance between each element of the original sequence $\mathbf{X}$ and the reconstructed sequence $\mathbf{X}^{'}$ is used to measure the content reconstruction loss L_c, which is defined as follows:

$$L_c = \sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{x}_i^{'}||, \tag{1}$$

where $\mathbf{x}_i$ and $\mathbf{x}_i^{'}$ denote the i-th vectors of $\mathbf{X}$ and $\mathbf{X}^{'}$, respectively; $||\cdot||$ is the notation for L2-norm. This unsupervised reconstruction loss of autoencoder is incorporated to the final training loss, which pushes our content branch to play its role.

3.1.2 Temporal Convolutional Block (TCB)

Now we introduce Temporal Convolutional Block (TCB), which is the basic component in both our content and semantic branches. Videos have temporal structures. Therefore, temporal structure modeling of videos is essential for video representation. Instead of using a RNN, we choose to use a simpler temporal modeling architecture, temporal convolutional block (TCB) as shown in Figure 4, which works effectively in our experiments.

As shown in Figure 2, our content and semantic branches both consist of a stack of TCBs. Let $\mathbf{X}_k = [\mathbf{x}_1^k, \mathbf{x}_2^k, \ldots, \mathbf{x}_n^k]$ denote input of the k-th TCB in either branch, where each $\mathbf{x}_i^k$ is a d_k-dimensional vector, $\mathbf{X}_k \in \mathbf{R}^{n \times d_k}$. Firstly, the k-th TCB passes $\mathbf{X}_k$ through TCN, a temporal convolutional layer with kernel size 3. The output of TCN is then passed through a ReLU [24] activation layer. To ease training of our SibNet, we adopt residual connection [12] by adding the output of ReLU activation layer with the original input of the k-th TCB $\mathbf{X}_k$:

$$\mathcal{F}(\mathbf{X}_k) = \text{ReLU}(\mathbf{W}_k * \mathbf{X}_k) \oplus \mathbf{X}_k. \tag{2}$$

Here $\mathcal{F}(\mathbf{X}_k) \in \mathbb{R}^{n \times d_k}$ represents the output of the k-th TCB, $\mathbf{W}_k \in \mathbb{R}^{3 \times d_k}$ denotes learnable parameters of TCN, $*$ and $\oplus$ represent convolutional operator and element-wise addition, respectively. $\mathcal{F}(\mathbf{X}_k)$ then becomes the input of the $(k+1)$-th TCB.

As shown in Figure 2, our content branch is composed of a stack of 3 TCBs , while the semantic branch is composed of a stack of 6 TCBs. In Section 4.3, we will investigate the impact of the TCB numbers in both branches and thus explain why we choose such numbers as above.

3.1.3 Semantic Branch

The task of our semantic branch is to learn a representation of $\mathbf{X}$ that encodes high-level semantics. Inspired by the success of visual-semantic joint embedding in image retrieval [33] and image classification [32], we propose to implement our semantic branch via visual-semantic embedding.

As shown in Figure 5, our visual-sem.antic joint embedding model is composed of two sub-modules, video embedding module and caption embedding module, which map videos and captions into a *common* semantic space. In such space, a video and its corresponding caption should be embedded closely, thus the distance in this space is empowered with semantic meaning. As shown in Figure 5, the video embedding module first maps the input $\mathbf{X}$ to the matrix $\mathbf{Z}^s$ using our semantic branch CNN_s. Then a self-attentive network (SAN) [20] is employed to map $\mathbf{Z}^s$ into a video embedding vector $\mathbf{v}_e$. Instead of averaging all the n vectors in $\mathbf{Z}^s$ as $\mathbf{v}_e$, we use SAN in video embedding module, because it has been proven that the embedding produced by SAN is better at capturing more meaningful information contained in certain frames. Thus, given a sequence $\mathbf{Z}^s = [\mathbf{z}_1^s, \mathbf{z}_2^s, \ldots, \mathbf{z}_n^s]$, SAN embeds it into a vector $\mathbf{v}_e$ by merging all $\mathbf{z}_i^s$ according to their relative importance to the final embedding. Similarly, in the caption embedding module, in order to embed the sentence $\mathbf{y} = [y_1, y_2, \ldots, y_m]$ into a caption embedding vector $\mathbf{c}_e$, it first constructs word vectors $\mathbf{w}_i \in \mathbb{R}^{d_w}$ by [42], and then utilizes another SAN [20] to embed it into vector $\mathbf{c}_e$.

