Supplementary Material for
Zero-Shot Natural Language Video Localization

1. Temporal Event Proposal

1.1. Details of Temporal Event Proposal

Atomic Event Discovery. To discover the atomic events from a video, we first extract frame-wise 3D CNN features of each video of Charades-STA and ActivityNet-Captions datasets by pre-trained I3D [5] and C3D models [46], respectively. We then uniformly sample 128 features from the frame-wise feature sequence to normalize the length of the video. With the 128 features, we build a similarity matrix with cosine similarity between each features of the video frames. As shown in the example similarity matrix in Fig.3-(a) of the main paper, clusters of events are clearly visible as diagonal squares on the similarity matrix. We then define each column vector of the similarity matrix concatenated with the frame index (a scalar value) as a *contextual feature* for each frame (L319 of main paper). The concatenated frame index is to encourage the cluster more temporally nearby. We cluster the contextual features using k-means algorithm to render the final *atomic events*. As a post-processing, we merge any clusters that has length shorter than 11 frames to its neighbor to remove too short clusters.

Composite Events. Once we have the atomic events, we generate composite events by populating all combinations of consecutive events, then sampling a few following a uniform distribution. We further investigate other choices of compositing the events (than the uniform sampling) in Sec. 1.3 (also some discussion in L327-L332 in the main paper). The procedure for temporal event proposal are illustrated in Fig.3-(a) of the main paper.

1.2. Hyper-parameter k for Atomic Event Clustering

We empirically choose the hyper-parameter k to be 5 among $\{3, 5, 7\}$ as summarized the results in Table. 1. When $k = 3$, we observe performance drops in all thresholds. We believe that small value of k makes the clustering overly merged so that the predicted regions are longer than expected. When $k = 7$, the performance marginally drops as most of clusters that can be resulted by $k = 5$ are similar to the ones by $k = 7$ but a number of overly fragmented atomic events are also proposed by the larger k that leads to learning the model with less precise regions.

k value	R@0.3	R@0.5	R@0.7	mIoU
3	45.12	27.45	13.21	30.16
5	**46.40**	**30.51**	**14.74**	**30.79**
7	45.71	29.53	13.29	30.31

Table 1: NLVL performance for various k values of k-means clustering

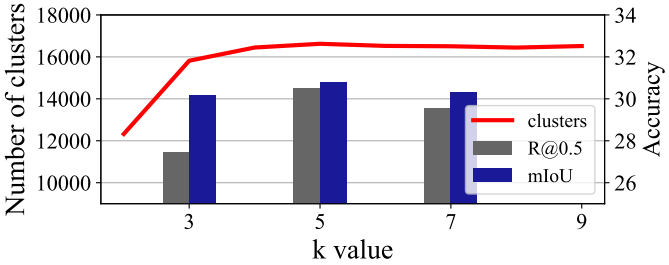

Figure 1: Change of the number of clusters and NLVL performance according to the hyper-parameter k values in Charades-STA dataset [19]. The red curve shows the number of clusters (left Y axis), and the bars represent the NLVL performance corresponding to the k values (right Y axis).

1.3. Scoring Functions for Composite Events

Since it is favorable to learn an NLVL model with regions with various semantics for better generalization, we consider three scoring functions in combining consecutive atomic events; atomic event's compactness, its diversity, and uniform ran-

dom sampling. We summarize the NLVL accuracy by different scoring functions (followed by the pseudo-query generation process) in Table 3 (bottom).

For both 'Compactness' and 'Diversity' methods, we compute the 'compactness' score of the proposed event regions and use it differently to compositing the atomic events. To compute the score, we average the difference between mean feature of the proposed event region and each feature in the event region. For the method of 'Compactness' in Table 3 (bottom), we choose the top-k event regions sorted by the highest compactness score. In contrast, for the 'Diversity' method, we chooses bottom-k event regions for high variety of the clusters. The 'Uniform sampling' method refers to our random sampling from the combinations of the consecutive atomic events.

Interestingly, the uniform random sampling performs the best. We believe that it contains both compact and diverse combinations of events thus leads to better coverage of training distribution that leads to better generalization.

