Video Captioning via Hierarchical Reinforcement Learning

Xin Wang,   Wenhu Chen,   Jiawei Wu,   Yuan-Fang Wang,   William Yang Wang

University of California, Santa Barbara

{xwang,wenhuchen,jiawei_wu,yfwang,william}@cs.ucsb.edu

Abstract

Video captioning is the task of automatically generating a textual description of the actions in a video. Although previous work (e.g. sequence-to-sequence model) has shown promising results in abstracting a coarse description of a short video, it is still very challenging to caption a video containing multiple fine-grained actions with a detailed description. This paper aims to address the challenge by proposing a novel hierarchical reinforcement learning framework for video captioning, where a high-level Manager module learns to design sub-goals and a low-level Worker module recognizes the primitive actions to fulfill the sub-goal. With this compositional framework to reinforce video captioning at different levels, our approach significantly outperforms all the baseline methods on a newly introduced large-scale dataset for fine-grained video captioning. Furthermore, our non-ensemble model has already achieved the state-of-the-art results on the widely-used MSR-VTT dataset.

1. Introduction

For most people, watching a brief video and describing what happened (in words) is an easy task. For machines, extracting the meaning from video pixels and generating natural-sounding description is a very challenging problem. However, due to its wide range of applications such as intelligent video surveillance and assistance to visually-impaired people, video captioning has drawn increasing attention from the computer vision community recently. Different from *image* captioning which aims at describing a static scene, *video* captioning is more challenging in the sense that a series of coherent scenes need to be understood in order to jointly generate multiple description segments (*e.g.*, see Figure 1).

Current video captioning tasks can mainly be divided into two families, single-sentence generation [38, 16] and paragraph generation [24]. Single-sentence generation tends to abstract a whole video to a simple and high-level

Caption #1: A woman offers her dog some food.
Caption #2: A woman is eating and sharing food with her dog.
Caption #3: A woman is sharing a snack with a dog.

Caption: A person sits on a bed and puts a laptop into a bag. The person stands up, puts the bag on one shoulder, and walks out of the room.

Figure 1: Video captioning examples. Top row is an example from MSR-VTT dataset [38], which is summarized by three single captions. Bottom row is an example from Charades [28] dataset, which consists of several dependent human activities and is described by multiple long sentences of complex structure.

descriptive sentence, while paragraph generation tends to grasp more detailed actions, and generates multiple sentences of descriptions. However, even for paragraph generation, the paragraph is often split into multiple, single-sentence generation scenarios associated with ground truth temporal video intervals.

In many practical cases, human activities are too complex to be described with short, simple sentences, and the temporal intervals are hard to be predicted ahead of time without a good understanding of the linguistic context. For instance, in the bottom example of Figure 1, there are five human actions in total: *sit on a bed*, *put a laptop into a bag* are happening simultaneously, and then followed by *stand up*, *put the bag on one shoulder* and *walk out of the room* in order. Such fine-grained caption requires a subtle and expressive mechanism to capture the temporal dynamics of the video content and associate that with semantic representations in natural language.

In order to tackle this issue, we propose a "divide and

conquer" solution, which first divides a long caption into many small text segments (*e.g.* different segments are in different colors as shown in Figure 1), and then employs a sequence model to conquer each segment. Instead of forcing the sequence model to generate the whole sequence in one shot, we propose to guide the model to generate sentences segment by segment. With a higher-level sequence model designing the context of each segment, the low-level sequence model follows the guidance to generate the segment word by word.

In this paper, we propose a novel hierarchical reinforcement learning (HRL) framework to realize this two-level mechanism. The textual and video context can be viewed as the reinforcement learning **environment**. Our framework is a fully-differentiable deep neural network (see Figure 2) and consists of (1) the higher-level sequence model **manager** that sets **goals** at a lower temporal resolution, (2) the lower-level sequence model **worker** that selects primitive **actions** at every time step by following the goals from the Manager, and (3) an **internal critic** that determines whether a goal is accomplished or not. More specifically, by exploiting the context from both the environment and finished goals, the manager emits a new goal for a new segment, and the worker receives the goal as guidance to generate the segment by producing words sequentially. Moreover, the internal critic is employed to evaluate whether the current textual segment is accomplished.

Furthermore, we equip both the manager and worker with an attention module over the video features (Sec 3.2) to introduce hierarchical attention internally so that the manager will focus on a wider range of temporal dynamics while the worker's attention is narrowed down to local dynamics conditioned on the goals. To the best of our knowledge, this is the first work that strives to develop a hierarchical reinforcement learning approach to reinforce video captioning at different levels. Our main contributions are four-fold:

- We propose a hierarchical deep reinforcement learning framework to efficiently learn the semantic dynamics when captioning a video.

- We formulate an alternative, novel training approach over stochastic and deterministic policy gradient.

- We introduce a new large-scale dataset for fine-grained video captioning, Charades Captions[1], and validate the effectiveness of the proposed method in it.

- We further evaluate our approach on MSR-VTT dataset and achieve the state-of-the-art results even when training on a single type of features.

