Cascaded Prediction Network via Segment Tree for Temporal Video Grounding

Yang Zhao, Zhou Zhao*, Zhu Zhang, Zhijie Lin
College of Computer Science, Zhejiang University
{awalk, zhaozhou, zhangzhu, linzhijie}@zju.edu.cn

Abstract

Temporal video grounding aims to localize the target segment which is semantically aligned with the given sentence in an untrimmed video. Existing methods can be divided into two main categories, including proposal-based approaches and proposal-free approaches. However, the former ones suffer from the extra cost of generating proposals and inflexibility in determining fine-grained boundaries, and the latter ones usually attempt to decide the start and end timestamps directly, which brings about much difficulty and inaccuracy. In this paper, we convert this task into a multi-step decision problem and propose a novel Cascaded Prediction Network (CPN) to generate the grounding result in a coarse-to-fine manner. Concretely, we first encode video and query into the same latent space and fuse them into integrated representations. Afterwards, we construct a segment-tree-based structure and make predictions via decision navigation and signal decomposition in a cascaded way. We evaluate our proposed method on three large-scale publicly available benchmarks, namely ActivityNet Caption, Charades-STA and TACoS, where our CPN surpasses the performance of the state-of-the-art methods.

1. Introduction

With the rapid development of Internet technology, video has become a significant medium for information communication and dissemination, which brings great application value and prospect for the field of automatic video analysis. After the release of several large-scale datasets [3, 4, 13, 36, 38], research in this area is gradually moving towards video-text understanding tasks, including video-text retrieval [39, 43], video captioning [21, 33], video question answering [30, 49] and so forth. Considering the application scenarios in video websites and search engines, an increasing number of researchers begin to focus on the task of temporal video grounding.

*Zhou Zhao is the corresponding author.

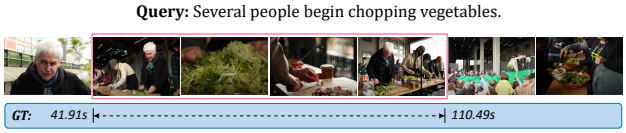


Figure 1. An illustration of temporal video grounding.

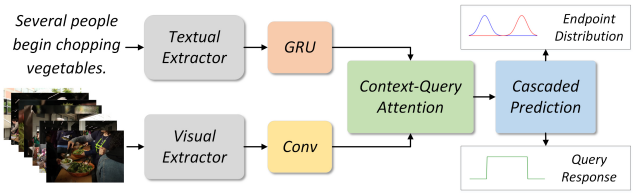


Figure 2. The overall perspective of our proposed model.

As the example in Figure 1 illustrates, temporal video grounding aims to automatically determine when an action or event corresponding to a given text query occurs in the video. The previous approaches in this area can be mainly categorized into two groups, including *proposal-based* methods [5, 9, 34, 37, 41, 44, 46, 48] and *proposal-free* methods [12, 20, 24, 35, 42, 45]. The former ones mainly follow the paradigm to manually predefine some proposals and select the best one by considering the correlation between proposal features and the given semantic information. And the latter ones try to tackle this problem by utilizing fully integrated features to determine the start and end timestamps aligned to the given description directly.

However, for the *proposal-based* methods, the handcrafted pre-definitions heavily rely on the prior knowledge to the length distribution of target segments for the specific datasets and bring much extra computational cost for preprocessing. Besides, the proposal boundaries are usually fixed, leading to the incapability to work out more flexible results. As for the *proposal-free* methods, the decision space for the final prediction is always too large for the model to generate accurate results in a single-shot classification or regression. Moreover, due to the lack of supervision from the inside of segments, these methods are strongly dependent on the expression ability of fusion modules.

To alleviate these problems, we devise a novel Cas-

caded Prediction Network (CPN) for temporal grounding as shown in Figure 2. Contrary to the existing approaches, we perform multiple cascaded prediction subtasks in a coarse-to-fine manner to generate fine-grained and flexible grounding results. Considering the effectiveness of segment tree in storing and representing segments of sequential data, we choose to use this data structure to maintain our CPN model, thus increasing computing speed and making the whole framework easy to maintain. Specifically, we first extract features from video and query and integrate them into fusion representations. Afterwards, we develop a segment-tree-based structure to generate segment features in different temporal scales and refine them in a message-passing way via graph neural network. Finally, we perform decision navigation and signal decomposition on each level to fully exploit the information from the boundary annotation and response signal associated with the sentence query.