1.4. Details about Temporal Event Proposal Baselines

In Table 3 (top), 'Random' is obtained by our NLVL model trained with randomly selected event regions followed by our pseudo-query generation method. The 'Sliding window' is obtained by a model trained with the windows obtained by 1) splitting videos into four equal length segments and 2) randomly choosing a combination from a set of combinations of consecutive segments [33]. Our method is the most similar to the sliding window except that we use atomic events to combine. The 'ActionBytes' is obtained by a model trained on temporal event proposals by [24]. They obtain the events by measuring the difference of each adjacent frame-wise 3D CNN features of the video, assuming that frame-wise CNN feature of a video changes abruptly at the event boundaries.

2. Pseudo-Query Generation

2.1. VerbBERT

To learn a model to predict the verbs, we fine-tune the pre-trained RoBERTa [34] model with the given language corpora to infer verbs with contextual nouns. We call it as **VerbBERT**. We briefly illustrate the training procedure of the VerbBERT in Fig. 2.

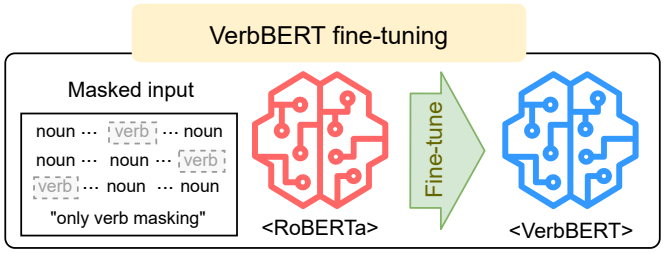

Figure 2: **Illustrative overview of the training procedure for VerbBERT**.

Data Collection, Preparation and Fine-Tuning. In order to fine-tune RoBERTa [34] to be the VerbBERT, we collect the sentences that contains 'person' from the given Flickr-description corpus [15]. Despite its small size, we use the 'Flickr' dataset because it is a corpus that describes person's action, which is characteristics for the benchmarks we used (*e.g.* Charades-STA, Activity-Net). We prepare training samples for VerbBERT by 1) tokenizing and POS-tagging the sentences using [22], 2) removing the words other than verbs and nouns, and 3) masking the verbs. For example, a sentence *"person is playing with the switch for the light."* would be "person [MASK] switch light" with the predicting verb of 'play.'

Using the prepared data, we fine tune the RoBERTa by masking the verbs, which is the task of masked language model (MLM). For the training details, we use Adam [3] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-6}$ and learning rate of 10^{-6} same as [34].

Inference. We use the following sentence template to predict the verb; "Person [MASK] [OBJECTS]." Here, the first word, "Person" provides context information for the 'human action' as most of the verbs are associated with person in many benchmark datasets [19,26]. But this heuristic can be changed to any other words when we address other domains. With this template, the model is expected to predict the verb with surrounding nouns.

In addition, the model is expected to predict the masked word that always positions in front of the object nouns. Since there may exist multiple probable verbs depending on various object combinations, we shuffle the position of object nouns in different permutations to predict verbs in different locations in a sentence.

2.2. Number of Nouns and Verbs in the Pseudo-Query

Different numbers of objects and verbs in the generated pseudo-query leads to a trade-off between the precision and recall of the pseudo-queries towards meaningful queries (*e.g.*, gold queries in the supervised data). If the number of words is large, it is likely to have correct signals (high recall), but it may have too much noisy signals from the incorrect words (low precision). In contrast, if we sample a small number of words, it is less likely to have correct signals (low recall), but if the verb is correct the information is not noisy (high precision).

To investigate the trade-off between quantity and quality of words in the pseudo-query, we vary the number of nouns and verbs, and summarize the NLVL performance in Table 2. For our final model, we use five nouns and three verbs (also for all other experiments). For the variations of number of nouns, we fix the number of verbs to be three. For the variations of number of verbs, we fix the number of nouns to be five. Comparing the result of other baselines ('{1,3,7} Nouns' and '{1,5} Verbs' of the table), we observe that it is important to find the proper number of nouns and actions (verbs) for the best performance.

