End-to-end Concept Word Detection for Video Captioning, Retrieval, and Question Answering

Youngjae Yu Hyungjin Ko Jongwook Choi Gunhee Kim
Seoul National University, Seoul, Korea
{yj.yu, hj.ko}@vision.snu.ac.kr, {wookayin, gunhee}@snu.ac.kr
http://vision.snu.ac.kr/project/lsmdc-2016

Abstract

We propose a high-level concept word detector that can be integrated with any video-to-language models. It takes a video as input and generates a list of concept words as useful semantic priors for language generation models. The proposed word detector has two important properties. First, it does not require any external knowledge sources for training. Second, the proposed word detector is trainable in an end-to-end manner jointly with any video-to-language models. To effectively exploit the detected words, we also develop a semantic attention mechanism that selectively focuses on the detected concept words and fuse them with the word encoding and decoding in the language model. In order to demonstrate that the proposed approach indeed improves the performance of multiple video-to-language tasks, we participate in all the four tasks of LSMDC 2016 [18]. Our approach has won three of them, including fill-in-the-blank, multiple-choice test, and movie retrieval.

1. Introduction

Video-to-language tasks, including video captioning [6, 8, 17, 27, 32, 35] and video question answering (QA) [23], are recent emerging challenges in computer vision research. This set of problems is interesting as one of frontiers in artificial intelligence; beyond that, it can also potentiate multiple practical applications, such as retrieving video content by users' free-form queries or helping visually impaired people understand the visual content. Recently, a number of large-scale datasets have been introduced as a common ground for researchers to promote the progress of video-to-language research (*e.g.* [4, 16, 18, 23]).

The objective of this work is to propose a *concept word detector*, as shown in Fig.1, which takes a training set of videos and associated sentences as input, and generates a list of high-level concept words per video as useful semantic priors for a variety of video-to-language tasks, including video captioning, retrieval, and question answering. We

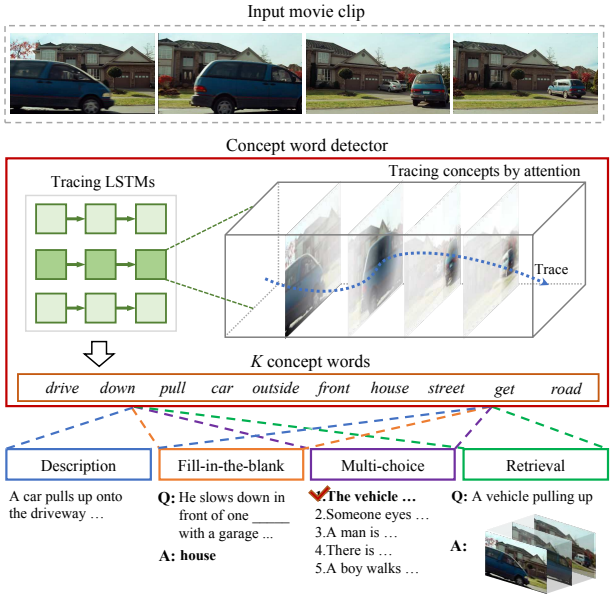

Figure 1. The intuition of the proposed concept word detector. Given a video clip, a set of tracing LSTMs extract multiple concept words that consistently appear across frame regions. We then employ semantic attention to combine the detected concepts with text encoding/decoding for several video-to-language tasks of LSMDC 2016, such as captioning, retrieval, and question answering.

design our word detector to have the following two characteristics, to be easily integrated with any video-to-language models. First, it does not require any external knowledge sources for training. Instead, our detector learns the correlation between words in the captions and video regions from the whole training data. To this end, we use a continuous soft attention mechanism that traces consistent visual information across frames and associates them with concept words from captions. Second, the word detector is trainable in an end-to-end manner jointly with any video-to-language models. The loss function for learning the word detector can be plugged as an auxiliary term into the model's overall cost function; as a result, we can reduce efforts to separately

collect training examples and learn both models.

We also develop language model components to to effectively exploit the detected words. Inspired by *semantic attention* in image captioning research [34], we develop an attention mechanism that selectively focuses on the detected concept words and fuse them with word encoding and decoding in the language model. That is, the detected concept words are combined with input words to better represent the hidden states of encoders, and with output words to generate more accurate word prediction.

In order to demonstrate that the proposed word detector and attention mechanism indeed improve the performance of multiple video-to-language tasks, we participate in four tasks of LSMDC 2016 (*Large Scale Movie Description Challenge*) [18], which is one of the most active and successful benchmarks that advance the progress of video-to-language research. The challenges include *movie description* and *multiple-choice test* as video captioning, *fill-in-the-blank* as video question answering, and *movie retrieval* as video retrieval. Following the public evaluation protocol of LSMDC 2016, our approach achieves the best accuracies in the three tasks (*fill-in-the-blank*, *multiple-choice test*, and *movie retrieval*), and comparable performance in the other task (*movie description*).

1.1. Related Work

Our work can be uniquely positioned in the context of two recent research directions in image/video captioning.

Image/Video Captioning with Word Detection. Image and video captioning has been actively studied in recent vision and language research, including [5, 6, 8, 17, 19, 27, 28], to name a few. Among them, there have been several attempts to detect a set of concept words or attributes from visual input to boost up the captioning performance. In image captioning research, Fang *et al*. [7] exploit a multiple instance learning (MIL) approach to train visual detectors that identify a set of words with bounding boxed regions of the image. Based on the detected words, they retrieve and rerank the best caption sentence for the image. Wu *et al*. [29] use a CNN to learn a mapping between an image and semantic attributes. They then exploit the mapping as an input to the captioning decoder. They also extend the framework to explicitly leverage external knowledge base such as DBpedia for question answering tasks. Venugopalan *et al*. [26] generate description with novel words beyond the ones in the training set, by leveraging external sources, including object recognition datasets like ImageNet and external text corpus like Wikipedia. You *et al*. [34] also exploit weak labels and tags on Internet images to train additional parametric visual classifiers for image captioning.

In the video domain, it is more ambiguous to learn the relation between descriptive words and visual patterns. There have been only few work in video captioning. Rohrbach *et al*. [17] propose a two-step approach for video captioning on the LSMDC dataset. They first extract verbs, objects, and places from movie description, and separately train SVM-based classifiers for each group. They then learn the LSTM decoder that generates text description based on the responses of these visual classifiers.

