Language-driven Temporal Activity Localization: A Semantic Matching Reinforcement Learning Model

Weining Wang [1,3] Yan Huang[1,3] Liang Wang[1,2,3,4]
[1]Center for Research on Intelligent Perception and Computing (CRIPAC),
National Laboratory of Pattern Recognition (NLPR)
[2]Center for Excellence in Brain Science and Intelligence Technology (CEBSIT),
Institute of Automation, Chinese Academy of Sciences (CASIA)
[3]University of Chinese Academy of Sciences (UCAS)
[4]Artificial Intelligence Research, Chinese Academy of Sciences (CAS-AIR)
weining.wang@cripac.ia.ac.cn {yhuang, wangliang}@nlpr.ia.ac.cn

Abstract

Current studies on action detection in untrimmed videos are mostly designed for action classes, where an action is described at word level such as jumping, tumbling, swing, etc. This paper focuses on a rarely investigated problem of localizing an activity via a sentence query which would be more challenging and practical. Considering that current methods are generally time-consuming due to the dense frame-processing manner, we propose a recurrent neural network based reinforcement learning model which selectively observes a sequence of frames and associates the given sentence with video content in a matching-based manner. However, directly matching sentences with video content performs poorly due to the large visual-semantic discrepancy. Thus, we extend the method to a semantic matching reinforcement learning (SM-RL) model by extracting semantic concepts of videos and then fusing them with global context features. Extensive experiments on three benchmark datasets, TACoS, Charades-STA and DiDeMo, show that our method achieves the state-of-the-art performance with a high detection speed, demonstrating both effectiveness and efficiency of our method.

1. Introduction

With the rapid growth of video surveillance systems, a large amount of video data are generated. Understanding the content of video data is becoming increasingly important. Action recognition is one of the hottest topics in this area, which first obtains a representation of the given video, and then trains a classifier to categorize it into one of specified actions. However, action recognition usually assumes that videos are manually precut and each action ubiquitously exists in the given video. Such assumption does not always hold in real-world scenarios, because most videos in real applications are untrimmed and contain various background segments without any action, especially in video surveillance. To address this problem, temporal action detection has emerged, which aims to recognize and localize actions in videos simultaneously.

There has been growing interest in temporal action detection, and different approaches have been proposed [19, 27, 6, 9, 28, 25, 14, 2]. Although significant progress has been made, there is a major limitation of temporal action detection. In particular, these studies just focus on a limited set of actions described at word level. They can not properly handle activities in practice, because activities in real world are more complex and diverse which consist of many semantic concepts, such as actors, actions, objects, *etc*. Take the sentence "the person cuts oranges on a cutting board" in Figure 1 for an example, besides the action "cuts", the activity contains an actor "person", and two objects, namely "oranges" and "cutting board". It is inappropriate to describe this activity by a single word or simply categorize it into an action class.

In this paper, we are interested in a more challenging and practical problem, namely language-driven temporal activity localization. There are few studies that investigate this problem [11, 8, 15], which all utilize a traditional cross-modal retrieval framework to match video clips and sentence queries with an alignment loss or a ranking loss. However, they utilize sliding windows to generate dense proposals and every frame needs to be processed, which is prohibitively time-consuming. Besides, they use average pooling to generate video features at clip level, thus temporal information might not be fully exploited.

Motivated by the work using reinforcement learning to

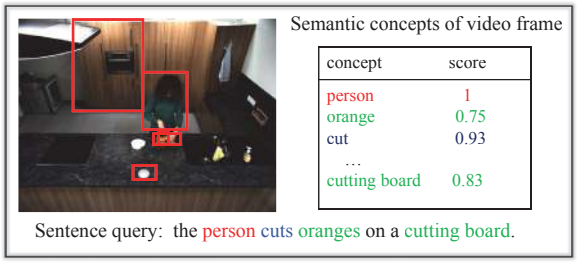

Figure 1. Illustration of semantic concepts in videos.

avoid time-consuming proposal generation in single-word video action detection [26], we also want to use a reinforcement learning based method for efficient language-driven temporal activity localization. However, directly applying the existing method to this new problem is infeasible, since they correlate the action word with video content in a classification way, while we focus on sentences rather than words, which include multiple words of action, actor and object. Although we could regard a sentence as multiple word-level classes and perform reinforcement learning in a multi-label learning way, this will ignore the intrinsic semantic order of the sentence and lead to confusion to the final semantic meanings [12]. In addition, the existing methods usually train an individual reinforcement learning model for each action class (totally 20 or 21 models), which cannot scale to our problem since words in the sentence can be diverse and the total number could be several thousands.