Our main contributions can be summarized as follows:

- We consider the temporal video grounding task as a multi-step prediction problem and propose a novel Cascaded Prediction Network (CPN) based on a segment-tree structure to address this problem in a coarse-to-fine manner.
- We devise an effective representation learning method to generate discriminative segment features in different scales, thus enhancing the grounding performance.
- The extensive experiments conducted on three challenging public benchmarks, namely ActivityNet Caption, Charades-STA and TACoS, demonstrate the effectiveness of our proposed CPN method.

2. Related Work

Given an untrimmed video and a natural language query, temporal video grounding aims to locate the start and end timestamps of the video segment that best matches the given query. Initially, Gao *et al*. [9] first formulate this problem and try to address it in a *proposal-based* mechanism. Following this paradigm, Chen and Yuan *et al*. [5, 41] utilize various fine-grained multi-modal fusion methods to generate better integrated representations. Zhang *et al*. [48] try to furthur leverage the inner structure of video and query to improve the expressiveness of features. And Xu *et al*. [37] employ 3D Region of Interest Pooling to generate proposals instead of using sliding windows. Wang *et al*. [34] exploit the boundary score to modulate the selection and refinement of anchors. Moreover, Zhang and Zhang *et al*. [44, 46] both establish a proposal-oriented structure and explore the relations between proposals to enhance the performance.

Considering the extra computational cost stemming from generating proposal features, some *proposal-free* methods are proposed to tackle this problem. Among them, Rodriguez and Yuan *et al*. [24, 42] devise attention-based structures to generate predictions according to the relative affinity between two modalities. Hahn and Wang *et al*. [12, 35] follow the reinforcement-learning paradigm to drive the intelligent agent to glance over the video in a discontinuous way. Mun *et al*. [20] decompose the query sentence into multiple phrases and model the local and global context sequentially. And Zhang *et al*. [45] attempt to transform this problem into a span-based question answering task and solve it accordingly.

However, although these methods exhibit their great application values on large-scale datasets, they still suffer from the cost of collecting handcrafted annotations. Therefore, researchers begin to study this task under the weakly-supervised setting. The mainstream strategy in this area is to follow the multiple instance learning (MIL) paradigm, which is widely adopted by [6, 10, 19, 29]. Apart from this, Bojanowski *et al*. [2] consider this task as a cross-modal alignment problem and solve it via matrix optimization. Recently, Duan, Lin and Song *et al*. [8, 17, 28] also attempt to construct dual architectures to address this problem through caption generation or sentence reconstruction.

3. Preliminary

3.1. Problem Formulation

Given an untrimmed video $\mathbf{V}$ and an assigned natural language query $\mathbf{Q}$, temporal video grounding is to ascertain the moment $\hat{\boldsymbol{\tau}}$ that is most relevant to the given text query. More specifically, the input video can be denoted as $\mathbf{V} = \{\mathbf{v}_i\}_{i=1}^{n_v}$ where n_v is the frame number of video and $\mathbf{v}_i$ is the visual feature of the i-th frame, and the corresponding text query can be denoted as $\mathbf{Q} = \{\mathbf{q}_i\}_{i=1}^{n_q}$ where n_q is the sentence length of query and $\mathbf{q}_i$ is the textual feature of the i-th word. Under this notation definition, our task is to construct a proper model $\boldsymbol{\Omega}$ and find a set of parameters $\boldsymbol{\theta}$ so that the visual information within the temporal range

$$\hat{\boldsymbol{\tau}} = (\hat{\tau}_s, \hat{\tau}_e) = \boldsymbol{\Omega}(\mathbf{V}, \mathbf{Q}; \boldsymbol{\theta}), \text{ where } 1 \leq \hat{\tau}_s < \hat{\tau}_e \leq n_v \tag{1}$$

can represent the semantic information contained in the query most accurately.

3.2. Model Architecture and Features

The Overall Network Structure The overall architecture of our proposed model is illustrated in Figure 2. Concretely, we first employ a set of extractors and encoders to project the input video and query into the same latent space. Afterwards, we utilize a Context-Query Attention module to integrate textual and visual features into fused representations. Finally, our Cascaded Prediction Network (CPN) module is employed to generate and refine segment-level presentations and predict the temporal grounding result.