	R@0.3	R@0.5	R@0.7	mIoU
1 Noun	39.91	18.05	6.60	25.61
3 Nouns	44.52	17.58	4.66	27.06
5 Nouns	46.47	31.29	14.17	31.24
7 Nouns	42.46	19.19	7.45	27.26
1 Verb	42.69	23.97	9.76	28.03
3 Verbs	46.47	31.29	14.17	31.24
5 Verbs	44.01	27.30	11.72	28.82
Ours (N:5,V:3)	**46.47**	**31.29**	**14.17**	**31.24**

Table 2: **NLVL performance when various number of nouns and verbs used in the pseudo-query.** 'k Nouns' and 'm Verbs' refer to the k nouns and m verbs in the pseudo-query (simplified sentence). Note that our final model uses 5 nouns and 3 verbs as they performs the best. For all 'k Nouns' entries, we use 3 verbs. For all 'm Verbs' entries, we use 5 nouns.

2.3. Details on Generated Noun Quality Analysis

Procedure for measuring noun overlaps. To compute matched word statistics, we utilize WordNet path distance after stemming and lemmatization to address language variation in matching. Specifically, we consider the set of words that differ less than 3 in WordNet path distance as a single word. With the threshold of ≤ 3, a popularly used value in NLP, *e.g.*, ('sneakers', 'shoe')$\rightarrow$(0.50), ('person','man')$\rightarrow$(0.33) are matched.

In our experiment, the number of nouns used in the original descriptions is 424 (α), and the number of the off-the-shelf object detector class is 1,600 (β). By this method, 192 out of 424 nouns are matched ($\alpha \cap \beta = 192$), therefore the overlap ratio (recall) is 45.28% ($(\alpha \cap \beta)/\alpha$=45.28%). Note that the overlap between the original nouns and object detector class (45.28%) is larger than the average overlap between ground truth nouns and detected nouns for each nouns (36.54%). One reason for latter being lower than former is the domain gap between the dataset that the off-the-shelf object detector is trained on and the target dataset.

NLVL performance as a function of the noun overlaps. We also measure the NLVL performance as a function of the number of overlapping objects between detected objects and original descriptions. Specifically, we measure NLVL performance by reducing the overlap ratio between detected nouns and nouns in each original sentence by removing the matched nouns. As the overlap decreases (36.54% $\rightarrow$ 27.48% $\rightarrow$ 17.97% $\rightarrow$ 9.64% $\rightarrow$ 1.15%), the NLVL's 'R@0.5' performance also decreases (31.88 $\rightarrow$ 31.09 $\rightarrow$ 28.25 $\rightarrow$ 25.94 $\rightarrow$ 23.82). This result shows the importance of the overlapping between detected nouns and original description's nouns. One should also note that for this experiment, we use the provided temporal region in the original supervision as it is the temporal region for the provided sentences. Thus, the 'R@0.5' result (31.88) is

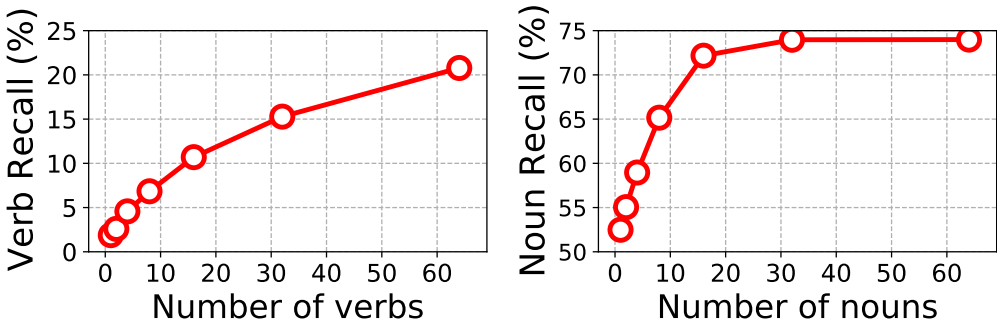

Figure 3: Matching accuracy of the ground truth query words (noun or verb) and the pseudo-query words. (left) Verb matching accuracy and (right) Noun matching accuracy. Note that verb is much more difficult to infer than the nouns.

marginally different from the one (31.29) in Table 2 in the main paper as we use the regions by the proposed 'temporal event proposal' there.