To deal with these issues, we propose a recurrent neural network based reinforcement learning model for language-driven temporal activity localization. Under the guidance of sentences, our model acts as a recurrent neural network based agent which dynamically observes a sequence of video frames and finally outputs the temporal boundaries of the given sentence query with a high detection speed. In particular, at each time step, the hidden state of the recurrent neural network is supervised by the sentence embedding to select the next observation location and output candidate detections. After observing several selected video frames, the model will output the final temporal boundaries of the activity. Different from separately training a model for each action class by classification [26], our model aims to associate the entire sentence with video content in a matching-based manner. In particular, we introduce a state value, namely a matching score, which measures the similarity of the given sentence query and current observed video frame. Here we use back-propagation to train the neural-network components and policy gradient to address the non-differentiablities due to the selecting manner.

However, we experimentally find that directly matching sentences with video content performs poorly, and the prediction of matching scores is not accurate, as well as the poor performance of activity localization. It is because there exists a huge visual-semantic gap between videos and sentence queries. As shown in Figure 1, the sentence contains highly abstract semantic concepts as actor, action, and object, while the representation of video usually lacks of such high-level semantic information. To make video representations semantically more comparable with sentences, we improve our method as a semantic matching reinforcement learning (SM-RL) model which improves video representation by introducing visual semantic concepts. To predict the semantic concepts of video frames, we exploit supervised learning methods based on the annotations of datasets. A fixed-length vector is created for each video frame, whose length is the size of the attribute set. Each element in the vector presents the prediction probability for a specific semantic concept. After applying semantic concept learning, the matching score becomes more accurate and reliable, and the final performance improves significantly. We evaluate our model on three benchmark datasets, TACoS, Charades-STA and DiDeMo. Experimental results show that our model outperforms the state-of-the-art method with high detection speed. The major contributions of this paper are summarized as follows:

- We propose a recurrent neural network based reinforcement learning model for language-driven temporal activity localization, which dynamically observes a sequence of video frames conditioned on the given language query and finally outputs temporal boundaries.
- To bridge the semantic gap between visual and semantic information, we further introduce mid-level semantic concepts into the model and we propose to correlate the visual and semantic information in a semantic matching manner.
- Our approach achieves the state-of-the-art performance on three benchmark datasets, and $6\times$ faster than the previous state-of-the-art work.

2. Related Work

We briefly review temporal action detection and temporal action proposal generation in this section.

Temporal Action Detection Temporal action detection aims to identify the start and end time as well as the action category for each action instance in a long untrimmed video, which has received significant attention. Approaches to video action detection in the literature can be roughly grouped into three categories. The first category is to employ temporal annotations to train the models in a supervised learning manner. Some of these works are in a two-stage proposal-classification manner [20, 7, 3, 19, 28], which first generate temporal video proposals and then classify the action categories for each proposal. However, most of these methods rely on external proposal generation or

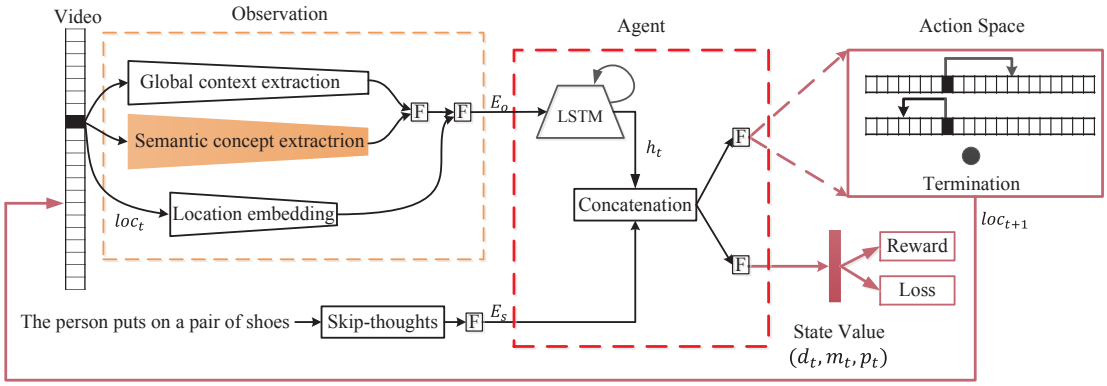

Figure 2. The framework of the proposed SM-RL model. In the forward pass, the sentence query is encoded by skip-thoughts [13] and further mapped into E_s. The global context feature of video frame is concatenated with the semantic concept feature. The location of current observed frame loc_t is embedded and concatenated with the video representation. The LSTM sequentially summarizes information from historically observed frames, and encodes temporal information of the video. The hidden state h_t is concatenated with E_s to output the action and state value. The action output is used to sample actions from the action space so as to select the next observation location loc_{t+1}. The state value consists of a candidate detection d_t, a matching score m_t, and a binary prediction indicator p_t. The reward and loss function are then calculated according to the state value. The agent takes the termination action to stop the observation process at time step *T*, and outputs final temporal boundaries according to the candidate detections. Rectangles with F denote fully connected layers.