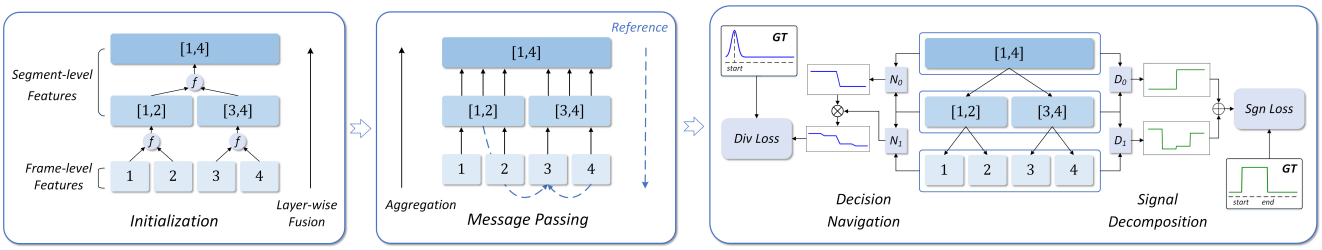


Figure 3. The concrete diagram of our proposed *Cascaded Prediction Network (CPN)*. For the sake of clarity, we take the case when $n_v = 4$ as an example to demonstrate the calculation process of this module.

Visual and Textual Representations Before generating more expressive representations, we first need to embed the given raw data into a continuous high-dimensional space. In our model, we employ the 300d Glove word2vec embedding [22] to extract textual features $\mathbf{Q}$ and pre-trained ConvNets to extract visual features $\mathbf{V}$ to keep consistent with other methods. And then, we utilize 1D convolutional layer and bi-directional GRU [7] as our generic choice of visual encoder and textual encoder to furthur encode these initial features into the same latent space. For the sake of clarity, we denote the refined visual and textual representations as $\tilde{\mathbf{V}} = \{\tilde{\mathbf{v}}_i\}_{i=1}^{n_v} \in \mathbb{R}^{n_v \times d}$ and $\tilde{\mathbf{Q}} = \{\tilde{\mathbf{q}}_i\}_{i=1}^{n_q} \in \mathbb{R}^{n_q \times d}$, respectively. It's worth noting that we perform a resampling operation on the given video to guarantee n_v is a power of 2 for the convenience of subsequent calculations.

Representation Fusion Following the standard strategy adopted in most reading comprehension models [26, 40], we utilize a Context-Query Attention (CQA) module to fuse the textual and visual representations. Specifically, we first calculate the cross-modal affinity matrix $\mathbf{A} \in \mathbb{R}^{n_v \times n_q}$ via an additive attention mechanism [1], given by

$$\mathbf{A}_{ij} = \mathbf{w}_s^\top(\tanh(\mathbf{W}_v\tilde{\mathbf{v}}_i + \mathbf{W}_q\tilde{\mathbf{q}}_j + \mathbf{b}_s)), \tag{2}$$

where $\mathbf{W}_v, \mathbf{W}_q \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_s, \mathbf{w}_s \in \mathbb{R}^d$ are all trainable parameters. And then, we normalize the affinity matrix through *SoftMax* calculation along the row and column axis to obtain the relative affinity intensity from one modality to the other, which can be denoted as $\mathbf{A}_r$ and $\mathbf{A}_c$ respectively.

Afterwards, the video-to-query attention $\mathcal{V} \in \mathbb{R}^{n_v \times d}$ and the query-to-video attention $\mathcal{Q} \in \mathbb{R}^{n_v \times d}$ can be calculated as

$$\mathcal{V} = \mathbf{A}_c \cdot \tilde{\mathbf{Q}}, \quad \mathcal{Q} = \mathbf{A}_c \cdot \mathbf{A}_r^T \cdot \tilde{\mathbf{V}} \tag{3}$$

And the final integrated representation $\bar{\mathbf{V}} \in \mathbb{R}^{n_v \times d}$ can be eventually given by

$$\bar{\mathbf{V}} = ([\tilde{\mathbf{V}}; \mathcal{V}; \tilde{\mathbf{V}} \odot \mathcal{V}; \tilde{\mathbf{V}} \odot \mathcal{Q}])\mathbf{W}_f + \mathbf{b}_f, \tag{4}$$

where $\mathbf{b}_f \in \mathbb{R}^d$ and $\mathbf{W}_f \in \mathbb{R}^{4d\times d}$ are all learnable parameters.

4. Cascaded Prediction Network

In this section, we will introduce our proposed *Cascaded Prediction Network (CPN)*.As shown in Figure 3, we construct a segment-tree-based structure to generate and refine the segment representations in different scales and make the final predictions in a cascaded manner. The whole prediction procedure can be divided into four stages, including *Tree Initialization*, *Message Passing*, *Decision Navigation* and *Signal Decomposition*.