In order to make CNN_s effectively encode semantic information, we follow [34] to utilize bi-directional ranking loss as our semantic training loss. Specifically, we define semantic loss L_s as follows:

$$\begin{aligned} L_s = &\sum_{\mathbf{v}_e} \sum_{\mathbf{c}_e^-} \max(0, m - \mathbf{v}_e \cdot \mathbf{c}_e + \mathbf{v}_e \cdot \mathbf{c}_e^-)) \\ &+ \sum_{\mathbf{c}_e} \sum_{\mathbf{v}_e^-} \max(0, m - \mathbf{c}_e \cdot \mathbf{v}_e + \mathbf{c}_e \cdot \mathbf{v}_e^-)), \end{aligned} \tag{3}$$

where $\cdot$ designates dot product operation. The margin m is set to be 0.1 by cross-validation. Given a video V with embedding vector $\mathbf{v}_e$, $\mathbf{c}_e$ denotes embedding of its ground truth caption, $\mathbf{c}_e^-$ denotes embedding of a negative caption that describes video other than $\mathbf{v}_e$; and vice-versa with $\mathbf{v}_e^-$. This semantic loss, which pushes

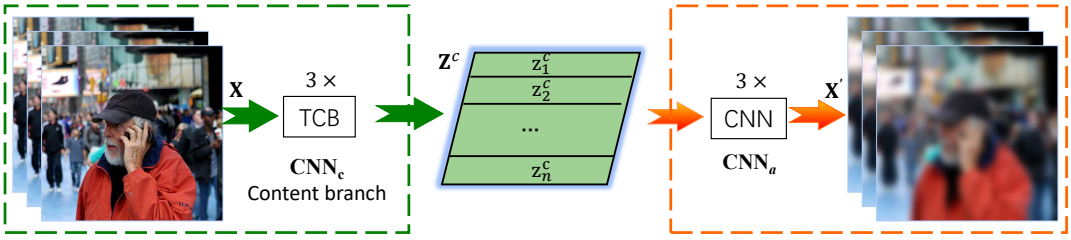


Figure 3: Illustration of the content branch CNN$_c$ implemented via autoencoder. Note that the content loss of autoencoder is

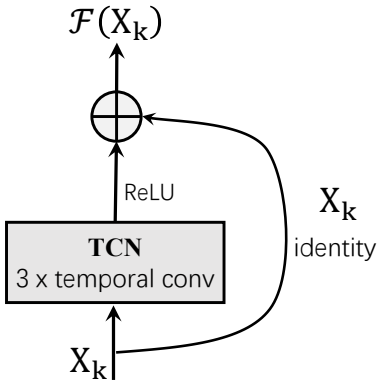


Figure 4: Illustration of our temporal convolutional block (TCB), which is the basic component of both the content branch and the semantic branch.

our semantic branch to play its role, is incorporated into our final training loss.

3.2 Decoder

After we obtain the encoded representation $\mathbf{Z}$, i.e., $\{\mathbf{Z}^c, \mathbf{Z}^s\}$, we follow previous work to use a RNN to decode it into a sentence $\mathbf{y}$. More specifically, given $\mathbf{Z}^c$ and $\mathbf{Z}^s$, the decoder predicts joint probability $p(\mathbf{y})$ of caption $\mathbf{y}$ by sequentially predicting the probability of each word y_i in $\mathbf{y}$. It can be seen from Figure 2 that our decoder is auto-regressive, indicating that it takes the output at all previous time steps as additional inputs. We maximize the probability of generating ground truth captions by minimizing cross-entropy loss. Our decoder loss L_d is defined as follows:

$$L_d = -\log(p(\mathbf{y}|\mathbf{Z}^c, \mathbf{Z}^s)). \tag{4}$$

3.2.1 Soft-attention Mechanism

How to effectively combine $\mathbf{Z}^c$ and $\mathbf{Z}^s$ is the key problem in decoding process. We utilize a soft-attention mechanism. Originally proposed in [1], variants of soft attention have been successfully applied to machine translation [1], image captioning [52] and video captioning [29, 53], etc. Different from standard soft-attention mechanism [1] which returns a fixed-length vector encoding information of one single matrix, our soft-attention mechanism merges visual information of *two* matrices $\mathbf{Z}^c$ and $\mathbf{Z}^s$ in a fixed-length vector. At the i-th decoding time step (when generating the i-th word), our soft-attention mechanism computes the input vector $\mathbf{u}_i$ of the RNN decoder as follows:

$$\mathbf{u}_i = \sum_{j=1}^{n} \text{softmax}_j(\mathbf{s}_i) \cdot \mathbf{z}_j^c \qquad j \in [1, n], \tag{5}$$

where $\text{softmax}_j(\cdot)$ denotes the j-th value of the softmax result vector, $\mathbf{z}_j^c$ is the j-th element of $\mathbf{Z}^c$ that encodes video content information, and $\mathbf{s}_i = [s_{i,1}, s_{i,2}, \ldots, s_{i,n}]$ is defined as follows:

$$s_{i,k} = \mathbf{W_s}^T \tanh(\mathbf{W}_h \mathbf{h}_i + \mathbf{W}_z \mathbf{z}_k^s) \qquad k \in [1, n]. \tag{6}$$