sliding windows at multiple temporal scales, leading to computationally infeasible for large-scale video processing. Some other works follow the architecture of faster R-CNN which is end-to-end trainable, such as [9, 6, 10, 25, 5]. Although these methods are more efficient to some extent, temporal information is not fully explored in video features. The second category is based on weakly supervised learning, where only video-level action labels are available for training without temporal annotations. Untrimmed-Net [24] learns attention weights on precut video segments using a temporal softmax function and thresholds the attention weights to generate action proposals. Nguyen *et al.* [16] propose to combine temporal class activation maps and class agnostic attentions for temporal localization of target actions. The third category uses reinforcement learning to learn the observation policies. Yeung *et al.* [26] formulate the model as an agent that learns a policy for sequentially forming and refining hypotheses about action instances. Our detection procedure is similar to the third category mentioned above, but we focus on localizing sentence-describing activities rather than actions described at word-level.

Temporal Action Proposal Generation The goal of temporal action proposal generation is to extract semantically important (*e.g.*, human actions) segments from untrimmed videos. In [20, 28, 4], the problem is formulated as a binary classification problem (*i.e.*, action vs. background). Heilbron *et al.* [4] propose the use of dictionary learning for class independent proposal generation. Zhao *et al.* [28] use an actionness classifier to obtain binary actionness probabilities for video clips, and then find those continuous temporal regions with mostly high actionness snippets to serve as proposals. Shou *et al.* [20] introduce a multi-stage framework to classify if the content of a video segment is an action or not. Gao *et al.* [10] utilize temporal boundary regression for actions based on clip-level units. Shyamal *et al.* [3] exploit a new architecture SST to run over the video in a single pass, without the use of overlapping temporal sliding windows. Different from these works that extract arbitrary action segments from videos, we aim to locate an activity with specific sentence description.

3. Methodology

Given a long untrimmed video $v = \{v_1, v_2, ..., v_N\}$, where v_i $(i = 1, 2, ..., N)$ is the i-th frame, as well as a sentence query s, the goal of language-driven temporal activity localization is to identify temporal boundaries of the visual content that the sentence refers to, namely (t_{start}, t_{end}). Inspired by human's decision-making process, we formulate the model as an agent that interacts with the environment (*i.e.*, the long videos) and takes a series of actions to optimize the target (*i.e.*, localizing the activity). The historical experience is also explored to assist current decision. As shown in Figure 2, we base the agent on a recurrent neural network which summarizes historical observations from the input video and the sentence query.

To locate an activity, it is important to first understand the overall meaning of the sentence. In this paper, we use skip-thoughts [13] to encode the sentences, because skip-

thoughts is trained on a large corpus of documents, which can produce generic sentence representations that are robust and perform well in cross-modal tasks. In the observation module, global context feature is extracted to represent the global information of the image, while semantic concept feature is predicted to focus on the regional information and bridges the gap between videos and sentences. Thus, we concatenate the global context feature with the semantic concept feature for their complementary property. The location of the observed video frame plays an important role in the network, because the goal of this task is closely related to the temporal information of videos. Hence, we combine the location information with the video representation. Then, the final output of observation module E_o is input into the LSTM to integrate visual information.

A decision-making process in reinforcement learning consists of a series of actions. After each action, a state value is obtained and a reward is assigned to the agent according to current state value. In our framework, the agent combines information from historical observations h_t with the embedding of sentence query E_s to sample action from the action space and output three state values (a candidate detection d_t, a matching score m_t, and a binary prediction indicator p_t). Finally, the agent executes the sampled action to determine the location of next observation frame loc_{t+1}. The agent aims to find the most precise localization result by maximizing the accumulated reward. Thus, after observing all the selected frames, the agent receives a reward according to the precision of candidate detections.

3.1. Semantic Concept Learning

TACoS provides attribute annotations for each video clip including actions, vegetables, kitchen items, *etc*. We build the semantic concept vocabulary directly based on these annotations. Charades-STA and DiDeMo do not provide attribute information but only sentences. We use NLP toolbox to select nouns, adjectives, verbs and numbers as semantic concepts. Since the size of the semantic concept vocabulary is very large, we exclude the words with low use frequencies. Consequently, the vocabulary containing K semantic concepts is obtained. A straightforward way is to formulate semantic concept learning as a multi-label classification based model.

Multi-label Classification Based Model Taking the extracted video features from fc-7 of the VGG-16 [21] network as input, we train a CNN network for multi-label classification. The CNN we used is a 2-layer fully connected network. The size of the first layer is 4094. The size of the second layer is K, each corresponding to the prediction probability for a specific semantic concept. Given a video frame, its multi-hot representation of ground truth semantic concepts is $y_i \in \{0,1\}^K$, and the semantic concept vector predicted by the multi-label CNN is $\widetilde{y}_i \in [0,1]^K$. The model can be learned by optimizing the following objective:

$$L_{CNN} = \sum_{j=1}^{K} log(1 + e^{-y_{i,j}\widetilde{y}_{i,j}}) \quad (1)$$

where $y_{i,j} = 1$ or 0 means whether the video frame v_i contains the j-th semantic concept or not. $\widetilde{y}_{i,j}$ means the probability that v_i contains the j-th concept generated by the second layer of the multi-label CNN.