While almost all previous captioning methods exploit external classifiers for concept or attribute detection, the novelty of our work lies in that we use only captioning training data with no external sources to learn the word detector, and propose an end-to-end design for learning both word detection and caption generation simultaneously. Moreover, compared to video captioning work of [17] where only *movie description* of LSMDC is addressed, this work is more comprehensive in that we validate the usefulness of our method for all the four tasks of LSMDC.

Attention for Captioning. Attention mechanism has been successfully applied to caption generation. One of the earliest works is [31] that dynamically focuses on different image regions to produce an output word sequence. Later this soft attention has been extended as temporal attention over video frames [33, 35] for video captioning.

Beyond the attention on spatial or temporal structure of visual input, recently You *et al*. [34] propose an attention on attribute words for image captioning. That is, the method enumerates a set of important object labels in the image, and then dynamically switch attention among these concept labels. Although our approach also exploits the idea of semantic attention, it bears two key differences. First, we extend the semantic attention to video domains for the first time, not only for video captioning but also for retrieval and question answering tasks. Second, the approach of [34] relies on the classifiers that are separately learned from external datasets, whereas our approach is learnable end-to-end with only training data of captioning. It significantly reduces efforts to prepare for additional multi-label classifiers.

1.2. Contributions

We summarize the contributions of this work as follows.

(1) We propose a novel end-to-end learning approach for detecting a list of concept words and attend on them to enhance the performance of multiple video-to-language tasks. The proposed concept word detection and attention model can be plugged into any models of video captioning, retrieval, and question answering. Our technical novelties can be seen from two recent trends of image/video captioning research. First, our work is a first end-to-end trainable model not only for concept word detection but also for language generation. Second, our work is a first semantic attention model for video-to-language tasks.

(2) To validate the applicability of the proposed approach, we participate in all the four tasks of LSMDC 2016.

Our models have won three of them, including *fill-in-the-blank*, *multiple-choice test*, and *movie retrieval*. We also attain comparable performance for *movie description*.

2. Detection of Concept Words from Videos

We first explain the pre-processing steps for representation of words and video frames. Then, we explain how we detect concept words for a given video.

2.1. Preprocessing

Dictionary and Word Embedding. We define a vocabulary dictionary $\mathcal{V}$ by collecting the words that occur more than three times in the dataset. The dictionary size is $|\mathcal{V}| = 12\,486$, from which our models sequentially select words as output. We train the word2vec skip-gram embedding [14] to obtain the word embedding matrix $\mathbf{E} \in \mathsf{R}^{d \times |\mathcal{V}|}$ where d is the word embedding dimension and V is the dictionary size. We set $d = 300$ in our implementation.

Video Representation. We first equidistantly sample one per ten frames from a video, to reduce the frame redundancy while minimizing loss of information. We denote the number of video frames by N. We limit the maximum number of frames to be $N_{max} = 40$; if a video is too long, we use a wider interval for uniform sampling.

We employ a convolutional neural network (CNN) to encode video input. Specifically, we extract the feature map of each frame from the res5c layer (*i.e.* $\mathsf{R}^{7\times7\times2,048}$) of ResNet [9] pretrained on ImageNet dataset [20], and then apply a 2×2 max-pooling followed by a 3×3 convolution to reduce dimension to $\mathsf{R}^{4\times4\times500}$. Reducing the number of spatial grid regions to 4×4 helps the concept word detector get trained much faster, while not hurting detection performance significantly. We denote resulting visual features of frames by $\{\mathbf{v}_n\}_{n=1}^{N}$. Throughout this paper, we use n for denoting video frame index.

2.2. An Attention Model for Concept Detection

Concept Words and Traces. We propose the *concept word detector* using LSTM networks with soft attention mechanism. Its structure is shown in the red box of Fig.2. Its goal is, for a given video, to discover a list of *concept words* that consistently appear across frame regions. The detected concept words are used as additional references for video captioning models (section 3.1), which generates output sentence by selectively attending on those words.

We first define a set of candidate words with a size of V from all training captions. Among them, we discover K concept words per video. We set $V = 2,000$ and $K = 10$. We first apply the automatic POS tagging of NLTK [3], to extract nouns, verbs and adjectives from all training caption sentences [7]. We then compute the frequencies of those words in a training set, and select the V most common words as concept word candidates.

Since we do not have groundtruth bounding boxes for concept words in the videos, we cannot train individual concept detectors in a standard supervised setting. Our idea is to adopt a soft attention mechanism to infer words by tracking regions that are spatially consistent. To this end, we employ a set of *tracing LSTMs*, each of which takes care of a single spatially-consistent meaning being tracked over time, what we call *trace*. That is, we keep track of spatial attention over video frames using an LSTM, so that spatial attentions in adjacent frames resemble the spatial consistency of a single concept (*e.g.* a moving object, or an action in video clips; see Fig.1). We use a total of L tracing LSTMs to capture out L traces (or concepts), where L is the number of spatial regions in the visual feature (*i.e.* $L = 4 \times 4 = 16$ for $\mathbf{v} \in \mathsf{R}^{4\times4\times D}$). Fusing these L concepts together, we finally discover K concept words, as will be described next.

Computation of Spatial Attention. For each trace l, we maintain spatial attention weights $\boldsymbol{\alpha}_n^{(l)} \in \mathsf{R}^{4\times4}$, indicating where to attend on (4×4) spatial grid locations of $\mathbf{v}_n$, through video frames $n = 1 \dots N$. The initial attention weight $\boldsymbol{\alpha}_0^{(l)}$ at $n = 0$ is initialized with an one-hot matrix, for each of L grid locations. We compute the hidden states $\mathbf{h}_n^{(l)} \in \mathsf{R}^{500}$ of the LSTM through $n = 1 \dots N$ by:

$$\mathbf{c}_n^{(l)} = \boldsymbol{\alpha}_n^{(l)} \otimes \mathbf{v}_n \tag{1}$$

$$\mathbf{h}_n^{(l)} = \text{LSTM}(\mathbf{c}_n^{(l)}, \mathbf{h}_{n-1}^{(l)}). \tag{2}$$

where $A \otimes B = \sum_{j,k} A_{(j,k)} \cdot B_{(j,k,:)}$. The input to LSTMs is the context vector $\mathbf{c}_n^{(l)} \in \mathsf{R}^{500}$, which is obtained by applying spatial attention $\boldsymbol{\alpha}_n^{(l)}$ to the visual feature $\mathbf{v}_n$. Note that the parameters of L LSTMs are shared.