4.1. Tree Initialization

Given the integrated frame-level features $\bar{\mathbf{V}}$, we first conduct a bottom-up fusion to generate segment-level representations and initialize the entire tree structure. Imitating the standard segment-tree construction process, the initialization of our CPN module can be formulated as follows.

We first denote the sequence of nodes contained in the h-th level as $\tilde{\mathbf{U}}^h = \{\tilde{\mathbf{u}}_i^h\}_{i=1}^{2^h}$, and the frame-level features serve as *leaf nodes* of the tree, i.e. $\tilde{\mathbf{U}}^H = \bar{\mathbf{V}}$. And then, the representations of $(h-1)$-th level can be given by

$$\tilde{\mathbf{u}}_i^{h-1} = f(\tilde{\mathbf{u}}_{2i-1}^h, \tilde{\mathbf{u}}_{2i}^h), \tag{5}$$

where $f(\cdot,\cdot)$ is the fusion module to aggregate the temporal adjacent features, which can be selected from 1D convolution, 1D max-pooling and cross gate module.

By performing this calculation recursively, we can work out the representations of all the other nodes in the tree, which are also named *branch nodes*. After doing so, we can obtain a full binary tree structure with $H = \log_2(n_v)$ levels and $2n_v - 1$ nodes in total as shown in Figure 3. In order to make the following descriptions clearer, we combine the node sequences of different levels and assign unified numbers to all nodes, given by $\mathbf{U} = \{\tilde{\mathbf{u}}_1^0, \tilde{\mathbf{u}}_1^1, \tilde{\mathbf{u}}_2^1, ..., \tilde{\mathbf{u}}_{n_v}^H\} = \{\mathbf{u}_i\}_{i=1}^{2n_v-1}$. Moreover, we define some special terms as the preliminary of the following descriptions.

- **Height** and **Order** The *Height* and *Order* of the i-th node can be expressed as $h_i = \lfloor log_2(i) \rfloor$ and $o_i = i - 2^{\lfloor log_2(i) \rfloor}$, which reflects the vertical and horizontal position of nodes.

- **Interval** The *Interval* corresponding to the i-th node is denoted as $\boldsymbol{\tau}_i = [s_i, e_i]$, and the start and end coordinates are given by $s_i = 1 + o_i 2^{h_i}$ and $e_i = (o_i + 1)2^{h_i}$, respectively.
- **Ancestor Node** and **Twin Node** The *Ancestor Node* of the i-th node with height difference of Δh is denoted as $a(i, \Delta h) = \lfloor \frac{i}{2^{\Delta h}} \rfloor$, and the *Twin Node* of the i-th node is given by $t(i) = 2a(i, 1) + 1 - i$, which points to the node that shares the same ancestor with height difference of 1.

4.2. Message Passing

Although we have obtained a series of fully integrated representations in different temporal scales, these features actually only capture the semantic information within the corresponding intervals, which leads to the insufficiency of comprehension to other parts of the video and makes it difficult to discriminate between target and the other segments.

A naive method to model context dependencies is to consider all pair-wise relations between nodes, resulting in a huge computational cost. Therefore, we attempt to prune redundant relations and devise a graph-based message passing mechanism to fuse context information efficiently. Concretely, given the node sequence $\mathbf{U}$, we establish two sets of edges between nodes, which are described as follows.

Reference Edge The *Reference Edge* allows information to flow from other nodes to the linked leaf node. In order to fuse context information at the least cost, we construct a *Reference Set* for every leaf node. The *Reference Set* is the smallest set in which the interval union of contained nodes can exactly cover the complement interval of the corresponding leaf node, given by

$$\mathcal{R}(i) = \{t(a(i, j))|j \in [0, h_i - 1]\}, \tag{6}$$

for the i-th node. And the set of *Reference Edge* can be formulated accordingly as below.