Here, $s_{i,k}$ is a real value; $\mathbf{W_s}, \mathbf{W}_h$ and $\mathbf{W}_z$ are learnable weight matrices; $\mathbf{h}_i$, a fixed-length vector, denotes the hidden state of the RNN decoder at the i-th time step; $\mathbf{z}_k^s$ is the k-th element of $\mathbf{Z}^s$, which encodes video semantic information.

As shown in Equation 5 and 6, the soft-attention mechanism utilizes semantic information in $\mathbf{Z}^s$ to determine a weighting value $\mathbf{s}_i$, which then effectively combine the visual content representation $\mathbf{Z}^c$ to generate a input vector $\mathbf{u}_i$ for RNN decoder. Such soft-attention mechanism is able to ensure our decoder pay more "attention" to the visual content of certain frames if they contain important semantic information. As we can see, by using the proposed soft-attention mechanism, the content and semantic branch in SibNet are effectively combined in a complementary fashion.

3.3 Training

We jointly train all the components of our model, the content branch, the semantic branch, and the RNN decoder in an end-to-end manner. As introduced before, autoencoder and visual-semantic embedding are utilized to impose more fine-grained supervision for both branches of SibNet. Thus, we define the final training loss function by adding three different losses together:

$$L = L_d + \alpha L_c + \beta L_s, \tag{7}$$

where L_d, L_c and L_s denote the decoder loss, content loss and semantic loss, as defined in Equation 4, Equation 1 and Equation 3, respectively; α and β are two scalars that control the influence of content loss and semantic loss during training. We set α and β to be 0.4 and 1 by cross validation.

4 EXPERIMENTS

We test the proposed SibNet on two video captioning benchmarks, YouTube2Text (MSVD) [3] and MSR-VTT [50]. For fair comparison, all the reported results are obtained using Microsoft COCO caption evaluation tool [6]. We utilize Bleu [28], METEOR [7], ROUGE [19]

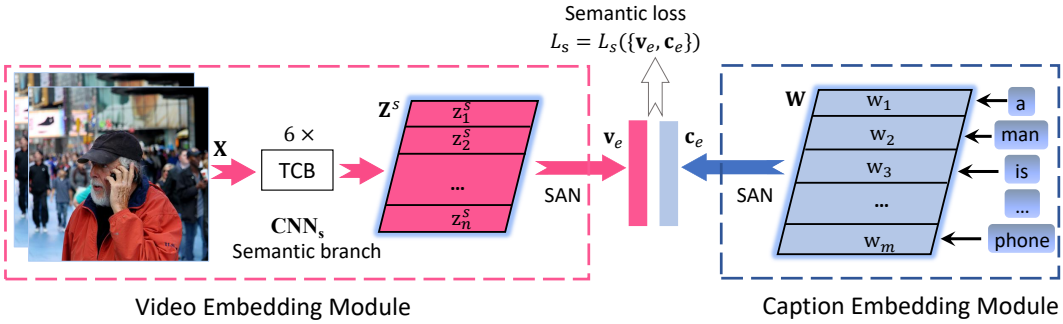


Figure 5: Illustration of the semantic branch $\mathbf{CNN}_s$ implemented via visual-semantic joint embedding. Note that the semantic loss of visual-semantic embedding is one component of our final training loss.

and CIDEr [43] as our evaluation metrics, which are commonly used for performance evaluation of video captioning methods.

4.1 Experiment Setup

Datasets: YouTube2Text is composed of 1970 YouTube videos and 78, 800 captions (40 captions per video, on average) annotated by Amazon Mechanical Turk (AMT) annotators. For fair comparison, we adopt the same evaluation scheme proposed in [45], which used 1200 videos for training, 100 videos for validation and 670 videos for testing. MSR-VTT is a large-scale video captioning dataset, which is comprised of 10, 000 videos and 200, 000 captions (20 unique captions per video). We adopt the standard dataset splits proposed in [50], which used 6513 videos for training, 497 videos for validation and 2990 videos for testing.