The attention weight vector $\boldsymbol{\alpha}_n^{(l)} \in \mathsf{R}^{4\times4}$ at time step n is updated as follows:

$$\mathbf{e}_n^{(l)}(j,k) = \mathbf{v}_n(j,k) \odot \mathbf{h}_{n-1}^{(l)}, \tag{3}$$

$$\boldsymbol{\alpha}_n^{(l)} = \text{softmax}\left(\text{Conv}(\mathbf{e}_n^{(l)})\right), \tag{4}$$

where $\odot$ is elementwise product, and $\text{Conv}(\cdot)$ denotes two convolution operations before the softmax layer in Fig.2. Note that $\boldsymbol{\alpha}_n^{(l)}$ in Eq.(3) is computed from the previous hidden state $\mathbf{h}_{n-1}^{(l)}$ of the LSTM.

The spatial attention $\boldsymbol{\alpha}_n^{(l)}$ measures how each spatial grid location of visual features is related to the concept being tracked through tracing LSTMs. By repeating these two steps of Eq.(1)–(3) from $n = 1$ to N, our model can continuously find important and temporally consistent meanings over time, that are closely related to a part of video, rather than focusing on each video frame individually.

Finally, we predict the concept confidence vector $\mathbf{p}$:

$$\mathbf{p} = \sigma\left(\mathbf{W}_p\left[\mathbf{h}_N^{(1)}; \cdots; \mathbf{h}_N^{(L)}\right] + \mathbf{b}_p\right) \in \mathsf{R}^V, \tag{5}$$

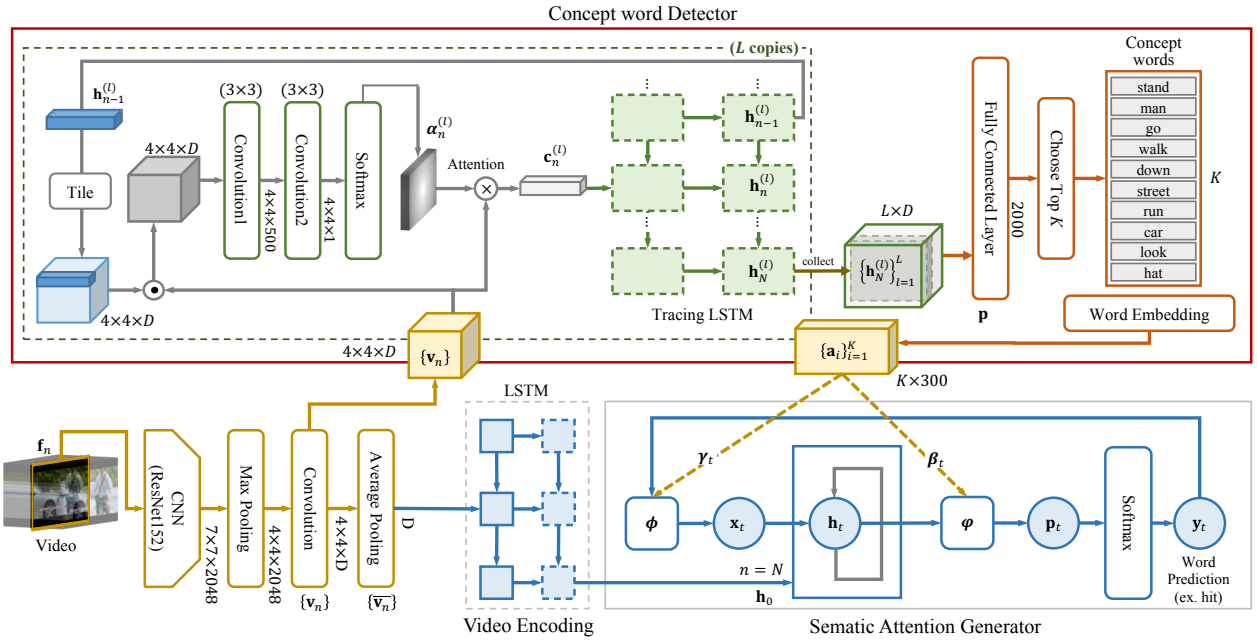

Figure 2. The architecture of the concept word detection in a top red box (section 2.2), and our video description model in bottom, which uses semantic attention on the detected concept words (section 3.1).

that is, we first concatenate the hidden states $\{\mathbf{h}_N^{(l)}\}_{l=1}^L$ at the last time step of all tracing LSTMs, apply a linear transform parameterized by $\mathbf{W}_p \in \mathsf{R}^{V\times(500L)}$ and $\mathbf{b}_p \in \mathsf{R}^V$, and apply the elementwise sigmoid activation σ.

Training and Inference. For training, we obtain a reference concept confidence vector $\mathbf{p}^* \in \mathsf{R}^V$ whose element p_i^* is 1 if the corresponding word exists in the groundtruth caption; otherwise, 0. We minimize the following sigmoid cross-entropy cost $\mathcal{L}_{con}$, which is often used for multi-label classification [30] where each class is independent and not mutually exclusive:

$$\mathcal{L}_{con} = -\frac{1}{V}\sum_{i=1}^{V}\left[p_i^*\log(p_i) + (1-p_i^*)\log(1-p_i)\right]. \quad (6)$$

Strictly speaking, since we apply an end-to-end learning approach, the cost of Eq.(6) is used as an auxiliary term for the overall cost function, which will be discussed in section 3.

For inference, we compute $\mathbf{p}$ for a given query video, and find top K words from the score $\mathbf{p}$ (*i.e.* $\text{argmax}_{1:K}\,\mathbf{p}$). Finally, we represent these K concept words by their word embedding $\{\mathbf{a}_i\}_{i=1}^K$.