Faster R-CNN Based Model We find that the predicted semantic concepts are not satisfying on accuracy with the above multi-label classification bases model. We think that there might be two reasons. First, semantic concepts usually exist in local regions rather than global image. The global image includes many concept-irrelevant contents that could be quite noisy. Second, the used datasets are relatively small and the numbers of semantic concepts are not balanced, therefore the trained models are biased. To deal with these two issues, we use Faster R-CNN in conjunction with Visual Genome dataset [22] to predict the semantic concepts. In particular, to obtain regional visual features, we use Faster R-CNN to detect regions and output their corresponding features. To overcome the limitation of our experimental datasets, we train the model with Visual Genome dataset [22], which is a large dataset containing very diverse content that could cover most instances appeared in our experimental datasets. The dataset has very rich and region-level annotations for each image, so we can use them to train the model. Moreover, some recent works have demonstrated the usefulness of this dataset for general cross-modal data analysis, *e.g.*, image captioning [23] and VQA [1].

Thus, we follow [1] to use Faster R-CNN with ResNet-101 pretrained for classification on ImageNet [18] and then finetune the model on the Visual Genome [22] dataset. The output box proposals of the Faster R-CNN are used to generate a set of image features, and non-maximum suppression is applied for each object class using an IoU threshold. All regions where any class detection probability exceeds a confidence threshold are selected. In order to predict semantic concepts for region i, we concatenate the mean-pooled convolutional feature from region i with a learned embedding of ground-truth object class. We feed this into a softmax function over each semantic concept. In the testing stage, we can obtain semantic concepts for each video frame by summarizing the semantic concepts contained in all regions.

3.2. SM-RL Model

After obtaining the semantic concept features of video frames, we detail the proposed semantic matching reinforcement learning (SM-RL) model as illustrated in Figure 2. At each time step t, we use skip-thoughts [13] to encode the sentence query, where the output is further embedded to E_s with a fully connected (FC) layer followed

by a sigmoid function. The global context feature of video frame is extracted from fc7 of the VGG-16 network [21]. We then concatenate the semantic concept feature with the global context feature as the final content representation of the video frame, and it is further embedded by a FC layer followed by a sigmoid function. The location of current frame loc_t is normalized to $[0, 1]$ in a video sequence, and it is encoded with a FC layer followed by a sigmoid function. Then the video representation and location information are concatenated and further embedded, as the visual input to the LSTM, namely E_o. In particular, E_o encodes both the content of the video and the location of the content. The other input of the LSTM is the previous hidden state h_{t-1} which summarizes information from all the historically observed frames. We then concatenate the hidden state h_t with E_s to jointly output the action and state values.

State and Action Space At each step, the agent decides the action to execute according to current information. The action is to select the temporal location loc_{t+1} of the video frame that the agent chooses to observe next. This location is not constrained, and the agent may skip both forwards and backwards around a video. The location is formulated as $loc_{t+1} = f_l(h_t||E_s; \theta_l)$, where f_l is a fully connected layer, such that the agent's decision is a function of its past observations, their temporal locations and the sentence query. At training stage, the location is sampled from a Gaussian distribution, $loc_{t+1} \sim p(\cdot \mid f_l(h_t||E_s; \theta_l))$, where $f_l(h_t||E_s; \theta_l)$ represents the mean of the distribution, and the variance δ is fixed as a constant. At testing stage, the MAP estimate is used to infer the next observation location.

There are three state values at each time step: a candidate detection d_t, a matching score m_t, and a binary prediction indicator p_t to indicate whether d_t should be emitted as a prediction. The candidate detection d_t is $(t_{start}, t_{end}) \in [0, 1]^2$, where t_{start} and t_{end} are the normalized start and end time of the sentence query, m_t is the cross-modal matching score indicating the cross-modal similarity of the given sentence query and the observed video frame. t_{start}, t_{end}, m_t and p_t are calculated as:

$$(t_{start}, t_{end}) = f_{se}(h_t||E_s; \theta_d) \tag{2}$$

$$m_t = f_m(h_t||E_s; \theta_m) \tag{3}$$

$$p_t = f_p(h_t||E_s; \theta_p) \tag{4}$$

At the training stage, f_p is used to parameterize a Bernoulli distribution from which p_t is sampled. At the testing stage, the MAP estimate is used. Note that, f_l, f_{se} and f_m mentioned above are all designed as a fully connected layer followed by a sigmoid function.