For both YouTube2Text and MSR-VTT datasets, we uniformly sample the videos with a sampling rate of 3 frames per second. We then extract visual features using GoogLeNet [38] for YouTube2Text dataset and Inception [39] for MSR-VTT dataset. Both GoogLeNet and Inception are trained by Wang et al. [49]. It is worth noting that most state-of-the-art methods [5, 9, 25–27, 35, 51, 53, 55] take a combination of multiple complementary features, including frame-level CNN features (ResNet [12], Inception [39], GoogleNet [38]), clip-level CNN features (C3D [40]) and audio features (MFCC [22]), as input to their encoders. We do not adopt feature combination in our experiments.

Network Architecture: For the content branch and the semantic branch, we set the output dimension of the TCN, a temporal convolutional layer, in each TCB to be 512. We adopt 1-layer LSTM with 1024-dimensional hidden state as our RNN decoder.[1] Many variants of RNN have been proposed in literature, e.g., GRU. Some state-of-the-art methods have utilized them and have reported better performance. Although we choose a basic LSTM as decoder in our experiments, our method is modular w.r.t. the decoder architecture.

Training Details: We train our model using Adam [16] algorithm with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e-8$. For YouTube2Text dataset, we set the batch size to be 32. The initial learning rates are set to be 8e-5 for the encoder and 4e-5 for the decoder, respectively.

[1] Although SibNet, abbreviation for Sibling Convolutional Encoder, only refers to the encoder of our method, we also use it to refer to a combination of our encoder and the RNN decoder in the following sections.

Table 1: Performance comparisons on YouTube2Text (MSVD) dataset. * indicates that external datasets were used to train these models.

Methods	Bleu-4	METEOR	CIDEr	ROUGE
S2VT [45]	37.0	29.8	-	-
Temporal Attention [53]	41.9	29.6	51.7	-
GRU-RCN [2]	43.3	31.6	68.0	-
aLSTM [11]	44.9	30.4	60.1	-
LSTM-E [26]	45.3	31.0	-	-
HRNE + Attention [25]	46.7	33.9	-	-
p-RNN [55]	49.9	32.6	65.8	-
Latent Topic [5]	48.8	34.4	80.5	-
AF [13]	52.4	32.0	68.8	-
mGRU [57]	49.5	33.4	75.5	-
MA-LSTM [51]	52.3	33.6	70.4	-
RecNet [47]	52.3	34.1	80.3	69.8
GloVe + DeepFussion* [44]	42.1	31.4	-	-
LSTM-YT* [46]	31.2	26.9	-	-
SCN* [9]	50.2	33.4	77.0	-
LSTM-TSA* [27]	52.8	33.5	74.0	-
Ours	**54.2**	**34.8**	**88.2**	**71.7**

For MSR-VTT dataset, we set the batch size to be 64. The initial learning rates are set to be 6e-5 for the encoder and 3e-5 for the decoder. For both datasets, the learning rates are divided by 5 after 10 epochs. We perform gradient clipping with a threshold of 2, and adopt weight initialization method proposed in [10]. We also regularize our model by applying dropout [37] to the output of each TCB with a rate of 0.2. Additional regularization methods, e.g., weight decay, are not utilized.

4.2 Comparison with the State-of-the-Art

On YouTube2Text: In Table 1, we present the results of SibNet and existing methods on YouTube2Text dataset. As we can see, our method achieves the best performance across all metrics, improving Bleu-4 from 52.8 to 54.2, METEOR from 34.4 to 34.8, CIDEr from 80.5 to 88.2 respectively. It is worth noting that large-scale external datasets (at least *two* times larger than YouTube2Text dataset) are

Table 2: Performance comparisons on the test set of MSR-VTT: comparisons with state-of-the-art methods and methods that rank top-4 on the Leaderboard of MSR-VTT Challenge. * indicates that extra training data was used during training. ^{e} indicates that the reported performance was achieved by an ensemble of multiple models. (As WSDC [35] conducts extensive data augmentation, which none of the others conducts, we report the performance of [35] achieved on the *validation* set under similar settings as our method.)