3. Video-to-Language Models

We design a different base model for each of LSMDC tasks, while they share the concept word detector and the semantic attention mechanism. That is, we aim to validate that the proposed concept word detection is useful to a wide range of video-to-language models. For base models, we take advantage of state-of-the-art techniques, for which we do not argue as our contribution. We refer to our video-to-language models leveraging the concept word detector as *CT-SAN* (*Concept-Tracing Semantic Attention Network*).

For better understanding of our models, we outline the four LSMDC tasks as follows: (i) *Movie description*: generating a single descriptive sentence for a given movie clip, (ii) *Fill-in-the-blank*: given a video and a sentence with a single blank, finding a suitable word for the blank from the whole vocabulary set, (iii) *Multiple-choice test*: given a video query and five descriptive sentences, choosing the correct one out of them, and (iv) *Movie retrieval*: ranking 1,000 movie clips for a given natural language query.

We defer more model details to the supplementary file. Especially, we skip the description of multiple-choice and movie retrieval models in Figure 3(b)–(c), which can be found in the supplementary file.

3.1. A Model for Description

Fig.2 shows the proposed video captioning model. It takes video features $\{\mathbf{v}_n\}_{n=1}^N$ and the detected concept words $\{\mathbf{a}_i\}_{i=1}^K$ as input, and produces a word sequence as output $\{\mathbf{y}_t\}_{t=1}^T$. The model comprises video encoding and caption decoding LSTMs, and two semantic attention models. The two LSTM networks have two layers in depth, with layer normalization [1] and dropout [22] with a rate of 0.2.

Video Encoder. The *video encoding LSTM* encodes a video into a sequence of hidden states $\{\mathbf{s}_n\}_{n=1}^N \in \mathsf{R}^D$.

$$\mathbf{s}_n = \text{LSTM}(\overline{\mathbf{v}_n}, \mathbf{s}_{n-1}) \quad (7)$$

where $\overline{\mathbf{v}_n} \in \mathsf{R}^D$ is obtained by $(4,4)$-average-pooling $\mathbf{v}_n$.

Caption Decoder. The *caption decoding LSTM* is a normal LSTM network as follows:

$$\mathbf{h}_t = \text{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}), \tag{8}$$

where the input $\mathbf{x}_t$ is an intermediate representation of t-th word input with semantic attention applied, as will be described below. We initialize the hidden state at $t = 0$ by the last hidden state of the video encoder: $\mathbf{h}_0 = \mathbf{s}_N \in \mathsf{R}^D$.

Semantic Attention. Based on [34], our model in Fig.2 uses the semantic attention in two different parts, which are called as *input* and *output* semantic attention, respectively.

The *input semantic attention* ϕ computes an attention weight $\gamma_{t,i}$, which is assigned to each predicted concept word $\mathbf{a}_i$. It helps the caption decoding LSTM focus on different concept words dynamically at each step t.

The attention weight $\gamma_{t,i} \in \mathsf{R}^K$ and input vector $\mathbf{x}_t \in \mathsf{R}^D$ to the LSTM are obtained by

$$\gamma_{t,i} \propto \exp((\mathbf{E}\mathbf{y}_{t-1})^\top \mathbf{W}_\gamma \mathbf{a}_i), \tag{9}$$

$$\begin{aligned}\mathbf{x}_t &= \phi(\mathbf{y}_{t-1}, \{\mathbf{a}_i\}) \\ &= \mathbf{W}_x(\mathbf{E}\mathbf{y}_{t-1} + \text{diag}(\mathbf{w}_{x,a}) \sum_i \gamma_{t,i}\mathbf{a}_i).\end{aligned} \tag{10}$$

We multiply a previous word $\mathbf{y}_{t-1} \in \mathsf{R}^{|\mathcal{V}|}$ by the word embedding matrix $\mathbf{E}$ to be d-dimensional. The parameters to learn include $\mathbf{W}_\gamma \in \mathsf{R}^{d\times d}$, $\mathbf{W}_x \in \mathsf{R}^{D\times d}$ and $\mathbf{w}_{x,a} \in \mathsf{R}^d$.

The *output semantic attention* φ guides how to dynamically weight the concept words $\{\mathbf{a}_i\}$ when generating an output word $\mathbf{y}_t$ at each step. We use $\mathbf{h}_t$, the hidden state of decoding LSTM at t as an input to the output attention function φ. We then compute $\mathbf{p}_t \in \mathsf{R}^D$ by attending the concept words set $\{\mathbf{a}_i\}$ with the weight $\beta_{t,i}$:

$$\beta_{t,i} \propto \exp(\mathbf{h}_t^\top \mathbf{W}_\beta \sigma(\mathbf{a}_i)), \tag{11}$$

$$\begin{aligned}\mathbf{p}_t &= \varphi(\mathbf{h}_t, \{\mathbf{a}_i\}) \\ &= \mathbf{h}_t + \text{diag}(\mathbf{w}_{h,a}) \sum_i \beta_{t,i}\mathbf{W}_\beta \sigma(\mathbf{a}_i),\end{aligned} \tag{12}$$

where σ is the hyperbolic tangent, and parameters include $\mathbf{w}_{h,a} \in \mathsf{R}^D$ and $\mathbf{W}_\beta \in \mathsf{R}^{D\times d}$.

Finally, the probability of output word is obtained as

$$p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}) = \text{softmax}(\mathbf{W}_y \mathbf{p}_t + \mathbf{b}_y), \tag{13}$$

where $\mathbf{W}_y \in \mathsf{R}^{|\mathcal{V}|\times D}$ and $\mathbf{b}_y \in \mathsf{R}^{|\mathcal{V}|}$. This procedure loops until $\mathbf{y}_t$ corresponds to the <EOS> token.

Training. To learn the parameters of the model, we define a loss function as the total negative log-likelihood of all the words, with regularization terms on attention weights $\{\alpha_{t,i}\}$, $\{\beta_{t,i}\}$, and $\{\gamma_{t,i}\}$ [34], as well as the loss $\mathcal{L}_{con}$ for concept discovery (Eq.6):

$$\mathcal{L} = -\sum_t \log p(\mathbf{y}_t) + \lambda_1(g(\beta) + g(\gamma)) + \lambda_2 \mathcal{L}_{con} \tag{14}$$

where λ_1, λ_2 are hyperparameters and g is a regularization function with setting to $p = 2, q = 0.5$ as

$$\begin{aligned}g(\alpha) &= \|\alpha\|_{1,p} + \|\alpha^\top\|_{1,q} \\ &= \left[\sum_i \left[\sum_t \alpha_{t,i}\right]^p\right]^{1/p} + \left[\sum_t \left[\sum_i \alpha_{t,i}\right]^q\right]^{1/q}.\end{aligned} \tag{15}$$

For the rest of models, we transfer the parameters of the concept word detector trained with the description model, and allow the parameters being fine-tuned.