Semantic Matching As illustrated above, we use m_t to indicate the cross-modal similarity of the given sentence query and the observed video frame, which plays an important role in localization. Since we want the agent to find more relevant video frame of the sentence query, m_t is intended to be closer to 1 if the observed video frame is related to the sentence query, and 0 otherwise. The training objective is then to minimize the standard cross-entropy loss:

$$L_{cls}(m_t; \theta_m) = -\sum_i (r_i) log P(r_i|m_i; \theta_m) \tag{5}$$

where r_i=1 or 0 denotes whether the i_{th} video frame is related to the sentence query.

3.3. Location Regression Loss

The goal of this work is to output temporal boundaries of the video clip related to the language query. To obtain more precise prediction, we train d_t and m_t at each time step using backpropagation.

Boundary Regression Loss With the temporal annotations of sentences, we are capable to train the candidate detection d_t using standard backpropagation. Since we want the candidate detection at each time step to get closer with the ground truth, the candidate detection at each time step is involved in the loss function no matter whether the candidate detection is emitted as a prediction. Suppose there is a set of candidate detections $D = \{d_t \mid t = 1, ..., T\}$ produced by the agent over T time steps, and the ground truth annotation is (g_{start}, g_{end}) for the sentence query. The loss function is defined as

$$L(D) = \omega_1 \sum_t L_{cls}(m_t) + \omega_2 \sum_t L_{loc}(d_t, (g_{start}, g_{end})) \tag{6}$$

The classification loss $L_{cls}(m_t)$ is a standard cross-entropy loss. The localization loss is defined as an L_2 regression loss, namely $L_{loc} = \|(t_{start}, t_{end}) - (g_{start}, g_{end})\|$. ω_1 and ω_2 are two hyper-parameters controlling the balance of these two loss functions.

Frame-level Regression Loss In Eq. (6), we directly regress the start time and end time with an L_2-form regression loss. In this section, we transfer the regression problem into a multi-label classification problem to verify whether each frame in the training sample is a relevant frame to the sentence query. After concatenation of h_t and E_s, we input the concatenated vector into a fully connected layer followed by a sigmoid function, where the number of output node is the number of frames in one training sample. We denote each output of the fully connected layer as p_{ij}, which indicates the probability that each frame belongs to the predicted video clip.

We try several loss functions, and find that the simple binary sigmoid cross-entropy loss works best. Therefore the location regression loss in Eq. (6) can be rewritten as

$$L_{loc} = \frac{1}{n} \sum_i^n \sum_j^M [x_{ij} log(p_{ij})] + (1 - x_{ij}) log(1 - p_{ij})] \tag{7}$$

where j is the j_{th} frame in a training sample, M is the total number of input frames, $x_i = [x_{i1}, x_{i2}, ..., x_{iM}]$ is the label vector of the i_{th} training sample. $x_{ij} = 1$ or 0 denotes whether the frame is related to the sentence query or not. p_{ij} denotes the probability that the frame is related to the query. At the testing stage, we regard those continuous temporal regions with mostly high probabilities as the detection results.

3.4. Reward

Because the prediction indicator p_t and observation location loc_{t+1} are non-differentiable which cannot be trained using backpropagation, we use reinforcement learning to solve them. Since we aim to find the most accurate location of an activity described by the sentence, the reward function should lead the agent to find a detection with high recall and high precision at the final time step. Thus, we formulate the reward function to encourage true positives, while depress false positives and false negatives:

$$r_T = \begin{cases} R_{FN}, & FN \\ N_{TP}R_{TP} + N_{FP}R_{FP}, & TP \ and \ FP \end{cases} \quad (8)$$

where FN (false negative) represents the model does not emit any prediction while there is a ground truth video clip in the video, and a negative reward R_{FN} is assigned to the agent. N_{TP} is the number of TP (true positive) predictions, where a prediction is emitted and the IoU between the prediction and the ground truth is larger than a threshold. R_{TP} is the positive reward assigned to the TP predictions. N_{FP} is the number of FP (false positive) predictions. There are two kinds of FP predictions. 1) The input video does not contain any relevant video clips to the sentence query but the model emits a prediction. 2) There exists a relevant video clip for the sentence query and the model indeed emits a prediction, but the IoU is smaller than the threshold. It should be noted that all reward is assigned at the T_{th} (final) time step, while the reward is zero in the middle time step, which leads the model to a high overall detection performance.

4. Experiments

4.1. Training Data

We evaluate our method on three benchmark datasets, TACoS [17], Charades-STA [8] and DiDeMo [11]. TACoS consists of 17,344 clip-sentence pairs. Following the same train/val/test split strategy as Gao *et al.* [8], we split the dataset in 50% for training, 25% for validation and 25% for testing. There are 19,509 clip-sentence pairs in the Charades-STA dataset. We split it as in [8] by dividing the dataset into 13,898 clip-sentence pairs in the training set, and 4,233 clip-sentence pairs in the testing set. DiDeMo dataset contains 10,464 videos with 40,543 clip-sentence pairs. We split the videos as [11], namely 8,395 for training, 1,065 for validation and 1,004 for testing.