$$\mathbf{E}_R = \{j \to i | j \in \mathcal{R}(i), h_i = H\} \tag{7}$$

Aggregation Edge The *Aggregation Edge* is used to recollect and aggregate the information from child nodes to ancestors dynamically. Similar to the *Reference Set*, we also construct an *Aggregation Set* containing all the descendants of a branch node, which is given by

$$\mathcal{A}(i) = \{j | a(j, h) = i, \exists h \in [1, h_j]\}, \tag{8}$$

for the i-th node. And the set of *Aggregation Edge* can be formulated as

$$\mathbf{E}_A = \{j \to i | j \in \mathcal{A}(i), h_i < H\} \tag{9}$$

After the establishment of edges, we obtain a graph structure $\mathbf{G} = \{\mathbf{U}, (\mathbf{E}_R \cup \mathbf{E}_A)\}$. And we use the unified notation $\mathcal{N}(i)$ to denote the neighbor set of i-th node instead of $\mathcal{A}(i)$ or $\mathcal{R}(i)$ annotations. In this structure, any representation update that occurs in the tree (except on the root node) can be broadcast to any node in up to 3 steps. When it comes to the message passing procedure, actually any kind of off-the-shelf graph neural networks can be utilized to carry out this function, and we choose to employ a L-layer *DyResGEN* architecture proposed by Li *et al*. [16].

Moreover, considering the position information usually plays an essential role for sequence-related task, we generate a series of position-aware initial features $\mathbf{U}^0 = \{\mathbf{u}_i^0\}_{i=1}^{2n_v - 1}$ for message passing, given by

$$\mathbf{u}_i^0 = \mathbf{u}_i + [PE(h_i); PE(o_i)], \tag{10}$$

where $[;]$ is the concatenation operator, and $PE(\cdot)$ is the sinusoid position encoding function utilized in [32]. Consequently, the iterative calculation can be formulated as

$$\mathbf{u}_i^l = \text{MLP}(\mathbf{u}_i^{l-1} + \text{AGG}(\{\sigma(\mathbf{u}_j^{l-1}) + \epsilon | j \in \mathcal{N}(i)\})), \tag{11}$$

where σ is the activation function and the operator AGG is selected as *SoftMax Aggregation*.

4.3. Decision Navigation

After initializing and refining the tree structure, our CPN module can be used to decide the importance of nodes to a specified task in a navigation-based iterative way. In this task, we adopt this mechanism to determine the importance score of leaf nodes to the start and end of the target event, and predict the boundary timestamps accordingly.

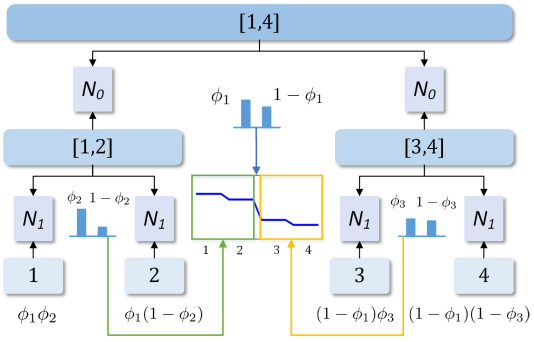


Figure 4. The detailed diagram of decision navigation.

Taking the start timestamp prediction as an example, we first define a series of events $A_{i,j} : \tau_s \in \boldsymbol{\tau}_{a(i,j)}$. Then the final prediction can be converted into a multi-step decision problem, given by

$$P(\tau_s \in \boldsymbol{\tau}_i) = \prod_{j=0}^{h_i - 1} P(A_{i,j} | A_{i,j+1}) P(A_{i,h_i}). \tag{12}$$

And this probability multiplication can be calculated by traversing the tree structure from the top down. As depicted in Figure 4, our decision navigation will start from the root node. When the decision proceeds to the i-th node, the h_i-th navigator will be applied to predict the probability of navigating left or right. Formally speaking, the probability of navigating to the left child at i-th node ϕ_i is given by

$$\phi_i^l = \mathbf{N}_{h_i}(\mathbf{u}_i^L, \mathbf{u}_{2i}^L), \quad \phi_i^r = \mathbf{N}_{h_i}(\mathbf{u}_i^L, \mathbf{u}_{2i+1}^L) \tag{13}$$

$$\phi_i = \frac{e^{\phi_i^l}}{e^{\phi_i^l} + e^{\phi_i^r}} \tag{14}$$

where $\mathbf{N}_i$ is the navigator of i-th level.

And we denote the cumulative probability of navigating to the i-th node as $\Phi_i = P(A_{i,0})$, then the recursive calculation formula can be written as

$$\Phi_{2i} = \phi_i \cdot \Phi_i, \quad \Phi_{2i+1} = (1 - \phi_i)\Phi_i, \tag{15}$$

Without loss of generality, we assign $\Phi_1 = 1$ to make the Equation 15 applicable to all branch nodes. Using mathematical induction, it's easy to prove that the cumulative results of each level conform to the definition of probability distribution. In this task, we just take the result of leaf nodes as our final boundary distribution prediction, which can be denoted as $\mathbf{P}_s$ and $\mathbf{P}_e$ for the prediction of start and end boundaries, respectively.