2.4. Quality of Pseudo Query

Furthermore, We analyze the quality of our pseudo-queries by calculating the matching accuracy between the ground truth query words and pseudo-query words generated in the ground truth temporal region as summarized in Fig. 3.

Verb matching accuracy is noticeably lower than the noun matching accuracy. It is expected as the objects are detected directly from the video but verbs are *inferred* with the objects from a text corpus. When we used top-5 objects and top-3 verbs, the matching accuracy are 60.74% and 3.58% for nouns and verbs, respectively. It implies that there is much to explore in this avenue.

3. Details of the Simple NLVL Model

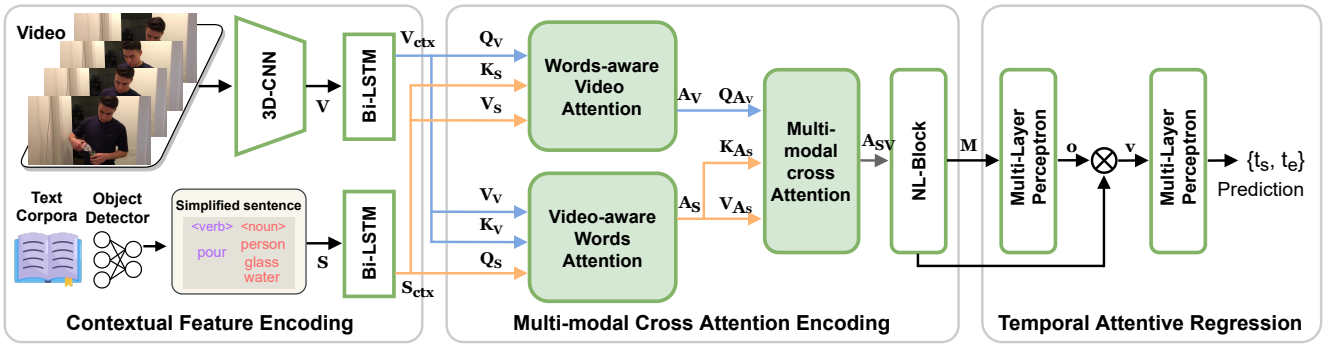

Figure 4: **Detailed architecture of the proposed simple NLVL model.** It encodes the simplified sentence and the proposed temporal event regions with cross modal attentions.

3.1. Architecture

We illustrate the NLVL model architecture in Fig. 4. The model is composed of three parts; contextual feature encoding, multi-modal cross attention and temporal attentive regression. Compared to the previous state-of-the-art models [39,44] for stronger supervision, our model highlights less on the sentence structures but focuses more on word-frame attention (L467-L471 of the main paper).

3.1.1 Contextual Feature Encoding

Simplified Sentence Query Encoding. The goal of simplified sentence query encoding is to produce a sentence feature with relational encoding. The input simplified sentence $\mathbf{S}$ with L words, is given as a sequence of words, expressed as

$\mathbf{S} = [w_1, \cdots, w_L] \in \mathbb{R}^{L \times d_w}$, where d_w is dimension of the word embedding vector. A bi-directional LSTM with a hidden state sized d_{hq} and a fully connected layer with the parameter $\mathbf{W_S}$ is applied to the input to obtain the sentence feature with relational encoding, $\mathbf{S_{ctx}}$, as:

$$\mathbf{S_{ctx}} = \text{ReLU}(\text{Bi-LSTM}(\mathbf{S})\mathbf{W_S}), \tag{1}$$

where $\mathbf{W_S} \in \mathbb{R}^{2d_{hq} \times d}$ is a learnable parameter and d_{hq} is dimension of the hidden state feature of the LSTM.