3.2. A Model for Fill-in-the-Blank

Fig.3(a) illustrates the proposed model for the fill-in-the-blank task. It is based on a *bidirectional LSTM network* (BLSTM) [21, 10], which is useful in predicting a blank word from an imperfect sentence, since it considers the sequence in both forward and backward directions. Our key idea is to employ the semantic attention mechanism on both input and output of the BLSTM, to strengthen the meaning of input and output words with the detected concept words.

The model takes word representation $\{\mathbf{c}_t\}_{t=1}^T$ and concept words $\{\mathbf{a}_i\}_{i=1}^K$ as input. Each $\mathbf{c}_t \in \mathsf{R}^d$ is obtained by multiplying the one-hot word vector by an embedding matrix $\mathbf{E}$. Suppose that the t-th text input is a blank for which we use a special token <blank>. We add the word prediction module only on top of the t-th step of the BLSTM.

BLSTM. The input video is represented by the *video encoding LSTM* in Figure 2. The hidden state of the final video frame $\mathbf{s}_N$ is used to initialize the hidden states of the BLSTM: $\mathbf{h}^b_{T+1} = \mathbf{h}^f_0 = \mathbf{s}_N$, where $\{\mathbf{h}^f_t\}_{t=1}^T$ and $\{\mathbf{h}^b_t\}_{t=1}^T$ are the forward and backward hidden states of the BLSTM, respectively:

$$\mathbf{h}^f_t = \text{LSTM}(\mathbf{x}_t, \mathbf{h}^f_{t-1}), \tag{16}$$

$$\mathbf{h}^b_t = \text{LSTM}(\mathbf{x}_t, \mathbf{h}^b_{t+1}). \tag{17}$$

We also use the layer normalization [1].

Semantic Attention. The input and output semantic attention of this model is almost identical to those of the captioning model in section 3.1, only except that the word representation $\mathbf{c}_t \in \mathsf{R}^d$ is used as input at each time step, instead of previous word vector $\mathbf{y}_{t-1}$. Then the attention weighted word vector $\{\mathbf{x}_t\}_{t=1}^T$ is fed into the BLSTM.

The output semantic attention is also similar to that of the captioning model in section 3.1, only except that we apply the attention only once at t-th step where the <blank> token is taken as input. We feed the output of the BLSTM

$$\mathbf{o}_t = \tanh(\mathbf{W}_o[\mathbf{h}^f_t; \mathbf{h}^b_t] + \mathbf{b}_o), \tag{18}$$

where $\mathbf{W}_o \in \mathsf{R}^{D\times 2D}$ and $\mathbf{b}_o \in \mathsf{R}^D$, into the output attention function φ, which generates $\mathbf{p} \in \mathsf{R}^D$ as in Eq.(12) of the description model, $\mathbf{p} = \varphi(\mathbf{o}_t, \{\mathbf{a}_i\})$.

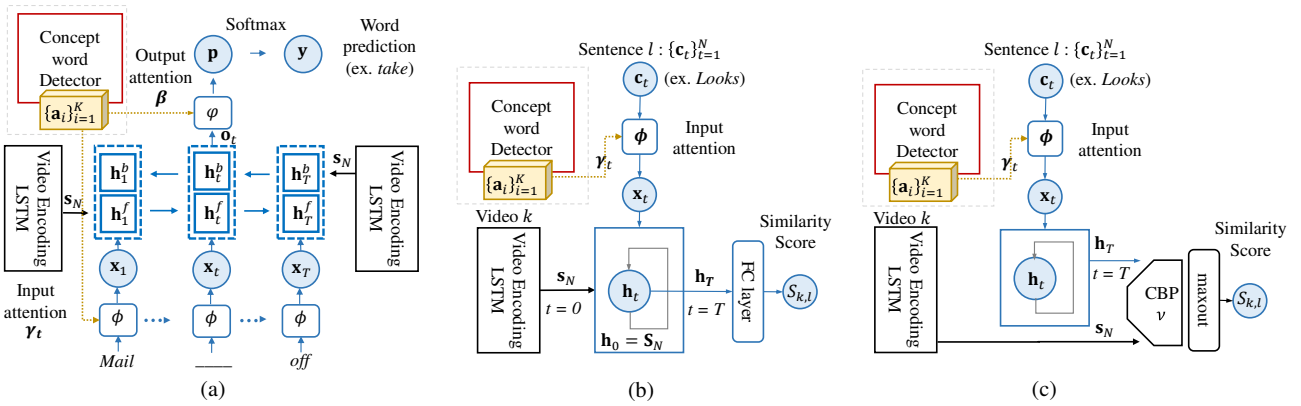

Figure 3. The model architectures for (a) fill-in-the-blank (section 3.2), (b) multiple-choice, and (c) movie retrieval task. The description of models for (b)–(c) can be found in the supplementary file. Each model takes advantage of the concept word detector in Fig.2, and semantic attention for the sake of its objective.

Finally, the output word probability $\mathbf{y}$ given $\{\mathbf{c}_t\}_{t=1}^T$ is obtained via softmax on $\mathbf{p}$ as

$$p(\mathbf{y} \mid \{\mathbf{c}_t\}_{t=1}^T) = \text{softmax}(\mathbf{W}_y \mathbf{p} + \mathbf{b}_y), \quad (19)$$

where parameters include $\mathbf{W}_y \in \mathsf{R}^{|\mathcal{V}| \times D}$ and $\mathbf{b}_y \in \mathsf{R}^{|\mathcal{V}|}$.

Training. During training, we minimize the loss $\mathcal{L}$ as

$$\mathcal{L} = -\log p(\mathbf{y}) + \lambda_1 (g(\beta) + g(\gamma)) + \lambda_2 \mathcal{L}_{con}, \quad (20)$$

where λ_1, λ_2 are hyperparameters, and g is the same regularization function of Eq.(15). Again, $\mathcal{L}_{con}$ is the cost of the concept word detector in Eq.(6).