Methods	Bleu-4	METEOR	CIDEr	ROUGE
Rank1: v2t-navigator [15]	40.8	28.2	44.8	**60.9**
Rank2: Aalto [36]	39.8	26.9	45.7	59.8
Rank3: VideoLAB [31]	39.1	27.7	44.1	60.6
Rank4: ruc-uva [8]	38.7	26.9	45.9	58.7
Mean Pooling [46]	30.4	23.7	35.0	52.0
Temporal Attention [53]	28.5	25.0	37.1	53.3
S2VT [45]	31.4	25.7	35.2	55.9
MA-LSTM [51]	36.3	26.3	40.1	59.1
aLSTM [11]	38.0	26.1	-	-
STAT [41]	37.4	26.6	41.5	-
AF [13]	39.4	25.7	40.4	-
RecNet [47]	39.1	26.6	42.7	59.3
M2M^{*e} [29]	40.8	**28.8**	47.1	60.2
WSDC^{v} [35]	39.0	27.7	44.0	60.1
Ours	**40.9**	27.5	**47.5**	60.2

utilized by LSTM-TSA [27], SCN [9], LSTM-YT [46] and GloVe + DeepFussion [44]. Surprisingly, even without using extra training data, our method significantly outperforms all of them. Besides, both LSTM-TSA [27] and SCN [9] rely on hundreds of dataset-specific "semantic attributes", which are manually selected from thousands of candidates. The laborious "semantic attribute" selection prevents [9, 27] to be applied to large dataset with more candidates. On the contrary, our method automatically learns representation of high-level semantics using the proposed semantic branch.

On MSR-VTT: In Table 2, we show a comparison of SibNet and previous state-of-the-art methods on MSR-VTT dataset. We also compare SibNet with methods that occupy top-4 positions of the Leaderboard of MSR-VTT Challenge [23], denoted as Rank1: v2t-navigator [15], Rank2: Aalto [36], Rank3: VideoLAB [31] and Rank4: ruc-uva [8]. Our method achieves the best performance across three of the four metrics. Note that the current best performing method, M2M [29], not only relies on two large-scale external datasets UCF101 and SNLI for training, but also utilizes an ensemble of multiple models. However, our SibNet is trained without extra training data and tested without ensemble.

From Table 1 and Table 2, it can be seen that SibNet consistently outperforms state-of-the-art methods by a large margin even without extra training data and model ensemble, which validates the effectiveness of encoding video contents using the proposed two-branch architecture.

Qualitative Analysis: Figure 6 shows some qualitative results of SibNet. It can be seen that our method can generate captions that correctly describe their corresponding videos. In addition, our method is able to handle challenging situations, such as scene changes. As shown in the second example in the second row, our method generates a correct caption whose subject "`man`", verb "`talk`" and object "`football game`" are extracted from different scene frames.

4.3 Ablation Study

Since SibNet differs from the encoders employed by existing video captioning approaches fundamentally, in this section we perform detailed ablation study to get a better understanding of the proposed model.

4.3.1 How much does each component contribute?

In order to analyze the impact of different components of our proposed model on the performance of video captioning, we evaluate five variants of our model, denoted as: *Single (3-layer)*, *Single (6-layer)*, *Ours (Sib-DL)*, *Ours (CL)* and *Ours (SL)*, respectively. First of all, *Single (3-layer)* and *Single (6-layer)* denote two single-branch encoders which only consist of 3 and 6 identical TCBs. These two variants, which encode visual information using a single branch, could be viewed as the baseline of our model. And both of them are trained using decoder loss L_d alone. To validate the superiority of the proposed two-branch architecture over the baseline, we construct *Ours (Sib-DL)*, which has both the content branch and the semantic branch. But *Ours (Sib-DL)* is also trained with decoder loss L_d alone. To evaluate the effectiveness of our proposed training scheme which provides more fine-grained training supervision, we construct two variants: *Ours (CL)* and *Ours (SL)*. *Ours (CL)* incorporates the autoencoder to impose a content loss L_c as defined in Equation 1 to *Ours (Sib-DL)*. Likewise, *Ours (SL)* incorporates visual-semantic embedding to impose a semantic loss L_s as defined in Equation 3 to *Ours (Sib-DL)*. Lastly, we evaluate *Ours (Full)*, which is our full model.

From Table 3 that shows the results of all variants above on both MSR-VTT and YouTube2Text, we observe that:

(1) Comparing with *Ours (Sib-DL)*, *Single (3-layer)* and *Single (6-layer)* have worse performance. This indicates the necessity of encoding visual information using our proposed two-branch architecture. It is worth noting that the performance of *Single (3-layer)* and *Single (6-layer)* is on a par with many existing methods, which validates the effectiveness of modeling video temporal structures of videos using TCB as described in Section 3.1.2.