---

[1]Charades Captions was obtained by preprocessing the raw Charades dataset [28]. The processed Charades Captions dataset can be downloaded here: http://www.cs.ucsb.edu/~xwang/data/CharadesCaptions.zip

2. Related Work

Video Captioning S2VT [33] first generalized LSTM to video captioning and proposed a sequence-to-sequence model for it. Since then, numerous improvements were introduced, such as attention [39, 41], hierarchical recurrent neural network (RNN) [40, 15, 2, 30, 36], C3D features [26], joint embedding space [20], language fusion [5], multi-task learning [17], etc. But most of them use the maximum-likelihood algorithm, which maximizes the probability of current ground-truth output given previous ground-truth output, while the previous ground-truth is in general unknown during test time. This inconsistency issue known as exposure bias has largely hindered the system performance.

In order to address the inconsistency issue, Ranzato *et al.* [21] proposed to directly optimize non-differentiable metric scores using the REINFORCE algorithm [37]. But the problem persisted that the expected gradient computed using policy gradient typically exhibited high variance and was often unstable without proper context-dependent normalization. Naturally, the variance could be reduced by adding a baseline [13, 23] or even an actor-critic method that trained an additional critic to estimate the value of each generated word [1, 22, 43]. Pasunuru and Bansal [18] applied policy gradient with baseline on video captioning and presented textual entailment loss to adjust the CIDEr reward. Unfortunately, these previous work for image/video captioning fail to grasp the high-level semantic flow. Our HRL model aims to address this issue with a hierarchical reinforcement learning framework.

Another line of work is dense video captioning [9], which focuses on detecting multiple events that occur in a video and describing each of them. But it does not aim to solve the single-sentence generation scenario. While our method aims to generate one or multiple sentences for a sequence of continuous actions (one or multiple).

Hierarchical Reinforcement Learning Recent work has revealed the effectiveness of hierarchical reinforcement learning frameworks on Atari games [11, 35]. Peng *et al.* built a composite dialogue policy using hierarchical Q-learning to fulfill complex dialogue tasks like traveling plans [19]. In the typical HRL setting, there was a high-level agent that operated at the lower temporal resolution to set a sub-goal, and a low-level agent that selected primitive actions by following the sub-goal from the high-level agent. Our proposed HRL framework for video captioning is aligned to these studies but has a key difference from the typical HRL setting: instead of having the internal critic to provide an intrinsic reward to encourage the low-level agent to accomplish the sub-goal, we focus on exploiting the extrinsic rewards in different time spans. Besides, we
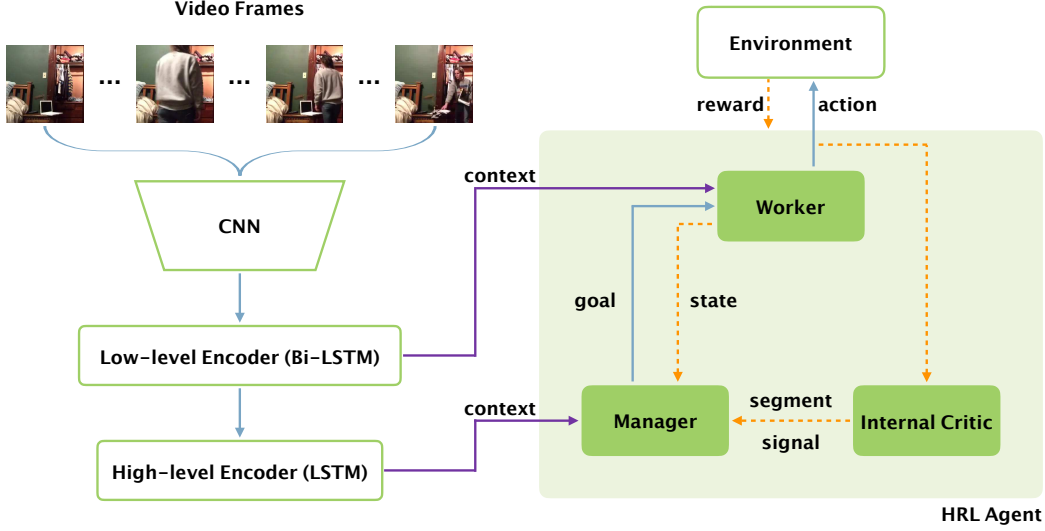
Figure 2: Overview of the HRL framework for video captioning. Please see Sec. 3.1 for explanation.

are the first to consider HRL in the intersection vision and language.

3. Our Approach

3.1. Overview

Our proposed HRL framework follows the general encoder-decoder framework (see Figure 2). In the encoding stage, video frame features $v = \{v_i\}$ are first extracted by a pretrained convolutional neural network (CNN) [10] model, where $i \in \{1, ..., n\}$ indexes the frames in the temporal order. Then the frame features are passed through a low-level Bi-LSTM[2] encoder and a high-level LSTM[3] encoder successively to obtain low-level encoder output $h^{E_w} = \{h_i^{E_w}\}$ (E_w denotes the encoder associated with the Worker), and high-level encoder output $h^{E_m} = \{h_i^{E_m}\}$ (E_m denoting the encoder associated with the Manager), where $i \in \{1, ..., n\}$. In the decoding stage, our HRL agent plays the role of a decoder, and outputs a language description $a_1 a_2 ... a_T \in V^T$, where T is the length of the generated caption and V is the vocabulary set.