We use sliding windows with a fixed scale to collect training video samples which are the input to our framework. For a sliding window clip, we align it as a positive training sample if it satisfies two constraints: 1) The IoU (intersection over union) of the sliding window clip and the ground truth temporal interval is larger than 0.5. 2) The nIoL (non intersection over length) of the sliding window clip and ground truth temporal interval is smaller than 0.2. We also collect negative samples which have no intersection with any sentence annotations.

The length of training samples of TACoS is fixed as 400 frames, the overlap of consecutive video clips is 40%. We further downsample the video clips by randomly selecting a single frame for every consecutive 16 frames. Therefore, the agent processes the video data in a sequence of 25 frames each time. For the Charades-STA dataset, the length of training samples is 252 frames, and the overlap of consecutive video clips is also set to 40%. We downsample the 252 frames to 21 frames, such that the agent processes the video data in a sequence of 21 frames. For DiDeMo dataset, we fix the training samples as 320 frames, and the overlap of consecutive video clips is also set to 40%. We then down sample the 320 frames to 20 frames.

4.2. Experimental Settings

The sizes of the semantic concept vocabularies for TACoS, Charades-STA and DiDeMo are 71, 60 and 85 respectively. In the Faster R-CNN based semantic learning model, the IoU threshold is set as 0.7 for region proposal suppression, and 0.3 for object class suppression. For the recurrent neural network, we use a 3-layer LSTM network with 1024 hidden units in each layer. The agent observes a fixed number of frames for each sequence, typically 6 in our experiments. We train the model with the batch size of 256, including 128 positive samples and 128 negative samples. The learning rate is assigned as 0.002. The standard variance δ of the Gaussian distribution is set to 0.08 for the observation location loc_{t+1} in the training stage. In Eq. (6), $w1$ and $w2$ are both set as 1. The hyper-parameters are determined by cross validation. We will analyze the effect of several important hyper-parameters in Section 4.4.

During testing, any candidate detections overlapping or crossing sequence bounds are merged with a simple union rule. For fair comparison, we adopt the same evaluation metric as [8] on TACoS and Charades-STA, which computes "$R@n$,IoU=m" meaning the percentage of at least one of the top-n results having IoU with the sentence annotation larger than m. Suppose there are N sentences in total, the overall performance is the average among all the sentences. $R(n,m) = \frac{1}{N}\sum_{i=1}^{N} r(n,m,s_i)$, where $r(n,m,s_i)$ is the recall for a sentence query s_i. For DiDeMo dataset,

Method	TACoS							Charades-STA				
	R@1 IoU=0.5	R@1 IoU=0.3	R@1 IoU=0.1	R@5 IoU=0.5	R@5 IoU=0.3	R@5 IoU=0.1	mR	R@1 IoU=0.5	R@1 IoU=0.7	R@5 IoU=0.5	R@5 IoU=0.7	mR
Random	0.83	1.81	3.28	3.57	7.03	15.09	5.27	8.51	3.03	37.12	14.06	15.68
ACRN [15]	14.62	19.52	24.22	24.88	34.97	47.42	27.61	-	-	-	-	-
CTRL [8]	13.30	18.32	24.32	25.42	36.69	48.73	27.80	23.63	8.89	58.92	29.52	30.24
RL(b)	11.76	17.70	22.42	22.61	33.24	45.10	25.47	19.78	5.60	55.65	25.07	26.53
RL(f)	12.79	18.53	23.87	24.56	35.30	47.64	27.15	21.18	7.33	56.01	27.85	28.09
SM-RL(attr+b)	13.50	18.83	23.72	24.01	34.19	46.56	26.80	21.00	7.63	57.25	28.06	28.49
SM-RL(attr+f)	14.01	19.02	23.96	24.55	36.42	47.14	27.51	22.54	8.56	58.95	29.74	29.95
SM-RL(attr*+b)	14.20	19.79	25.17	25.38	36.69	48.22	28.24	23.56	9.52	60.17	32.53	31.45
SM-RL(attr*+f)	**15.95**	**20.25**	**26.51**	**27.84**	**38.47**	**50.01**	**29.84**	**24.36**	**11.17**	**61.25**	**32.08**	**32.22**

Table 1. Comparison of different methods on TACoS and Charades-STA.

we measure the performance with $Rank@1$, $Rank@5$, and mean intersection over union ($mIoU$) as [11].

4.3. Experimental Results

As shown in Table 1, we report the results as $R@\{1, 5\}$ with $IoU \in \{0.1, 0.3, 0.5\}$ for the TACoS dataset, and $R@\{1, 5\}$ with $IoU \in \{0.5, 0.7\}$ for the Charades-STA dataset. We also compute the mean value mR of the above evaluation metrics. The top row means that we randomly select n windows from the test sliding windows and evaluate $R@n$ with IoU=m. The second and third row shows the experimental results of previous methods ACRN [15] and CTRL [8]. Rows 4-9 show the experimental results of different variants of our model. "RL" means that we directly use the recurrent neural network based reinforcement learning model. "SM-RL" means that we use the semantic matching reinforcement learning model which incorporates semantic concepts into the framework. "b" means that the location is regressed with the L_2-form loss function which directly regresses the boundaries, and "f" means that the model is trained with the frame-level regression loss. "attr" means that the semantic concept model is trained by the multi-label classification based model. "attr*" means that the semantic concepts are learned with the Faster R-CNN based model.