4.4. Signal Decomposition

Moreover, our proposed CPN module can also implement the function of signal decomposition as shown in Figure 5. Given a target signal, we can reconstruct this signal in different sampling frequencies via our cascaded structure and consequently generate a decomposition sequence. Similar to the *Decision Navigation* subtask, we also perform the signal decomposition in a top-down manner. Denoting the

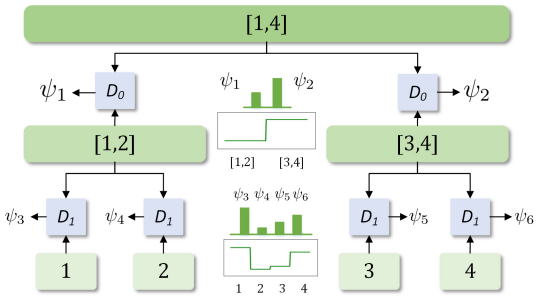


Figure 5. The detailed diagram of signal decomposition.

decomposition value of i-th node as ψ_i, this procedure can be formulated as below.

$$\psi_i^l = \sigma(\mathbf{D}_{h_i}(\mathbf{u}_i^L, \mathbf{u}_{2i}^L)), \quad \psi_i^r = \sigma(\mathbf{D}_{h_i}(\mathbf{u}_i^L, \mathbf{u}_{2i+1}^L)), \tag{16}$$

where the $\sigma(\cdot)$ is the *Sigmoid* function, and $\mathbf{D}_i$ is the decomposer of i-th level. And the cumulative decomposition result Ψ_i of i-th node is given by

$$\Psi_i = \alpha_0\psi_i + \alpha_1\psi_{a(i,1)} + \cdots = \sum_{j=0}^{h_i} \alpha_j \psi_{a(i,j)}, \tag{17}$$

where α_j is the coefficient of result with height difference of j. Considering that the decomposition result of higher level always have a lower sampling frequency and resolution, we manually assign α_j to be 2^{-j-1} and add a constant to ensure the magnitude of Ψ_i remains unchanged.

In our architecture, the navigators and decomposers are essentially multi-layer perceptron (MLP) modules and we share weights between the navigator and decomposer of the same level except the last linear layer in order to reduce the amount of parameters and further fuse the supervision information from boundary annotations and response signals.

4.5. Training and Inference

Based on the calculation formula and module structure mentioned previously, we apply a multi-task loss function to train our CPN network in an end-to-end manner. The final loss function is composed of two seperate parts, namely *Boundary Loss* and *Signal Loss*.

Boundary Loss Considering that the ground-truth boundary timestamps are given in a scalar form, we need to convert them into the corresponding distributions first. Formally speaking, we suppose the ground-truth boundary distributions can be formulated as $\hat{\mathbf{P}}_s \sim \mathcal{N}(\hat{\tau}_s, \sigma^2)$ and $\hat{\mathbf{P}}_e \sim \mathcal{N}(\hat{\tau}_e, \sigma^2)$ due to the uncertainty of data annotation, where $\mathcal{N}(\mu, \sigma^2)$ is the normal distribution with expectation of μ and standard deviation of σ. And under the assumption that a longer duration usually result in more blurred boundaries, we set σ as $(1 + \frac{\hat{\tau}_e - \hat{\tau}_s}{n_v})$ to control the smoothness of the distributions adaptively. Therefore, the loss function for boundary decision navigation is given by

$$\mathcal{L}_{div} = D_{\mathrm{KL}}(\mathbf{P}_s \parallel \hat{\mathbf{P}}_s) + D_{\mathrm{KL}}(\mathbf{P}_e \parallel \hat{\mathbf{P}}_e), \tag{18}$$

where $D_{\mathrm{KL}}(\mathbf{P}\|\mathbf{Q})$ is the Kullback-Leibler divergence from $\mathbf{Q}$ to $\mathbf{P}$.