The HRL agent is composed of three components: a low-level *worker*, a high-level *manager*, and an *internal critic*. The manager operates at a lower temporal resolution and emits a goal when needed for the worker to accomplish, and the worker generates a word for each time step by following the goal proposed by the manager. In other words, the manager asks the worker to generate a semantic segment, and the worker generates the corresponding words in the next few time steps in order to fulfill the job. The internal critic determines if the worker has accomplished the goal

---
[2]Bidirectional long short-term memory [25]
[3]Long short-term memory [7]

and sends a binary segment signal to the manager to help it update goals. The whole pipeline terminates once an end of sentence token is reached.

3.2. Policy Network

Attention Module As mentioned above, the CNN-RNN encoder receives the video inputs to generate a sequence of vectors $h^{E_w} = \{h_i^{E_w}\}$ and $h^{E_m} = \{h_i^{E_m}\}$. One may directly take them as the inputs to the worker and the manager. We instead adopt an attention mechanism to better capture the temporal dynamics, and form the context vector for their use. In our model, both the manager and the worker are equipped with an attention module.

The left-hand side of Figure 3 is a demo attention module for the worker, at each time step t, the context vector c_t^W is computed as a weighted sum over the encoder's all hidden states $\{h_i^{E_w}\}$

$$c_t^W = \sum \alpha_{t,i}^W h_i^{E_w} \qquad (1)$$

These attention weights $\{\alpha_{t,i}^W\}$ act as an alignment mechanism by giving higher weights to certain encoder hidden states which match the worker's current status, and are defined as

$$\alpha_{t,i}^W = \frac{\exp(e_{t,i})}{\sum_{k=1}^n \exp(e_{t,k})} \qquad (2)$$

where

$$e_{t,i} = w^T \tanh(W_a h_i^{E_w} + U_a h_{t-1}^W + b_a) \qquad (3)$$

where w, W_a, U_a and b_a are learned parameters; h_{t-1}^W is the worker LSTM's hidden state at previous step.

The manager's attention module follows the same paradigm as the worker's, which can be described by replacing the corresponding terms in Equation 1, 2, and 3.
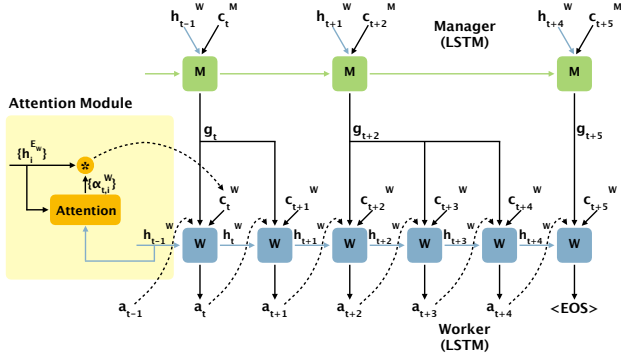
4215

Figure 3: An example of the unrolled HRL agent in the decoding stage (from time step t to $t + 5$). The yellow region shows how the attention module is incorporated into the encoder-decoder framework.

Manager and Worker As is shown in Figure 3, the concatenation of $[c_t^M, h_{t-1}^W]$ is fed as the input to the manager LSTM to produce the semantically meaningful goal. With the help of the context and the sentence state at previous time steps, the manager can obtain the knowledge of the environment status. The output of the manager LSTM h_t^M is then projected as a latent continuous goal vector g_t. Formally,

$$h_t^M = S^M(h_{t-1}^M, [c_t^M, h_{t-1}^W]) \tag{4}$$

$$g_t = u_M(h_t^M) \tag{5}$$

where S^M denotes the non-linear function of the manager LSTM and u_M is a function to project hidden states into goal space.

The worker receives the goal g_t, takes the concatenation of $[c_t^W, g_t, a_{t-1}]$ as the input, and outputs the probabilities π_t over all actions $a_t \in V$ after a series of computations:

$$h_t^W = S^W(h_{t-1}^W, [c_t^W, g_t, a_{t-1}]) \tag{6}$$

$$x_t = u_W(h_t^W) \tag{7}$$

$$\pi_t = SoftMax(x_t) \tag{8}$$

where S^W is the non-linear function of the worker LSTM and u_W is a also a function to project hidden states into the input to softmax layer.

Internal Critic In order to determine whether the worker has accomplished a goal g_t, we employ an internal critic to evaluate worker's progress. The internal critic uses an RNN structure, which takes a word sequence as the input to discriminate whether an end has been reached. Let z_t denote the signal of internal critic and h_t^I denote the hidden state of the RNN at time step t, formally we describe the

probability $p(z_t)$ as follows:

$$h_t^I = RNN(h_{t-1}^I, a_t)$$
$$p(z_t) = sigmoid(W_z h_t^I + b_z) \tag{9}$$

where a_t is the action taken by the worker and W_z, b_z denotes the parameters of the feed-forward neural network. In order to train the parameters of the linear layer and recurrent network, we propose to maximize the likelihood of given ground truth signal $\{z_t^*\}$:

$$\text{argmax} \sum_t \log p(z_t^* | a_1, \cdots, a_{t-1}) \tag{10}$$

Once the critic model is optimized, we will fix it to service the usage of the manager.