4. Experiments

We report the experimental results of the proposed models for the four tasks of LSMDC 2016. More experimental results and implementation details can be found in the supplementary file.

4.1. The LSMDC Dataset and Tasks

The LSMDC 2016 comprises four video-to-language tasks on the LSMDC dataset, which contains a parallel corpus of 118,114 sentences and 118,081 video clips sampled from 202 movies. We strictly follow the evaluation protocols of the challenge. We defer more details of the dataset and challenge rules to [18] and the challenge homepage[1].

Movie Description. This task is related to video captioning; given a short video clip, its goal is to generate a single descriptive sentence. The challenge provides a subset of LSMDC dataset named *LSMDC16*. It is divided into training, validation, public test, and blind test set, whose sizes are 91,941, 6,542, 10,053, and 9,578, respectively. The official performance metrics include BLEU-1,2,3,4 [15], METEOR [2], ROUGE-L [12] and CIDEr [25].

[1] https://sites.google.com/site/describingmovies/.

Multiple-Choice Test. Given a video query and five candidate captions, from which its goal is to find the best option. The correct answer is the GT caption of the query video, and four other distractors are randomly chosen from the other captions that have different activity-phrase labels from the correct answer. The evaluation metric is the percentage of correctly answered test questions out of 10,053 public-test data.

Movie Retrieval. The objective is, given a short query sentence, to search for its corresponding video out of 1,000 candidate videos, sampled from the LSMDC16 public-test data. The evaluation metrics include Recall@1/5/10, and Median Rank (MedR). The Recall@k means the percentage of the GT video included in the first k retrieved videos, and the MedR indicates the median rank of the GT. Each algorithm predicts $1,000 \times 1,000$ pairwise rank scores between phrases and videos, from which all the evaluation metrics are calculated.

Movie Fill-in-the-Blank. This task is related to visual question answering; given a video clip and a sentence with a blank in it, its goal is to predict a single correct word to fill in the blank. The test set includes 30,000 examples from 10,000 clips (*i.e.* about 3 examples per sentence). The evaluation metric is the prediction accuracy, which is the percentage of predicted words that match with GTs.

4.2. Quantitative Results

We compare with the results on the public dataset in the official evaluation server of LSMDC 2016 as of the submission deadline (*i.e.* November 15th, 2016 UTC 23:59). Except award winners, the LSMDC participants have no obligation to disclose their identities or used technique. Below we use the IDs in the leaderboard to denote participants.

Movie description. Table 1 compares the performance of movie description between different algorithms. Among comparable models, our approach ranks (5, 4, 1, 1)-th in the

Movie Description	B1	B2	B3	B4	M	R	Cr
EITanque [11]	0.144 (4)	0.042 (5)	0.016 (3)	0.007 (2)	0.056 (7)	0.130 (7)	0.098 (2)
S2VT [27]	**0.162** (1)	**0.051** (1)	**0.017** (1)	0.007 (2)	0.070 (4)	0.149 (4)	0.082 (4)
SNUVL	0.157 (2)	0.049 (2)	0.014 (4)	0.004 (6)	0.071 (2)	0.147 (5)	0.070 (6)
sophieag	0.151 (3)	0.047 (3)	0.013 (5)	0.005 (4)	**0.075** (1)	0.152 (2)	0.072 (5)
ayush11011995	0.116 (8)	0.032 (7)	0.011 (7)	0.004 (6)	0.070 (4)	0.138 (6)	0.042 (8)
rakshithShetty	0.119 (7)	0.024 (8)	0.007 (8)	0.003 (8)	0.046 (8)	0.108 (8)	0.044 (7)
Aalto	0.070 (9)	0.017 (9)	0.005 (9)	0.002 (9)	0.033 (9)	0.069 (9)	0.037 (9)
Base-SAN	0.123 (6)	0.038 (6)	0.013 (5)	0.005 (4)	0.066 (6)	0.150 (3)	0.090 (3)
CT-SAN	0.135 (5)	0.044 (4)	**0.017** (1)	**0.008** (1)	0.071 (2)	**0.159** (1)	**0.100** (1)

Fill-in-the-Blank	Accuracy
Simple-LSTM	30.9
Simple-BLSTM	31.6
Base-SAN (Single)	34.5
Merging-LSTM [13]	34.2
Base-SAN (Ensemble)	36.9
SNUVL (Single)	38.0
SNUVL (Ensemble)	40.7
CT-SAN (Single)	41.9
CT-SAN (Ensemble)	**42.7**

Table 1. **Left**: Performance comparison for the movie description task on the LSMDC 2016 public test dataset. For language metrics, we use BLEU (B), METEOR (M), ROUGE (R), and CIDEr (Cr). We also show the ranking in parentheses. **Right**: Accuracy comparison (in percentage) for the movie fill-in-the-blank task.

Tasks	Multiple-Choice	Movie Retrieval			
Methods	Accuracy	R@1	R@5	R@10	MedR
Aalto	39.7	–	–	–	–
SA-G+SA-FC7 [24]	55.1	3.0	8.8	13.2	114
LSTM+SA-FC7 [24]	56.3	3.3	10.2	15.6	88
C+LSTM+SA-FC7 [24]	58.1	4.3	12.6	18.9	98
Base-SAN (Single)	60.1	4.3	13.0	18.2	83
Base-SAN (Ensemble)	64.0	4.4	13.9	19.3	74
SNUVL (Single)	63.1	3.8	13.6	18.9	80
EITanque [11]	63.7	4.7	15.9	23.4	64
SNUVL (Ensemble)	65.7	3.6	14.7	23.9	50
CT-SAN (Single)	63.8	4.5	14.1	20.9	67
CT-SAN (Ensemble)	**67.0**	**5.1**	**16.3**	**25.2**	**46**

Table 2. Performance comparison for the multiple-choice test (accuracy in percentage) and movie retrieval task: Recall@k (R@k, higher is better) and Median Rank (MedR, lower is better).