(2) By adding content loss L_c to decoder loss L_d used by *Ours (Sib-DL)*, *Ours (CL)* achieves better performance than *Ours (Sib-DL)*. This verifies the efficiency of regularizing the content branch using autoencoder. Similarly, by adding semantic loss L_s, *Ours (SL)* also outperforms *Ours (Sib-DL)* by a large margin. This validates the importance of regularizing the semantic branch by leveraging visual-semantic joint embedding. Finally, we can see that *Ours (Full)* performs slightly better than both *Ours (CL)* and *Ours (SL)*. Hence, we can conclude that our autoencoder and visual-semantic embedding

GT: a man is about to shoot someone in forest

Ours: a man is shooting a gun

GT: a man is swimming in the pool

Ours: a man is swimming in the water

GT: a girl is singing on stage

Ours: a girl is singing on stage

GT: a man is talking about football

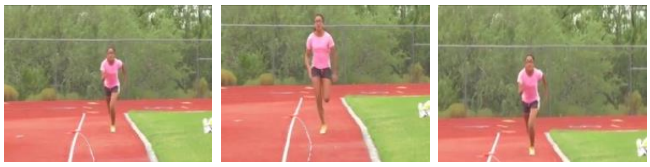
Ours: a man is talking about a football game

GT: a crowd of fireworks

Ours: fireworks are exploding in the sky

GT: a group of people dance on the beach

Ours: a group of people are dancing on the beach



GT: a woman is running in the play ground

Ours: a woman in a red shirt is running on a track

GT: two wrestlers are fighting in the ring

Ours: two men are wrestling in a ring

Figure 6: Qualitative results of our method on MSR-VTT dataset. "GT" denotes ground truth captions; "Ours" denotes captions generated by our method.

Table 3: Performance of different variants of our method on YouTube2Text and MSR-VTT datasets.

Methods	Dataset	L_d	L_c	L_s	Bleu-4	METEOR	CIDEr	ROUGE
Single (3-layer)	MSR-VTT	✓			38.9	26.4	44.8	59.2
Single (6-layer)		✓			39.0	26.8	43.7	59.4
Ours (Sib-DL)		✓			39.4	26.9	45.3	59.6
Ours (CL)		✓	✓		40.0	27.1	46.2	60.1
Ours (SL)		✓		✓	40.4	27.1	46.8	60.0
Ours (Full)		✓	✓	✓	**40.9**	**27.5**	**47.5**	**60.2**
Ours (Sib-DL)	YouTube2Text	✓			51.9	33.1	81.9	69.9
Ours (CL)		✓	✓		52.8	34.0	85.6	71.1
Ours (SL)		✓		✓	53.3	34.5	86.0	71.2
Ours (Full)		✓	✓	✓	**54.2**	**34.8**	**88.2**	**71.7**

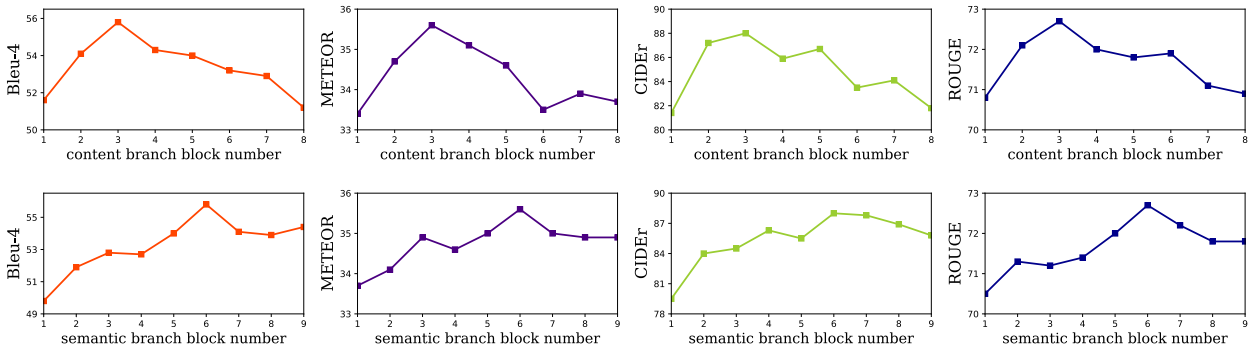


Figure 7: Evaluation of the impact of both branches' depths on the performance of our method. First row: impact of the TCB block number in content branch, where the TCB number in semantic branch is fixed to 6. Second row: impact of the TCB block number in semantic branch, where the TCB number in content branch is fixed to 3.

collaboratively provide *complementary* training guidance to the proposed encoder.

4.3.2 Why semantic branch is deeper than content branch?

In this section, we discuss the impact of the depths of the two branches. We first increase the number of TCB blocks in the content branch from 1 to 8 while the number of blocks in the semantic branch to is fixed to 6. As shown in the first row of Figure 7, the performance drops in a monotonic manner as number of blocks in the content branch goes from 3 to 1 or from 3 to 8. We notice that when the number of blocks is 3, our method can achieve the best performance overall.