Video Encoding. Given the extracted video features from an individual video segment using 3D CNN [5,46], denoted by $\mathbf{V} = [v_1, v_2, \cdots, v_T] \in \mathbb{R}^{T \times d_v}$, where d_v is dimension of the video frame embedding vector (we use I3D [5] and C3D [46] as the 3D CNN for Charades-STA and ActivityNet-Captions, respectively), we apply bi-directional LSTM with hidden state size d_{hv} and a fully connected layer with parameter $\mathbf{W_V}$ as follows:

$$\mathbf{V_{ctx}} = \text{ReLU}((\mathbf{V}||\text{Bi-LSTM}(\mathbf{V}))\mathbf{W_V}), \tag{2}$$

where $\mathbf{V_{ctx}} \in \mathbb{R}^{T \times d}$ is the context encoded video feature and $\mathbf{W_V} \in \mathbb{R}^{(2d_{hv}+d_v) \times d}$ is a learnable parameter, where d_{hv} is dimension of the hidden state feature of the LSTM.

3.1.2 Multi-modal Cross Attention

The multi-modal cross attention module produces an attended feature $\mathbf{A_{SV}}$ on the two different modalities; $\mathbf{S_{ctx}}$ and $\mathbf{V_{ctx}}$. We utilize the attention mechanism proposed in [5] to learn the cross modal attended feature as follows:

$$\text{Attention}(Q, K, V) = \mathcal{S}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{3}$$

where Q, K and V are a query vector, a key vector and a value vector, respectively. The function $\mathcal{S}$ denotes a softmax.

Following [4], we use three multi-modal co-attention layers to obtain a combination of multi-modal features of word-aware video attention (WVA), video-aware word attention (VWA) and multi-modal cross attention (MCA). We first learn the word-aware video attention (WVA) $\mathbf{A_V} \in \mathbb{R}^{T \times d}$ and video-aware word attention (VWA) $\mathbf{A_S} \in \mathbb{R}^{L \times d}$ from $\mathbf{V_{ctx}}$ and $\mathbf{S_{ctx}}$ as:

$$\begin{aligned} \mathbf{A_V} &= \text{Attention}(\mathbf{V_{ctx}}, \mathbf{S_{ctx}}, \mathbf{S_{ctx}}), \\ \mathbf{A_S} &= \text{Attention}(\mathbf{S_{ctx}}, \mathbf{V_{ctx}}, \mathbf{V_{ctx}}). \end{aligned} \tag{4}$$

Then, we fuse $\mathbf{A_V}$ and $\mathbf{A_S}$ by a cross attention module to obtain a multi-modal cross attention (MCA) $\mathbf{A_{SV}} \in \mathbb{R}^{T \times d}$ of the video and query sentence as:

$$\mathbf{A_{SV}} = \text{Attention}(\mathbf{A_V}, \mathbf{A_S}, \mathbf{A_S}) \tag{5}$$

In addition to the three attentions, we further apply the non-local block (NL-Block) [52] to encode the global information over $\mathbf{A_{SV}}$. The NL-Block is popularly used for encoding global information over a feature [1][39]. We apply NL-Block to obtain $\mathbf{M} \in \mathbb{R}^{T \times d}$ from $\mathbf{A_{SV}}$ as:

$$\begin{aligned} \mathbf{M} &= \text{NL-Block}(\mathbf{A_{SV}}) \\ &= \mathbf{A_{SV}} + \mathcal{S}\left(\frac{(\mathbf{A_{SV}}\mathbf{W_{nq}})(\mathbf{A_{SV}}\mathbf{W_{nk}})^T}{\sqrt{d}}\right)(\mathbf{A_{SV}}\mathbf{W_{nv}}). \end{aligned} \tag{6}$$

3.1.3 Temporally Attentive Regression

Once we obtain the cross-modal attended feature of video and query sentence, we finally learn the starting point and ending point of the event. Following [39], we learn weights $\mathbf{W}_{ta1} \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_{ta2} \in \mathbb{R}^{d}$ in a multi layer perceptron (MLP) for the *temporal attention* $\mathbf{o} \in \mathbb{R}^T$ on $\mathbf{M}$ to regress the regions as follows:

$$\begin{aligned} \mathbf{o} &= \mathcal{S}(\tanh(\mathbf{M}\mathbf{W_{ta1}})\mathbf{W_{ta2}}), \\ \mathbf{v} &= \sum_{t=1}^{T} \mathbf{o}_t \mathbf{M}_t. \end{aligned} \tag{7}$$

where $\mathbf{v}$ is an temporally attended feature from $\mathbf{M}$.