We can observe from Table 1 that the performance is relatively poor when directly applying "RL(b)" or "RL(f)" in this task, and the performance is lower than CTRL [8] and ACRN [15]. When we further enhance the model with semantic concepts learned by the multi-label classification model, namely "SM-RL(attr+b)" and "SM-RL(attr+f)", the performance becomes comparable with CTRL, where our method outperforms CTRL at some of the evaluation metrics such as $R@1$, IoU=0.5, $R@1$, IoU=0.3 on the TACoS dataset, and $R@5$, IoU=0.5, $R@5$, IoU=0.7 on the Charades-STA dataset. In particular, when the semantic concepts are learned with the Faster R-CNN based model, namely "SM-RL(attr*+b)" and "SM-RL(attr*+f)", the performance further increases and exceeds the state-of-the-art method. Models trained by frame-level regression loss consistently outperform models trained with L_2-form loss. In particular, the "SM-RL(attr*+f)" model performs best and outperforms the state-of-the-art method CTRL by up to 2.65 percent at $R@1$, IoU=0.5 on the TACoS dataset. The average performance mR exceeds CTRL on the TACoS dataset by 2.04 percent, and 1.98 percent on the Charades-STA dataset. We further compare our method with MCN [11] on DiDeMo dataset. As shown in Table 2, our methods consistently outperform MCN.

Method	Rank@1	Rank@5	mIoU
MCN [11]	28.10	78.21	41.08
SM-RL(attr*+b)	29.64	79.38	42.17
SM-RL(attr*+f)	**31.06**	**80.45**	**43.94**

Table 2. Comparison between our method and MCN on DiDeMo.

Method	Average running time (per minute video)
CTRL [8]	202ms
Ours	**32ms**

Table 3. Comparison of detection speeds.

In addition, the proposed model is very efficient using a fraction (less than 8%) of all the video frames. We compare the detection speed of our model with the state-of-the-art method CTRL [8]. Both of the two methods are tested on a single Titan X GPU. The comparison results are shown in Table 3. It can be seen from the table that our method is $6\times$ faster than CTRL, showing the potential capacity to be applied in real-world applications.

4.4. Ablation Study

4.4.1 Semantic Concept Learning Models

As illustrated in Table 1, semantic concepts significantly increase the performance. This is because that visual features are enhanced with mid-level semantic concepts. The semantic gap between visual information and language information is decreased to some extent. Besides, the Faster R-CNN based semantic concept learning model performs bet-

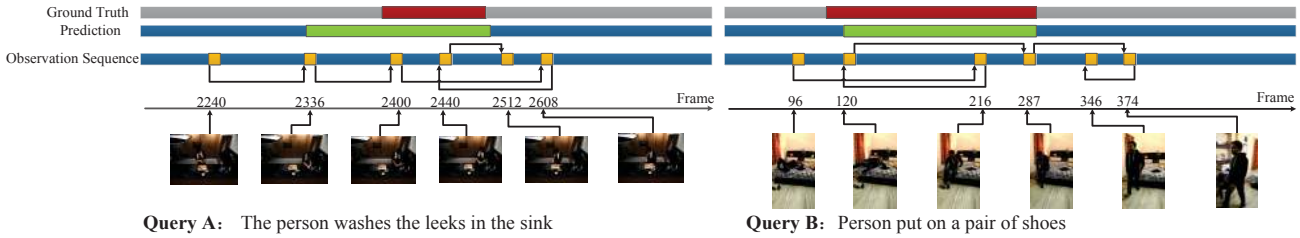

Figure 3. Examples of the observation policies and predictions by our proposed method.

ter than multi-label based semantic concept learning model in our experiments. This is because the Faster R-CNN can attend to more accurate regions of interest and the features are more fine-grained in comparison with the single multi-label classification model.

4.4.2 Comparison on Different Loss Functions

As shown in Table 1, the model trained with frame-level regression loss consistently outperforms the regular L_2-form regression loss. The possible reason is when training the model with frame-level regression loss the f_{se} in Eq. (2) is modeled as a fully connected layer with 25, 21 or 20 output nodes instead of just 2 nodes which directly regress the start point and end point. Accordingly the binary sigmoid cross-entropy loss forces the model to encode more information from the previous summarization of videos, leading to more accurate location estimation.