We also change the number of blocks in the semantic branch from 1 to 9 while fixing the number of blocks in the content branch as 3. The results are demonstrated in the second row of Figure 7. We can see that consistent performance drop exists when number of blocks in the semantic branch goes from 6 to 1 or from 6 to 9. In particular, using less than 3 blocks in the semantic branch severely affects the performance. This validates that in order to encode semantic information, which has a high level of abstraction, it is better to use deeper semantic branch. Another benefit for stacking more blocks is that, as the number of blocks in our semantic branch goes up, the temporal receptive field of it increases, which enables it to model longer temporal dynamics of videos. Based on the results shown in Figure 7, we empirically choose 3 TCBs to form the content branch and 6 TCBs to form the semantic branch.

4.3.4 Number of parameters

The number of parameters in SibNet and previous state-of-the-art models are reported in Table 4. The reported numbers do not include parameters in the decoder's fully connected layer, whose output is then normalized by softmax function to generate probability distribution of words in the vocabulary. Because the number of parameters in it is proportional to the vocabulary size, it is not reported in most previous work. It can be seen from Table 4 that SibNet has much smaller number of parameters than previous state-of-the-art approaches (44% of [45], 57% of [9], 77% of [29] and 90% of [27]). It is worth noting that the number of parameters in our encoder ($2.3M$) is less than 25% of that of the RNN decoder ($9.2M$).

Table 4: The number of parameters of SibNet and previous state-of-the-art models.

Methods	Parameters
S2VT [45]	26.4M
SCN [9]	20.1M
M2M [29]	14.9M
LSTM-TSA [27]	12.8M
Ours	11.5M
- CNN_s	1.5M
- CNN_c	0.8M
- RNN_d	9.2M

Our encoder is able to achieve greater representation power with far less number of parameters than existing encoders employed by previous methods.

5 CONCLUSIONS

In this paper, we propose SibNet, which encodes rich video information using a two-branch architecture. The content branch learns video representation using visual information with an autoencoder, while the semantic branch learns video semantic-specific representation with visual-semantic joint embedding. To jointly optimize the encoder and decoder, we propose a new loss function that includes the content loss, semantic loss, and decoder loss. Extensive experiments conducted on standard video captioning benchmarks show that by jointly optimizing the proposed loss function, our SibNet can encode better video representations thus achieving better video captioning results. The comparisons with existing results validate that SibNet outperforms previous state-of-the-art models by a large margin.

6 ACKNOWLEDGEMENT

This work is supported in part by start-up funds from State University of New York at Buffalo and gift grant from Snap.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[2] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. 2015. Delving deeper into convolutional networks for learning video representations. In *Proceedings of International Conference on Learning Representations (ICLR)*.

[3] David L Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*. ACL, 190–200.

[4] Fuhai Chen, Rongrong Ji, Jinsong Su, Yongjian Wu, and Yunsheng Wu. 2017. StructCap: Structured Semantic Embedding for Image Captioning. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 46–54.

[5] Shizhe Chen, Jia Chen, Qin Jin, and Alexander Hauptmann. 2017. Video captioning with guidance of multimodal latent topics. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1838–1846.

[6] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325* (2015).

[7] John R Crouse, Joel S Raichlen, Ward A Riley, Gregory W Evans, Mike K Palmer, Daniel H O'Leary, Diederick E Grobbee, Michiel L Bots, METEOR Study Group, et al. 2007. Effect of rosuvastatin on progression of carotid intima-media thickness in low-risk individuals with subclinical atherosclerosis: the METEOR Trial. *Jama* 297, 12 (2007), 1344–1353.

[8] Jianfeng Dong, Xirong Li, Weiyu Lan, Yujia Huo, and Cees GM Snoek. 2016. Early embedding and late reranking for video captioning. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 1082–1086.

[9] Zhe Gan, Chuang Gan, Xiaodong He, Yunchen Pu, Kenneth Tran, Jianfeng Gao, Lawrence Carin, and Li Deng. 2017. Semantic compositional networks for visual captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[10] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 249–256.

[11] Zhao Guo, Lianli Gao, Jingkuan Song, Xing Xu, Jie Shao, and Heng Tao Shen. 2016. Attention-based LSTM with semantic consistency for videos captioning. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 357–361.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

[13] Chiori Hori, Takaaki Hori, and Teng-Yok Lee. 2017. Attention-based multimodal fusion for video description. (2017).

[14] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[15] Qin Jin, Jia Chen, Shizhe Chen, Yifan Xiong, and Alexander Hauptmann. 2016. Describing videos using multi-modal fusion. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 1087–1091.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)* (2014).