From the attended feature $\mathbf{v}$, we finally obtain the temporal boundaries of event as:

$$\{\mathbf{t^s}, \mathbf{t^e}\} = \text{ReLU}(\mathbf{v}\mathbf{W_{tr1}})\mathbf{W_{tr2}}, \tag{8}$$

where $\mathbf{W_{tr1}} \in \mathbb{R}^{d\times d}$ and $\mathbf{W_{tr2}} \in \mathbb{R}^{d\times 2}$ are learned for final boundary prediction.

3.2. Objective Function

For $\mathcal{L}_{reg}$ in Eq.1 of the main paper, following [6], we use the Huber loss function [2] between the predicted timestamp $\{\hat{t^s}, \hat{t^e}\}$ and ground-truth timestamp $\{t^s, t^e\}$ as:

$$\mathcal{L}_{reg} = \text{Huber}(\hat{t^s} - t^s) + \text{Huber}(\hat{t^e} - t^e). \tag{9}$$

For $\mathcal{L}_{guide}$ in Eq.1 of the main paper, we use a temporal attention guidance loss proposed in [39,44] as:

$$\mathcal{L}_{guide} = -\frac{\sum_{i=1}^{T} \mathbb{1}_{\{t_{start} \le i \le t_{end}\}} \log(\mathbf{o}_i)}{\sum_{i=1}^{T} \mathbb{1}_{\{t_{start} \le i \le t_{end}\}}}, \tag{10}$$

where $\mathbb{1}(\cdot)$ is indicator function and set to 1 if the time segment is in the ground-truth region and otherwise 0, and $\mathbf{o}_i$ is temporal attentive weight for each temporal segment. This loss encourages the model to attend on the temporally segmented regions within the target.

4. Datasets and Setups

Charades-STA. It is built upon an action recognition dataset, Charades [19] by adding temporal event annotations and corresponding natural language descriptions. There are 6,672 videos and each are 30 seconds long on average. The videos are split into 5,338 and 1,334 videos for training and test sets, respectively. It has total 16,128 video-query pairs, with 12,408 for training split and 3,720 for test split. The descriptions are 6.21 words on average.

ActivityNet-Captions. As a subset of ActivityNet challenge dataset [26], ActivityNet-Captions is originally designed to evaluate dense video captioning methods. The dataset consists of 20,000 YouTube videos with an average length of 120 seconds. It is split into 10,024 training videos, 4,926 validation videos, and 5,044 test videos. The number of annotations are 37,421 and 34,536 for training and validation split, respectively. The annotations for the test split are privately kept for evaluations. The descriptions are composed of 13.48 words on average. Following the prior arts [39,44], we report the performance on the combined two validation sets of *val_1* and *val_2*.

5. More Qualitative Results

Fig. 5 shows more qualitative results by our model when the prediction is correct. We observe that the ground truth temporal segments, our predicted temporal segments, and the temporal attention weights are well aligned. This indicates that our model successfully attends to the temporal region that is related to the event, and correctly predicts the event regions using the attended features. Also, the attention weights on the words are generally higher in verbs such as *"eat"*, *"sit"* and *"open."* This implies that the verbs are more important than nouns for describing an event.

Fig. 6 shows more qualitative results when the prediction is incorrect. Even in the incorrect predictions, a fairly good amount of model outputs were similar to what is shown in Fig. 6-(a), *i.e.*, the predicted temporal attention includes ground truth event region as its sub-event. In these examples, our method is able to roughly locate the event, but fails to snap at the exact event region. On the other hand, Fig. 6-(c) shows an example that our model completely fails to predict the regions as it attends to the completely wrong temporal segments. In most of failed cases, our models output overlaps a small portion of the correct temporal segment as shown in Fig. 6-(b).