3.3. Learning

As described in Sec. 3.2, the manager policy is actually deterministic, which can be further denoted as $g_t = \mu_{\theta_m}(s_t)$ with θ_m representing the parameters of the manager, while the worker policy is a stochastic policy denoted by $a_t \sim \pi_{\theta_w}(a_t; s_t, g_t)$, where θ_w represents the parameters of the worker. The reason why the worker policy is stochastic is that its action a_t is selecting a word from the vocabulary V. But for the manager, the generated goal is latent, which cannot be directly supervised. Thus with a deterministic manager policy, we can warm start both the manager and worker simultaneously by viewing them as a composite agent.

In this section, we first derive the mathematical reinforce learning methods for the policies separately (Sec. 3.3.1 and 3.3.2), and then introduce the training algorithm of the proposed HRL method (Sec. 3.3.4). We also discuss the reward definitions (Sec. 3.3.3) and imitation learning of our HRL policy (Sec. 3.3.5).

3.3.1 Stochastic Worker Policy Learning

We consider a standard reinforcement learning setup. At each step t, the worker select an action a_t ($a_t \in V$) conditioned on g_t from the manager. The environment responds with a new state s_{t+1} and a scalar reward r_t. The process continues until a <EOS> token is generated. The objective of the worker is to maximize the discounted return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$. Thus its loss function can be written as

$$L(\theta_w) = -\mathbb{E}_{a_t \sim \pi_{\theta_w}}[R(a_t)] \tag{11}$$

to minimize the negative expected reward function. Based on REINFORCE algorithm [37], the gradient of non-differentiable, reward-based loss function can be derived as

$$\nabla_{\theta_w} L(\theta_w) = -\mathbb{E}_{a_t \sim \pi_{\theta_w}}[R(a_t) \nabla_{\theta_w} \log \pi_{\theta_w}(a_t)] \tag{12}$$

In practice $L(\theta_w)$ is typically estimated with a single sample from π_{θ_w}:

$$\nabla_{\theta_w} L(\theta_w) \approx -R(a_t)\nabla_{\theta_w} \log \pi_{\theta_w}(a_t) \qquad (13)$$

The policy gradient given by REINFORCE can be further generalized to reduce the variance without changing the expected gradient, by subtracting the reward with a baseline [31]:

$$\nabla_{\theta_w} L(\theta_w) \approx -(R(a_t) - b_t^w)\nabla_{\theta_w} \log \pi_{\theta_w}(a_t) \qquad (14)$$

where b_t^w is the estimated baseline, which can be a function of θ_w or t [21]. In our case, the baseline is estimated by a linear regressor with the worker's hidden state h_t^W as the input. During back propagation, the gradient passing is cut off between the worker LSTM and the baseline estimator.

For a better understanding of the policy gradient, we can further derive the loss function using the chain rule

$$\nabla_{\theta_w} L(\theta_w) = \sum_{t=1}^{T} \frac{\partial L}{\partial x_t} \frac{\partial x_t}{\partial \theta_w} \qquad (15)$$

where x_t is the input to the *SoftMax* layer (see Equation 7). Using REINFORCE with baseline the estimation of $\frac{\partial L}{\partial x_t}$ is given by [42]:

$$\frac{\partial L}{\partial x_t} = (R(a_t) - b_t^w)(\pi_{\theta_w}(a_t) - 1_{a_t}) \qquad (16)$$

which means if the reward $R(a_t)$ of the sample word a_t is greater than the baseline b_t, the gradient is negative and thus the model encourages the distribution by increasing the probability of the word, otherwise, it discourages the distribution accordingly.

3.3.2 Deterministic Manager Policy Learning

The key to our HRL framework is to effectively learn the goal g_t generated by the manager and then guides the worker to achieve the latent objective. But the difficulty of training the manager is that it does not directly interact with the environment because the action it takes is to produce a latent vector g_t in a continuous high-dimensional space, which indirectly influences the environment by directing the Worker's behavior. Therefore, we are especially interested in coming up solutions to encourage the manager towards more effective caption generation.

Inspired by the deterministic policy gradient algorithms [29, 12], we propose to learn the deterministic policy $\mu_{\theta_m}(s_t)$ from trajectories generated by the stochastic worker policy $\pi_{\theta_w}(a_t; s_t, g_t)$. When training the target manager policy, we fix the worker policy as an Oracle behavior policy. More specifically, the manager outputs a goal g_t at step t and the worker then runs c steps to generate

the expected segment $e_{t,c} = a_t a_{t+1}...a_{t+c-1}$ by following the goal (c is length of the generated segment). Since the worker is fixed as an Oracle behavior policy, we only need to consider the training of the manager. Then the environment responds with an new state s_{t+c} and a scalar reward $r(e_{t,c})$. Thus the objective becomes minimizing the negative discounted return $R_t(e_{t,c})$, in formula

$$L(\theta_m) = -\mathbb{E}_{g_t}[R(e_t)\pi(e_{t,c}; s_t, g_t = \mu_{\theta_m}(s_t)] \qquad (17)$$