Signal Loss Similar to *Boundary Loss*, we also need to generate the corresponding square wave manually for the ground-truth annotations. Concretely, the frame-wise sampling sequence of the target response signal $\hat{\mathbf{\Psi}} = \{\hat{\Psi}_i\}_{i=1}^{n_v}$ can be constructed by assigning the items within the ground-truth range to 1 and the others to 0. So the loss function for signal decomposition can be formulated as

$$\mathcal{L}_{sgn} = -\sum_{i=1}^{n_v}((1 - \hat{\Psi}_i)\log(1 - \Psi_i) + \hat{\Psi}_i \log \Psi_i), \tag{19}$$

Finally, the overall loss function in the training process can be summarized as

$$\mathcal{L} = \mathcal{L}_{div} + \lambda \mathcal{L}_{sgn}, \tag{20}$$

where λ is the hyper-parameter to balance these two parts.

While in the inference process, the start and end timestamps of grounding results are only determined by the navigation predictions, which are given by

$$\begin{aligned} \boldsymbol{\tau} = (\tau_s, \tau_e) = \arg\max_{(\tau_s, \tau_e)} (\mathbf{P}_s(\tau_s))(\mathbf{P}_e(\tau_e)), \\ \text{s.t. } 1 \leq \tau_s < \tau_e \leq n_v \end{aligned} \tag{21}$$

5. Experiments

5.1. Datasets

In order to validate the effectiveness of our proposed method, we conduct a series of experiments on ActivityNet Caption[15], Charades-STA[9] and TACoS[23].

ActivityNet Caption This dataset is generated by Krishna *et al*. from ActivityNet dataset [3] and contains about 20k various untrimmed videos of open-domain activities. We follow the split principle used in [47, 48], leading to 37,417, 17,505, 17,031 clip-sentence pairs used for training, validation and testing respectively.

Charades-STA This dataset is constructed by Gao *et al*. [9] from the original Charades dataset [27] and includes 9,848 videos of indoor activities. For model training and evaluation purpose, a total of 16,128 clip-sentence pairs can be furthur split into 12,408 and 3,720 ones as training and testing dataset respectively.

TACoS This dataset contains 127 videos collected from the MPII Cooking Composite Activities video corpus [25]. Taking the standard split used in [9] as a reference, the number of clip-sentence pairs in training, validation and testing dataset are 10,146, 4,589 and 4,083, respectively.

5.2. Implementation Details

Data Processing For ActivityNet Caption dataset, we utilize the same visual features as previous methods [46, 48], which are extracted via a publicly available pre-trained C3D model [31] and reduced to 500 dimensions using PCA. For Charades-STA and TACoS datasets, it's noteworthy that some newly-proposed state-of-the-art methods adopt different feature extractors due to the lack of unified feature extraction principle. To make a fair comparison, we get different features and annotations from the download link provided by other authors and compare with their proposed methods using the same features. Besides, in order to ensure the validity of our experiments, we fix all hyper-parameters of our model when conducting experiments on different features and annotations of the same dataset.

Model Setting The frame number of video n_v is set to 64, 32, 128 for ActivityNet Caption, Charades-STA and TACoS, respectively. And the layer number L of graph neural network is set to 4 for ActivityNet Caption and TACoS, and 2 for Chardes-STA. Besides, we adopt the multi-head mechanism proposed in [32] to improve the stability. Concretely, we represent visual and textual features into 2048 dimensions via encoders, split them into multiple chunks and take the average result over all chunks, in which the number of chunk is set to 16 for ActivityNet Caption and 8 for Charades-STA and TACoS. It's worth noting that this process doesn't increase the total amount of parameters in our CPN module. In the training phase, we employ Adam optimizer [14] with warmup strategy [32]. The learning rate is set to 0.001 for ActivityNet Caption and TACoS, and 0.0008 for Charades-STA. And the batch size is set to 64 for ActivityNet Caption and 32 for Charades-STA and TACoS.

5.3. Evaluation Metrics

Following the standard setting used in [9, 20, 45, 47], we adopt the "R@n, IoU= m" metric to evaluate the model performance automatically and objectively. This metric represents the ratio of language queries whose corresponding top-n grounding results have a maximum of IoU (i.e. Intersection over Union) being larger than m when compared with the ground-truth annotations. And we also use the "mIoU" metric (i.e. the mean average IoU over all results) to compare the overall performance.

5.4. Comparison with Other Methods

We compare our method with other existing state-of-the-art approaches proposed in recent years, which can be grouped into three categories as follows.

- **Proposal-based Methods** We compare our model with some works adopting this stategy, including CTRL [9], TGN [5], QSPN [37], CMIN [48] and 2D-TAN [47].
- **Reinforcement-learning-based Methods** We consider the following two RL-based methods, namely SM-RL [35] and TripNet [12].
- **Proposal-free Methods** Our proposed method can be also grouped into this category and will be compared with ABLR [42], PfTML-GA [24], LGI [20], VSLNet [45], DEBUG [18] and ExCL [11].