4.4.3 Hyper-parameters Analysis

When training the proposed model, we vary the number of observations, $w1$ and $w2$ in Eq. (6), and the standard variance δ of the Gaussian distribution of loc_t. The experiments are conducted with our SM-RL(attr*+f) model on the TACoS dataset. The corresponding results in Table 4 show that the accuracy dramatically decreases when the number of observations is set to 1. Nevertheless, when the value is larger than 4, the model is very robust. As shown in Table 5, the result achieves best when $\delta = 0.08$. As shown in Table 6, the result achieves best when $w1 = 1$ and $w2 = 1$.

Observation	1	2	4	6	8
R@1,IoU=0.5	4.26	8.45	15.58	**15.95**	15.20

Table 4. Performance comparison for the number of observations.

δ	0	0.04	0.08	0.12	0.16
R@1,IoU=0.5	9.57	9.36	**15.95**	10.16	8.93

Table 5. Performance comparison for parameter δ.

$w1$	1			2		3	
$w2$	1	2	3	1	3	1	2
R@1, IoU=0.5	**15.95**	9.48	7.52	7.33	12.45	8.36	13.12

Table 6. Performance comparison for $w1$ and $w2$.

4.4.4 Result Visualization

As shown in Figure 3, we illustrate two typical examples of observation policies that our model learns. The sentence query of the left example is "the person washes the leeks in the sink". Note that when the agent is near to the end time of the activity, it takes a step backward to refine its hypothesis. However, the prediction is a little longer than the ground truth, as the start location of this activity is hard to define. The model outputs the prediction from the time when the person puts the leeks in the sink, while the ground truth annotates this activity from the time when the person starts to wash it. The possible reason is that the image details are not well captured and the definition of an activity in real life is obscure. The right example illustrates the observation policy for the sentence query "person put on a pair of shoes". For this sentence query, the agent takes two step backward to refine its start time and end time. It is obvious that the prediction is much more accurate than the left one, because it is easier to verify the start and end positions for this activity.

5. Conclusion and Future Work

This paper has studied a rarely investigated and challenging problem, namely language-driven temporal activity localization. To deal with this problem, we have proposed a semantic reinforcement learning (SM-RL) model for temporal activity localization. The experimental results on three benchmark datasets have shown that our method outperforms the state-of-the-art method with a much higher speed. Currently we associate the hidden state of LSTM and sentence embedding with a simple concatenation. In the future, we would like to apply the gated fusion unit to associate the multi-modal data.

6. Acknowledgements

This work is jointly supported by National Key Research and Development Program of China (2016YFB1001000), National Natural Science Foundation of China (61525306, 61633021, 61721004, 61420106015, 61806194), Capital Science and Technology Leading Talent Training Project (Z181100006318030), and Beijing Science and Technology Project (Z181100008918010).

References

[1] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6077–6086, 2018.

[2] S Buch, V Escorcia, B Ghanem, L Fei-Fei, and JC Niebles. End-to-end, single-stream temporal action detection in untrimmed videos. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[3] Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6373–6382, 2017.

[4] Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1914–1923, 2016.

[5] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1130–1139, 2018.

[6] Xiyang Dai, Bharat Singh, Guyue Zhang, Larry S Davis, and Yan Qiu Chen. Temporal context network for activity localization in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5727–5736, 2017.

[7] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 768–784, 2016.

[8] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5277–5285, 2017.

[9] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Cascaded boundary regression for temporal action detection. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[10] Jiyang Gao, Zhenheng Yang, Chen Sun, Kan Chen, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3648–3656, 2017.

[11] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5803–5812, 2017.

[12] Yan Huang, Qi Wu, and Liang Wang. Learning semantic concepts and order for image and sentence matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6163–6171, 2018.

[13] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3294–3302, 2015.

[14] Tianwei Lin, Xu Zhao, and Zheng Shou. Single shot temporal action detection. In *Proceedings of the ACM on Multimedia Conference (ACM MM)*, pages 988–996, 2017.

[15] Meng Liu, Xiang Wang, Liqiang Nie, Xiangnan He, Baoquan Chen, and Tat-Seng Chua. Attentive moment retrieval in videos. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 15–24, 2018.

[16] Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6752–6761, 2018.

[17] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association of Computational Linguistics*, 1:25–36, 2013.

[18] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[19] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: convolutional-deconvolutional networks for precise temporal action localization in untrimmed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1417–1426, 2017.

[20] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1049–1058, 2016.

[21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2014.

[22] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[23] Subhashini Venugopalan, Lisa Anne Hendricks, Marcus Rohrbach, Raymond J. Mooney, Trevor Darrell, and Kate Saenko. Captioning images with diverse objects. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1170–1178, 2017.

[24] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 6402–6411, 2017.

[25] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In

Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 5794–5803, 2017.

[26] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2678–2687, 2016.

[27] Zehuan Yuan, Jonathan C Stroud, Tong Lu, and Jia Deng. Temporal action localization by structured maximal sums. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3215–3223, 2017.

[28] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2933–2942, 2017.