After applying the chain rule to the loss function with respect to the manager's parameters θ_m, the manager is updated with

$$\nabla_{\theta_m} L(\theta_m) = -\mathbb{E}_{g_t}[R(e_{t,c})\nabla_{g_t}\pi(e_{t,c}; s_t, g_t)\nabla_{\theta_m}\mu_{\theta_m}(s_t)] \qquad (18)$$

The above gradients can be approximated from a single sampled segment $e_{t,c}$ and after adopting policy gradient on the worker policy,

$$\nabla_{\theta_m} L(\theta_m) = -R(e_{t,c})\nabla_{g_t} \log \pi(e_{t,c})\nabla_{\theta_m}\mu_{\theta_m}(s_t) \quad (19)$$

Since the worker LSTM is indeed a Markov decision process and the probability of the current action a_t is conditioned on the action a_{t-1} at previous step (see Equation 6,7,8), we have

$$\log \pi(e_{t,c}) = \log \pi(a_t..a_{t+c-1}) = \sum_{i=t}^{t+c-1} \log \pi(a_i) \quad (20)$$

Combining Equation 19 and 20, then the gradients become

$$\nabla_{\theta_m} L(\theta_m) = -R(e_{t,c})[\sum_{i=t}^{t+c-1} \nabla_{g_t} \log \pi(a_i)]\nabla_{\theta_m}\mu_{\theta_m}(s_t) \qquad (21)$$

The final gradients for the manager training is obtained by adding the baseline estimator to reduce the variance as follows:

$$\nabla_{\theta_m} L(\theta_m) = $$
$$-(R(e_{t,c}) - b_t^m)[\sum_{i=t}^{t+c-1} \nabla_{g_t} \log \pi(a_i)]\nabla_{\theta_m}\mu_{\theta_m}(s_t) \qquad (22)$$

where b_t^m is the baseline estimator, which is a linear regressor with the manager's hidden state h_t^M as the input.

A major challenge of learning in continuous action spaces is exploration. We follow the known DDPG [12] to construct an exploration policy μ' by adding perturbation ϵ sampled from a Gaussian distribution $\mathcal{N}$ to our manager policy

$$\mu'(s_t) = \mu_{\theta_m}(s_t) + \epsilon \qquad (23)$$

and the variance of Gaussian noise can be chosen to suit the environment.

3.3.3 Reward Definition

Recent work on image captioning [23] has shown that CIDEr as a reward performs the best among the traditional evaluation metrics (*e.g.* CIDEr, BLEU or METEOR) for image/video captioning and can gain improvement on all other metrics. In our model, we also use CIDEr score to compute the reward. But instead of directly using the final CIDEr score of the whole generated caption as the reward for each word a_t, we adopt delta CIDEr score as the immediate reward. Let $f(x) = \text{CIDEr}(sent + x) - \text{CIDEr}(sent)$, where $sent$ is the previous generated caption. Then the discounted return for the worker is

$$R(a_t) = \sum_{k=0}^{\infty} \gamma^k f(a_{t+k}) \qquad (24)$$

where k denotes the time step of the worker's temporal resolution, and the discounted return for the manager is

$$R(e_t) = \sum_{n=0}^{\infty} \gamma^n f(e_{t+n}) \qquad (25)$$

where n is the time step of the manager's lower temporal resolution. Note that our approach is not limited to CIDEr score, other reasonable rewards (*e.g.* deltaBLEU [4]) can also be applied to the HRL framework.

3.3.4 Training Algorithm

Above we illustrate the learning methods to train the manager and the worker. In Algorithm 1 we present the pseudo-code of our HRL training algorithm for video captioning. The manager policy and the worker policy are trained alternately. Basically, when training the worker, we assume the manager is well-posed, so we disable the goal exploration and only update the worker policy according to Equation 14; when training the manager, we treat the worker as the Oracle behavior policy, so we generate the caption by greedy decoding and only update the manager policy following Equation 22.

During testing, goal exploration is disabled, and beam search is employed to generate the results. Only one forward pass is needed at test time.

3.3.5 Imitation Learning

A major challenge for a reinforcement learning agent to have good convergence property is that the agent must start with a good policy at the beginning stage. For our model, we apply the cross-entropy loss optimization to warm start both the worker and the manager simultaneously, where the manager is completely treated as the latent parameters. θ be the parameters of the whole model and $a_1^*, a_2^*, ..., a_T^*$ be the

---

Algorithm 1 HRL training algorithm

Require: Training pairs <video, GT caption>
1: Randomly initialize the model parameters θ
2: Load the pretrained CNN model and internal critic
3: **for** iteration=1,M **do**
4:     Randomly sample a minibatch
5:     **if** Train-Worker **then**
6:         Disable the goal exploration
7:         Run a forward pass to get the sampled caption $a_1 a_2 ... a_T$
8:         Calculate $R(a_t)$ for each a_t
9:         Freeze the manager
10:        Update the worker policy using Equation 14
11:     **else if** Train-Manager **then**
12:        Initialize a random process $\mathcal{N}$ for goal exploration
13:        Run a forward pass to get the greedily decoded caption $e_1 e_2 ... e_n$
14:        Calculate $R(e_t)$ for each e_t
15:        Freeze the worker
16:        Update the manager policy using Equation 22
17:     **end if**
18: **end for**

---

ground-truth word sequence, then the cross-entropy loss is defined as

$$L(\theta) = -\sum_{t=1}^{T} \log(\pi_\theta(a_t^*; a_1^*, ..., a_{t-1}^*)) \qquad (26)$$

4. Experimental Results

4.1. Datasets

MSR-VTT   MSR-VTT [38] is a dataset for general video captioning, which is derived from a wide variety of video categories (7,180 videos from 20 general categories), and contains 10,000 video clips (6,513 for training, 497 for validation, and the remaining 2,990 for testing). Each video contains 20 human annotated reference captions collected by Amazon Mechanical Turk (AMT).