BLEU language metrics, and (2, 1, 1)-th in the other language metrics. That is, our approach ranks first in four metrics, which means that our approach is comparable to the state-of-the-art methods. In order to quantify the improvement by the proposed concept word detection and semantic attention, we implement a variant (Base-SAN), which is our model of Fig.2 without those two components. As shown in Table 1, the performance gaps between (CT-SAN) and (Base-SAN) are significant.

Movie Fill-in-the-Blank. Table 1 also shows the results of the fill-in-the-blank task. We test an ensemble of our models, denoted by (CT-SAN) (Ensemble); the answer word is obtained by averaging the output word probabilities of three identical models trained independently. Our approach outperforms all the participants with large margins. We also compare our model with a couple of baselines: (CT-SAN) outperforms the simple single-layer LSTM/BLSTM variants with the scoring layer on top of the blank location, and (Base-SAN), which is the base model of (CT-SAN) without the concept detector and semantic attention.

Movie Multiple-Choice Test. For the multiple-choice test, our approach also ranks first as shown in Table 2. As in the fill-in-the-blank, the multiple-choice task also benefits from the concept detector and semantic attention. Moreover, an ensemble of six models trained independently further improves the accuracy from 63.8% to 67.0%.

Movie Retrieval. Table 2 compares Recall@k (R@k) and Median Rank (MedR) metrics between different methods. We also achieve the best retrieval performance with significant margins from baselines. Our (CT-SAN) (Ensemble) obtains the video-sentence similarity matrix with an ensemble of two different models. First, we train six retrieval models with different parameter initializations. Second, we obtain the similarity matrix using the multiple-choice version of (CT-SAN), because it can also generate a similarity score for a video-sentence pair. Finally, we average the seven similarity matrices into the final similarity matrix.

4.3. Qualitative Results

Fig.4 illustrates qualitative results of our algorithm with correct or wrong examples for each task. In each set, we show sampled frames of a query video, groundtruth (GT), our prediction (Ours), and the detected concept words. We provide more examples in the supplementary file.

Movie Description. Fig.4(a)-(b) illustrates examples of our movie description. The predicted sentences are often related to the content of clips closely, but the words themselves are not always identical to the GTs. For instance, the generated sentence for Fig.4(b) reads *the clock shows a minute*, which is relevant to the video clip although its GT sentence much focuses on *awards on a shelf*. Nonetheless, the concept words relevant to the GT sentence are well detected such as *office* or *clock*.

Movie Fill-in-the-Blank. Fig.4(c) shows that the detected concept words are well matched with the content of the clip, and possibly help predict the correct answer. Fig.4(d) is a near-miss case where our model also predict a plausible answer (*e.g. run* instead of *hurry*).

Movie Multiple-Choice Test. Fig.4(e) shows that our concept detection successfully guides the model to select the correct answer. Fig.4(f) is an example of failure to understand the situation; the fifth candidate is chosen because

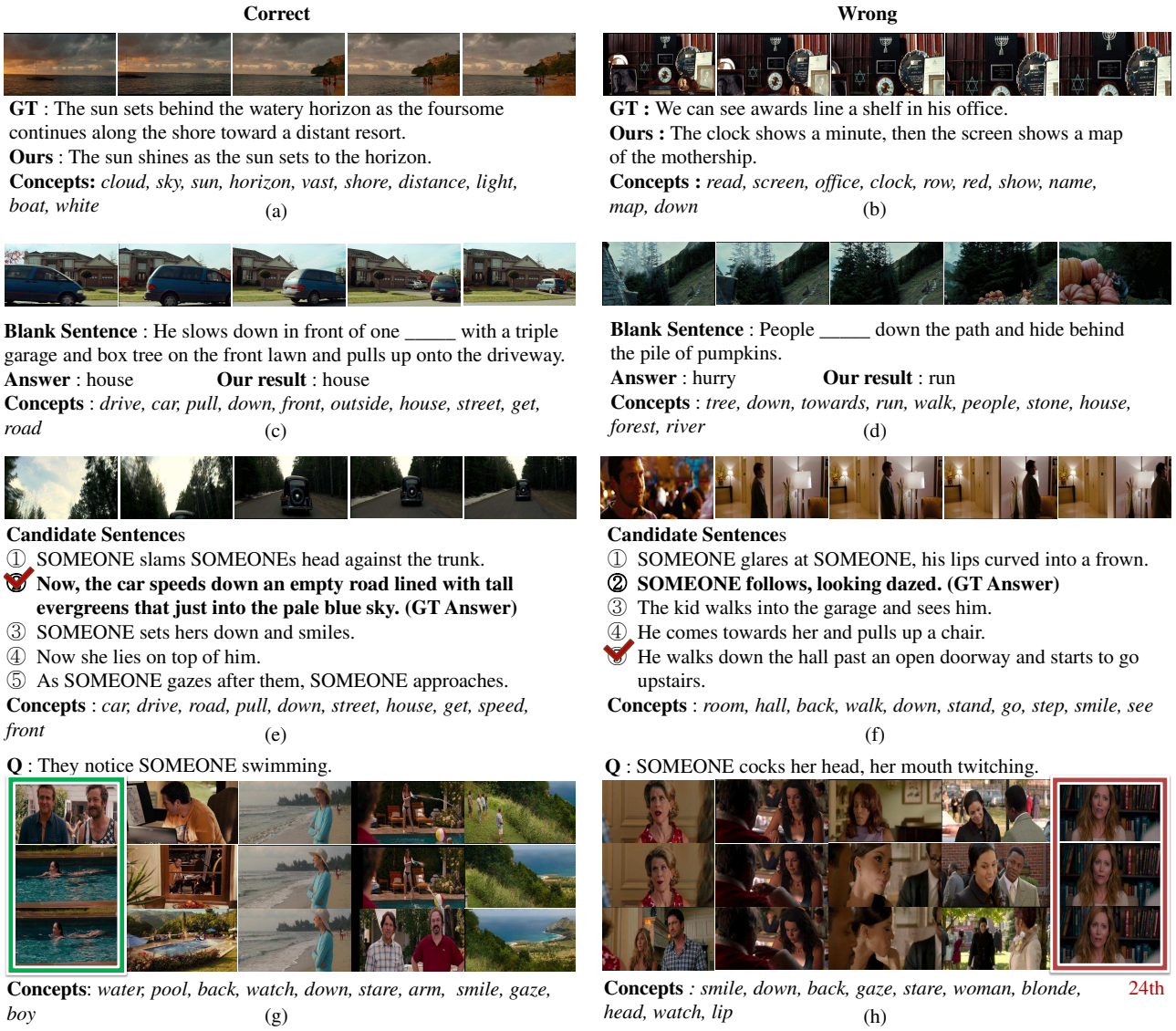

Figure 4. Qualitative examples of the four vision-to-language tasks: (a)-(b) movie description, (c)-(d) fill-in-the-blank, (e)-(f) multiple-choice, and (g)-(h) movie retrieval. The left column shows correct examples while the right column shows wrong examples. In (h), we also show our retrieval ranks of the GT clips (the red box), 24th. We present more, clearer, and larger examples in the supplementary file.

it is overlapped with much of detected words such as *hall*, *walk*, *go*, although the correct answer is the second.