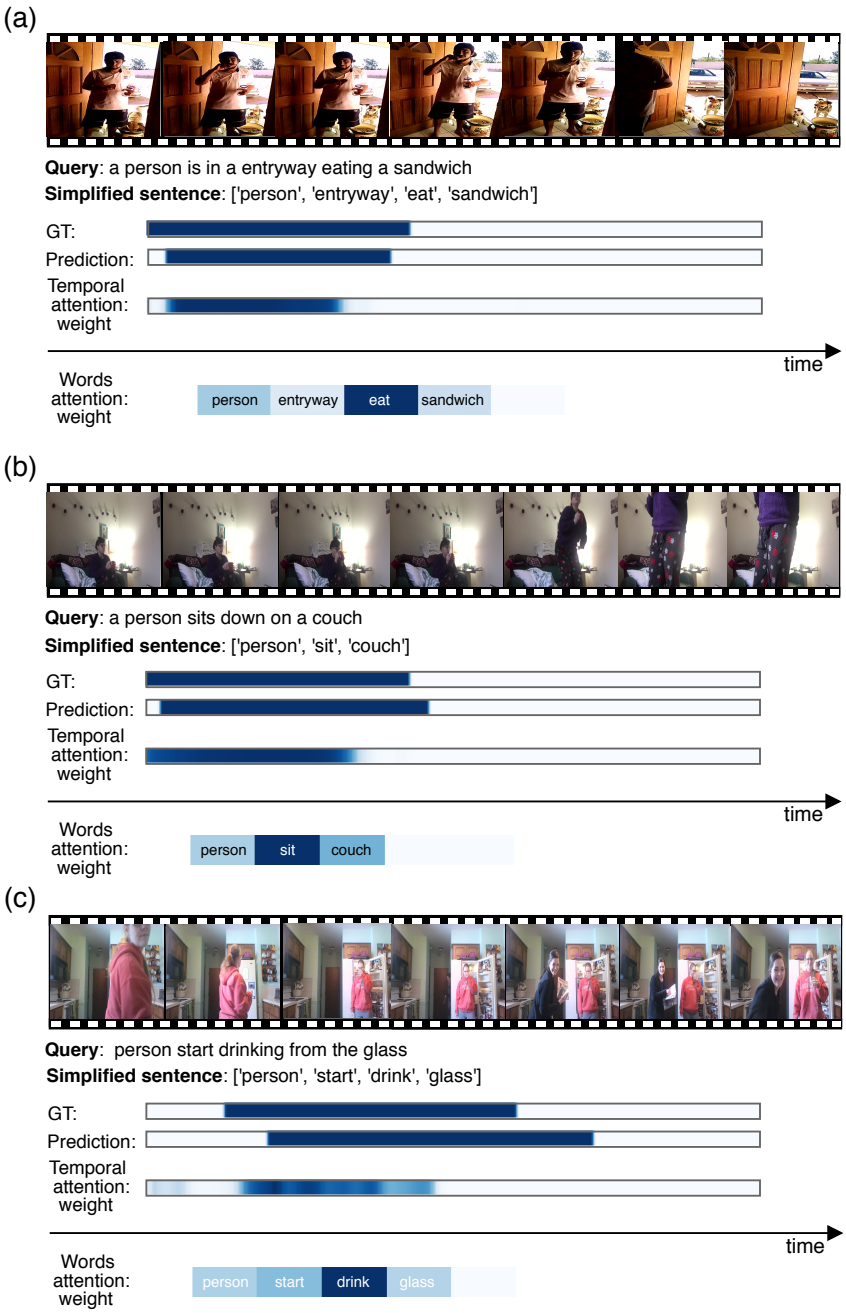

Figure 5: **Qualitative results when PSVL predicts correctly.** All of the ground truth temporal segments, our predicted temporal segments and the temporal attention weights are aligned well.

Figure 6: **Qualitative results when PSVL predicts incorrectly.** (a) Our prediction is a super-set of the ground truth temporal region. (b) The temporal attention intersects with ground truth temporal region. (c) both prediction and temporal attention is wrong.

References

[1] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 5

[2] Peter J Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, pages 73–101, 1964. 6

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[4] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019. 5

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 5

[6] Yitian Yuan, Tao Mei, and Wenwu Zhu. To find where you talk: Temporal sentence localization in video with attention based location regression. In *AAAI*, 2019. 6