Charades Captions   Charades [28] is a large-scale dataset composed of 9,848 videos of daily indoors activities collected through AMT. 267 different users were presented with a sentence script (*e.g.* a person fixes the bed then throws pillow on it) that included objects and actions from a fixed vocabulary, and the users recorded a video following the script using provided objects and actions. The original dataset contains 66,500 temporal annotations for 157 action classes, 41,104 labels for 46 object classes, and 27,847 textual descriptions of the videos.

Method	BLEU@4	METEOR	ROUGE-L	CIDEr
Mean-Pooling	30.4	23.7	52.0	35.0
Soft-Attention	28.5	25.0	53.3	37.1
S2VT	31.4	25.7	55.9	35.2
v2t_navigator	40.8	28.2	60.9	44.8
Aalto	39.8	26.9	59.8	45.7
VideoLAB	39.1	27.7	60.6	44.1
XE-baseline	**41.3**	27.6	59.9	44.7
RL-baseline	40.6	28.5	60.7	46.3
HRL (Ours)	**41.3**	**28.7**	**61.7**	**48.0**

Table 1: Comparison with state of the arts on MSR-VTT dataset.

Method	B@1	B@2	B@3	B@4	M	R	C
XE-baseline	55.0	36.4	23.6	15.0	18.7	39.0	16.7
RL-baseline	57.6	41.4	28.0	**18.8**	17.7	39.8	21.6
HRL-16	**64.4**	**44.3**	**29.4**	**18.8**	**19.5**	**41.4**	23.2
HRL-32	64.0	43.4	28.4	17.9	19.2	41.0	21.3
HRL-64	61.7	43.0	28.8	**18.8**	18.7	31.2	**23.6**

Table 2: Results on Charades Captions dataset. We reported BLEU (B), METEOR (M), ROUGH-L (R) and CIDEr (C) scores of our HRL method and two baselines for comparison.

While the Charades dataset is mainly used for action recognition and segmentation, one should note that the collected textual descriptions are very detailed and depict the fine-grained human activities happening in long videos. Thus, we preprocessed the raw Charades dataset by combining the textual descriptions and sentence scripts verified through AMT[4], and built a new large-scale dataset for detailed video captioning – Charades Captions, which consists of 6,963 videos for training, 500 for validation and 1,760 for testing. Each video clip is annotated by multiple (typically 2-5) captions. The captions are more detailed and longer than those of MSR-VTT (average caption length: 24.13 *vs* 9.28 words), which is more suitable for fine-grained video captioning.

Caption Segmentation In order to train the internal critic that determines if a goal is accomplished, we preprocessed the ground truth captions of the training sets of both datasets by breaking each caption into multiple semantic chunks. We segmented the captions mainly based on the Noun Phrase (NP) and Verb Phrase (VP) labels provided by the constituency parsing results (We utilized the open source toolkits Stanford CoreNLP[5] [14] and NLTK[6] for constituency parsing). For instance, the caption *The person then tidies his area after he is done eating* was segmented into three sub-phrases, *The person*, *then tidies his area* and *after he is done eating* with labels NP, VP and VP respectively. However, all we need to train the internal critic were the chunks, and labels were not used.

4.2. Experimental Setup

Evaluation Metrics We adopted four diverse automatic evaluation metrics: BLEU, METEOR, ROUGE-L, and CIDEr-D. We used the standard evaluation code from MS-COCO server [3] to obtain the results.

---

[4]For example, the sentence script of a video can be *A person is taking a picture of a light while sitting in a chair.*, and the textual description is *A person in a bedroom appears to use their phone to film or take a picture of the light fixture on the ceiling.* The latter is usually more detailed.

[5]https://stanfordnlp.github.io/CoreNLP/
[6]http://www.nltk.org

Training Details All the hyperparameters were tuned on the validation set. For both datasets, we sampled each video at $3fps$ and extracted ResNet-152 features [6] from these sampled frames without fine-tuning. More training details can be found in the supplementary material.

4.3. Results and Analysis

Comparison with state of the arts on MSR-VTT In Table 1, we compared our single-sentence captioning results with the-state-of-the-art methods on MSR-VTT dataset. We listed the results of Mean-Pooling [34], Soft-Attention [39] and S2VT [33] as reported in previous work [26]. We also compared with the top-3 results from MSR-VTT challenge, including v2t navigator [8], Aalto [27], VideoLAB [20].