Movie Retrieval. Interestingly, the concept words of Fig.4(g) capture the abstract relation between *swimming*, *water*, and *pool*. Thus, the first to fifth retrieved clips include *water*. Fig.4(h) is a near-miss example in which our method fails to catch rare word like *twitch* and *cocks*. The first to fourth retrieved clips contain a woman's head and mouth, yet miss to catch subtle movement of mouth.

5. Conclusion

We proposed an end-to-end trainable approach for detecting a list of concept words that can be used as semantic priors for multiple-video-to-language models. We also developed a semantic attention mechanism that effectively exploits the discovered concept words. We implemented our approach into multiple video-to-language models to participate in four tasks of LSMDC 2016. We demonstrated that our method indeed improved the performance of video captioning, retrieval, and question answering, and finally won three tasks in LSMDC 2016, including *fill-in-the-blank*, *multiple-choice test*, and *movie retrieval*.

Acknowledgements. This research is partially supported by Convergence Research Center through National Research Foundation of Korea (2015R1A5A7037676). Gunhee Kim is the corresponding author.

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer Normalization. *arXiv:1607.06450*, 2016. 4, 5

[2] S. Banerjee and A. Lavie. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *ACL*, 2005. 6

[3] S. Bird, E. Loper, and E. Klein. *Natural Language Processing with Python*. O'Reilly Media Inc., 2009. 3

[4] D. L. Chen and W. B. Dolan. Collecting Highly Parallel Data for Paraphrase Evaluation. In *ACL*, 2011. 1

[5] P. Das, C. Xu, R. F. Doell, and J. J. Corso. A Thousand Frames in Just a Few Words: Lingual Description of Videos through Latent Topics and Sparse Object Stitching. In *CVPR*, 2013. 2

[6] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. In *CVPR*, 2015. 1, 2

[7] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollar, J. Gao, X. He, M. Mitchell, J. C. Platt, C. Lawrence Zitnick, and G. Zweig. From Captions to Visual Concepts and Back. In *CVPR*, 2015. 2, 3

[8] 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. 1, 2

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 3

[10] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. In *IEEE*, 1997. 5

[11] D. Kaufman, G. Levi, T. Hassner, and L. Wolf. Temporal tessellation for video annotation and summarization. *arXiv:1612.06950*, 2016. 7

[12] C.-Y. Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *WAS*, 2004. 6

[13] A. Mazaheri, D. Zhang, and M. Shah. Video fill in the blank with merging lstms. *arXiv:1610.04062*, 2016. 7

[14] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, 2013. 3

[15] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL*, 2002. 6

[16] A. Rohrbach, M. Rohrbach, W. Qiu, A. Friedrich, M. Pinkal, and B. Schiele. Coherent Multi-Sentence Video Description with Variable Level of Detail. In *GCPR*, 2014. 1

[17] A. Rohrbach, M. Rohrbach, and B. Schiele. The Long-Short Story of Movie Description. In *GCPR*, 2015. 1, 2

[18] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. Pal, H. Larochelle, A. Courville, and B. Schiele. Movie Description. *arXiv:1605.03705*, 2016. 1, 2, 6

[19] M. Rohrbach, W. Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele. Translating Video Content to Natural Language Descriptions. In *ICCV*, 2013. 2

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 3

[21] M. Schuster and K. K. Paliwal. Bidirectional Recurrent Neural Networks. In *IEEE TSP*, 1997. 5

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *JMLR*, 2014. 4

[23] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler. MovieQA: Understanding Stories in Movies through Question-Answering. In *CVPR*, 2016. 1

[24] A. Torabi, N. Tandon, and L. Sigal. Learning Language-Visual Embedding for Movie Understanding with Natural-Language. *arXiv:1609.08124*, 2016. 7

[25] R. Vedantam, C. L. Zitnick, and D. Parikh. CIDEr: Consensus-based Image Description Evaluation. In *CVPR*, 2015. 6

[26] S. Venugopalan, L. A. Hendricks, M. Rohrbach, R. Mooney, T. Darrell, and K. Saenko. Captioning Images with Diverse Objects. *arXiv:1606.07770*, 2016. 2

[27] S. Venugopalan, R. Marcus, D. Jeffrey, M. Raymond, D. Trevor, and S. Kate. Sequence to Sequence - Video to Text. In *ICCV*, 2015. 1, 2, 7

[28] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating Videos to Natural Language Using Deep Recurrent Neural Networks. In *HLT-NAACL*, 2015. 2

[29] Q. Wu, C. Shen, A. v. d. Hengel, P. Wang, and A. Dick. Image Captioning and Visual Question Answering Based on Attributes and Their Related External Knowledge. *arXiv:1603.02814*, 2016. 2

[30] Q. Wu, C. Shen, L. Liu, A. Dick, and A. van den Hengel. What value do explicit high level concepts have in vision to language problems? In *CVPR*, 2016. 4

[31] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, 2015. 2

[32] R. Xu, C. Xiong, W. Chen, and J. J. Corso. Jointly Modeling Deep Video and Compositional Text to Bridge Vision and Language in a Unified Framework. In *AAAI*, 2015. 1

[33] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing Videos by Exploiting Temporal Structure. In *ICCV*, 2015. 2

[34] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *CVPR*, 2016. 2, 5

[35] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. Video Paragraph Captioning Using Hierarchical Recurrent Neural Networks. In *CVPR*, 2016. 1, 2