[17] Weiyu Lan, Xirong Li, and Jianfeng Dong. 2017. Fluency-Guided Cross-Lingual Image Captioning. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1549–1557.

[18] Guang Li, Shubo Ma, and Yahong Han. [n. d.]. Summarization-based video caption via deep neural networks. In *Proceedings of the 2015 ACM on Multimedia Conference, pages=1191–1194, year=2015, organization=ACM*.

[19] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* (2004).

[20] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *International Conference on Learning Representations (ICLR)*.

[21] Yuan Liu and Zhongchao Shi. 2016. Boosting video description generation by explicitly translating from frame-level captions. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 631–634.

[22] Beth Logan et al. 2000. Mel frequency cepstral coefficients for music modeling. In *ISMIR*, Vol. 270. 1–11.

[23] Tao Mei, Yong Rui, Xinmei Tian, and Ting Yao. 2017. MSR-VTT Challenge. http://ms-multimedia-challenge.com/2017/challenge. (2017).

[24] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*. 807–814.

[25] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. 2016. Hierarchical recurrent neural encoder for video representation with application to captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1029–1038.

[26] Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. 2016. Jointly modeling embedding and translation to bridge video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[27] Yingwei Pan, Ting Yao, Houqiang Li, and Tao Mei. 2017. Video captioning with transferred semantic attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[28] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting on Association for Computational Linguistics (ACL)*. ACL, 311–318.

[29] Ramakanth Pasunuru and Mohit Bansal. 2017. Multi-task video captioning with video and entailment generation. In *Proceedings of Association for Computational Linguistics (ACL)*. ACL.

[30] Yuxin Peng, Yunzhen Zhao, and Junchao Zhang. 2018. Two-stream Collaborative Learning with Spatial-Temporal Attention for Video Classification. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)* (2018).

[31] Vasili Ramanishka, Abir Das, Dong Huk Park, Subhashini Venugopalan, Lisa Anne Hendricks, Marcus Rohrbach, and Kate Saenko. 2016. Multimodal video description. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 1092–1096.

[32] Zhou Ren, Hailin Jin, Zhe Lin, Chen Fang, and Alan Yuille. 2016. Joint image-text representation by gaussian visual-semantic embedding. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 207–211.

[33] Zhou Ren, Hailin Jin, Zhe Lin, Chen Fang, and Alan Yuille. 2017. Multiple Instance Visual-Semantic Embedding. In *Proceeding of the British Machine Vision Conference (BMVC)*.

[34] Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. 2017. Deep Reinforcement Learning-based Image Captioning with Embedding Reward. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[35] Zhiqiang Shen, Jianguo Li, Zhou Su, Minjun Li, Yurong Chen, Yu-Gang Jiang, and Xiangyang Xue. 2017. Weakly supervised dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[36] Rakshith Shetty and Jorma Laaksonen. 2016. Frame-and segment-level features and candidate pool evaluation for video caption generation. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 1073–1076.

[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*. 1–9.

[39] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2818–2826.

[40] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 4489–4497.

[41] Yunbin Tu, Xishan Zhang, Bingtao Liu, and Chenggang Yan. 2017. Video Description with Spatial-Temporal Attention. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1014–1022.

[42] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of Association for Computational Linguistics (ACL)*. ACL, 384–394.

[43] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4566–4575.

[44] Subhashini Venugopalan, Lisa Anne Hendricks, Raymond Mooney, and Kate Saenko. 2016. Improving lstm-based video description with linguistic knowledge mined from text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL.

[45] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2015. Sequence to sequence-video to text. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 4534–4542.

[46] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. 2014. Translating videos to natural language using deep recurrent neural networks. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

[47] Bairui Wang, Lin Ma, Wei Zhang, and Wei Liu. 2018. Reconstruction Network for Video Captioning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[48] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. 2016. Image captioning with deep bidirectional LSTMs. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 988–997.

[49] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision (ECCV)*. Springer, 20–36.

[50] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (ICCV)*. 5288–5296.

[51] Jun Xu, Ting Yao, Yongdong Zhang, and Tao Mei. 2017. Learning Multimodal Attention LSTM Networks for Video Captioning. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 537–545.

[52] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML)*. 2048–2057.

[53] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. 2015. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 4507–4515.

[54] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4651–4659.

[55] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. 2016. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4584–4593.

[56] Youngjae Yu, Hyungjin Ko, Jongwook Choi, and Gunhee Kim. 2017. End-to-end concept word detection for video captioning, retrieval, and question answering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[57] Linchao Zhu, Zhongwen Xu, and Yi Yang. 2017. Bidirectional multirate reconstruction for temporal modeling in videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.