We implemented two baseline methods: an attention-based sequence-to-sequence model trained with cross-entropy loss (*XE-baseline*), and the same model trained with policy gradient and CIDEr score as the RL reward (*RL-baseline*). As shown in Table 1, our XE-baseline achieved comparable results with the state-of-the-art results, and our RL-baseline further improved on all metrics. Moreover, our novel HRL method outperformed all the other algorithms listed in the table, which proved the effectiveness of our proposed method.

Result Analysis on Charades Captions Since there were no other papers reporting results on Charades Captions, we mainly compared our HRL model with our implementation of XE-baseline and RL-baseline. Meanwhile, we explored the dimension of the latent goal vector (We used HRL-X to denote the HRL model with a goal dimension of X). As can be observed from Table 2, all our HRL models outperformed the baseline methods and brought significant improvements in different evaluation metrics. Note that our HRL model achieved bigger improvement over the baseline methods on Charades Captions dataset than on MSR-VTT. Given that fact that the average cation length of Charades Captions was much longer than that of MST-VTT (24.13 *vs* 9.28 words), the difference of the improvement gaps demonstrated that our HRL model can gain better improvement on detailed descriptions of longer videos.

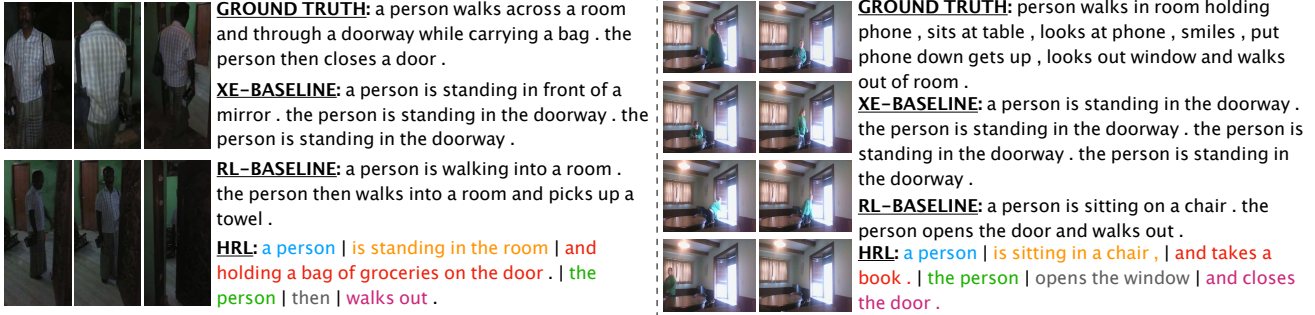Among the HRL models, HRL-16 achieved the best on

Figure 4: Qualitative comparison with the baseline methods. The given examples were from the test set of the Charades Caption dataset.

almost all metrics (CIDEr score was the second-best and slightly worse than HRL-64). Even though HRL-64 obtained better results on BLEU@4 and CIDEr, its results on other metrics were worse than HRL-32 (the ROUGE-L score was much lower than HRL-32). Thus, comparing the results of different HRL models, we could conclude that HRL-16 > HRL-32 $\geq$ HRL-64. This result accorded with our speculation: higher dimension does not guarantee better performance, conversely, the exploration space grows exponentially as the dimension increases, making the learning even harder. A latent vector of small dimension like 16 is able to represent the semantically meaningful goal well.

Qualitative Comparison with Baseline Methods In Figure 4, we illustrated two examples from Charades Captions test set. According to the captions generated by different models, it is obvious that the generated results of our HRL model matched the ground truth captions better than the baseline methods. Moreover, due to the segment-by-segment generation manner, our HRL model was able to output a sequence of semantically meaningful phases (different phases were in different colors and segmented by "|" as in Figure 4).

Learning Curve For a more intuitive view of the models, we drew the learning curves of the CIDEr scores on validation set (see Figure 5). Note that the RL-baseline model was first warmed up with cross-entropy loss, and then improved using the REINFORCE algorithm. Particularly, after we trained the XE-baseline model, we switched to policy gradient and continued training the RL-baseline model on it. HRL models were resumed training on a shorter warm-start period. As is shown in Figure 5, the HRL models converged faster and achieved better peak points than the baseline methods. HRL-16 reached the highest point.

5. Conclusion

In this paper, we propose a hierarchical reinforcement learning framework for video captioning, which aims at im-
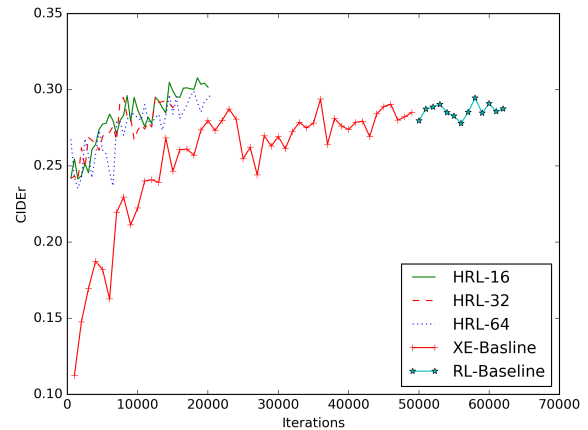

Figure 5: Learning curves of the CIDEr scores of different captioning models, including XE-baseline, RL-baseline and HRL models with goal dimension of 16, 32 and 64.

proving the fine-grained generation of video descriptions with rich activities. Our HRL model obtains the state-of-the-art performance on both the widely used MSR-VTT dataset and the newly introduced Charades Captions dataset for fine-grained video captioning.

In the future, we plan to explore the attention space and utilize features from multiple modalities to boost our HRL agent. We believe that the results of our method can be further improved by employing different types of features, *i.e.* C3D features [32], optical flows, etc. Meanwhile, we will investigate the HRL framework in other similar sequence generation tasks like video/document summarization.

Acknowledgement

References

[1] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*, 2016. 2

[2] L. Baraldi, C. Grana, and R. Cucchiara. Hierarchical boundary-aware neural encoder for video captioning. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 3185–3194. IEEE, 2017. 2

[3] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015. 7

[4] M. Galley, C. Brockett, A. Sordoni, Y. Ji, M. Auli, C. Quirk, M. Mitchell, J. Gao, and B. Dolan. deltableu: A discriminative metric for generation tasks with intrinsically diverse targets. *arXiv preprint arXiv:1506.06863*, 2015. 6

[5] Z. Gan, C. Gan, X. He, Y. Pu, K. Tran, J. Gao, L. Carin, and L. Deng. Semantic compositional networks for visual captioning. *arXiv preprint arXiv:1611.08002*, 2016. 2

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[7] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[8] Q. Jin, J. Chen, S. Chen, Y. Xiong, and A. Hauptmann. Describing videos using multi-modal fusion. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1087–1091. ACM, 2016. 7

[9] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, page 6, 2017. 2

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 3

[11] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 3675–3683, 2016. 2

[12] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015. 5

[13] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy. Improved image captioning via policy gradient optimization of spider. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2

[14] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014. 7

[15] P. Pan, Z. Xu, Y. Yang, F. Wu, and Y. Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1029–1038, 2016. 2

[16] Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui. Jointly modeling embedding and translation to bridge video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4594–4602, 2016. 1

[17] R. Pasunuru and M. Bansal. Multi-task video captioning with video and entailment generation. *arXiv preprint arXiv:1704.07489*, 2017. 2

[18] R. Pasunuru and M. Bansal. Reinforced video captioning with entailment rewards. *arXiv preprint arXiv:1708.02300*, 2017. 2

[19] B. Peng, X. Li, L. Li, J. Gao, A. Celikyilmaz, S. Lee, and K.-F. Wong. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2221–2230, 2017. 2

[20] V. Ramanishka, A. Das, D. H. Park, S. Venugopalan, L. A. Hendricks, M. Rohrbach, and K. Saenko. Multimodal video description. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1092–1096. ACM, 2016. 2, 7

[21] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015. 2, 5

[22] Z. Ren, X. Wang, N. Zhang, X. Lv, and L.-J. Li. Deep reinforcement learning-based image captioning with embedding reward. In *Proceeding of IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[23] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 6

[24] A. Rohrbach, M. Rohrbach, W. Qiu, A. Friedrich, M. Pinkal, and B. Schiele. Coherent multi-sentence video description with variable level of detail. In *German Conference on Pattern Recognition (GCPR)*, September 2014. Oral. 1

[25] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. 3

[26] Z. Shen, J. Li, Z. Su, M. Li, Y. Chen, Y.-G. Jiang, and X. Xue. Weakly supervised dense video captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 7

[27] R. Shetty and J. Laaksonen. Frame-and segment-level features and candidate pool evaluation for video caption generation. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1073–1076. ACM, 2016. 7

[28] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, 2016. 1, 2, 6

[29] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 387–395, 2014. 5

[30] J. Song, Z. Guo, L. Gao, W. Liu, D. Zhang, and H. T. Shen. Hierarchical lstm with adjusted temporal attention for video captioning. *arXiv preprint arXiv:1706.01231*, 2017. 2

[31] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. 5

[32] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 8

[33] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE international conference on computer vision*, pages 4534–4542, 2015. 2, 7

[34] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. In *NAACL HLT*, 2015. 7

[35] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1703.01161*, 2017. 2

[36] X. Wang, Y.-F. Wang, and W. Y. Wang. Watch, listen, and describe: Globally and locally aligned cross-modal attentions for video captioning. *NAACL HLT*, 2018. 2

[37] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 2, 4

[38] J. Xu, T. Mei, T. Yao, and Y. Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5288–5296, 2016. 1, 6

[39] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pages 4507–4515, 2015. 2, 7

[40] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593, 2016. 2

[41] Y. Yu, H. Ko, J. Choi, and G. Kim. Video captioning and retrieval models with semantic attention. *arXiv preprint arXiv:1610.02947*, 2016. 2

[42] W. Zaremba and I. Sutskever. Reinforcement learning neural turing machines. *arXiv preprint arXiv:1505.00521*, 419, 2015. 5

[43] L. Zhang, F. Sung, F. Liu, T. Xiang, S. Gong, Y. Yang, and T. M. Hospedales. Actor-critic sequence training for image captioning. In *NIPS Workshop on Visually-Grounded Interaction and Language*, 2017. 2