The overall evaluation results of our CPN and other methods on ActivityNet Caption, Charades-STA and TACoS datasets are presented in the Table 1, 2 and 3 respectively. The best results are given in **bold** and the second best ones are underlined in the tables. The experimental results reveal some notable points listed as follows.

- Compared with other approaches, our CPN method achieves superior performance on almost all criteria of

Table 1. Performance evaluation results on ActivityNet Caption ($n = 1$ and $m \in \{0.3, 0.5, 0.7\}$).

Method	IoU=0.3	IoU=0.5	IoU=0.7	mIoU
MCN	39.35	21.36	6.43	15.83
TGN	45.51	28.47	-	-
CTRL	47.43	29.01	10.34	20.54
TripNet	48.42	32.19	13.93	-
PfTML-GA	51.28	33.04	19.26	-
QSPN	52.13	33.26	13.43	-
ABLR	55.67	36.79	-	36.99
DEBUG	55.91	39.72	-	39.51
LGI	58.52	41.51	23.07	41.13
CMIN	**63.61**	43.40	23.88	-
VSLNet	63.16	43.22	26.16	43.19
2D-TAN	58.75	44.05	27.38	-
Ours	62.81	**45.10**	**28.10**	**45.70**

Table 2. Performance evaluation results on Charades-STA ($n = 1$ and $m \in \{0.3, 0.5, 0.7\}$).

Method	IoU=0.3	IoU=0.5	IoU=0.7	mIoU
2D ConvNet without fine-tuning as visual feature extractor				
CTRL	-	21.42	7.15	-
ABLR	-	24.36	9.01	-
SM-RL	-	24.36	11.17	-
TripNet	-	36.61	14.50	-
QSPN	54.70	35.60	15.80	-
DEBUG	-	37.69	17.69	36.34
MAN	-	41.24	20.54	-
2D-TAN$^\diamond$	-	42.80	23.25	-
Ours$^\diamond$	**64.41**	**46.08**	**25.06**	**43.90**
3D ConvNet without fine-tuning as visual feature extractor				
VSLNet*	64.30	47.31	30.19	45.15
Ours*	**68.48**	**51.07**	**31.54**	**48.09**
3D ConvNet with fine-tuning as visual feature extractor				
ExCL	65.10	44.10	23.30	-
PfTML-GA	-	52.02	33.74	-
VSLNet*	70.46	54.19	35.22	50.02
LGI$^\circ$	72.96	59.46	35.48	51.38
Ours*	72.94	56.70	36.62	51.85
Ours$^\circ$	**75.53**	**59.77**	**36.67**	**53.14**

$^\diamond$ The same data as *2D-TAN* are adopted.
* The same data as *VSLNet* are adopted.
$^\circ$ The same data as *LGI* are adopted.

these three datasets, which verifies the effectiveness of our proposed representation learning method and cascaded prediction mechanism.

- On the TACoS and Charades-STA datasets, our CPN outperforms all state-of-the-art methods which employ different feature extraction strategies. This observation suggests that our proposed model is applicable and robust to diverse features.
- By observing the evaluation results on Charades-STA dataset, we can find that the *fine-tuning* strategy is always helpful to boost the performance to a large extent. And 3D ConvNets are usually better choices for feature extraction since they can capture motion features and provide richer temporal information.
- On the TACoS dataset, our CPN gains a large margin compared with other methods, which may stem from the intrinsic characteristics of this dataset. In the videos collected from the original cooking-oriented dataset, there are only slight differences between adjacent frames, and the target segments might take up a small proportion of the total length, making other models confused and ineffective. And our cascaded prediction procedure can handle these two problems well.

Table 3. Performance evaluation results on TACoS ($n = 1$ and $m \in \{0.1, 0.3, 0.5, 0.7\}$).

Method	IoU=0.1	IoU=0.3	IoU=0.5	IoU=0.7	mIoU
MCN	14.42	-	5.58	-	-
ABLR	34.70	19.50	9.40	-	13.40
DEBUG	41.15	23.45	11.72	-	16.03
CTRL	24.32	18.32	13.30	-	-
QSPN	25.31	20.15	15.23	-	-
SM-RL	26.51	20.25	15.95	-	-
CMIN	32.48	24.64	18.05	-	-
TGN	41.87	21.77	18.90	-	-
TripNet	-	23.95	19.17	-	-
VSLNet*	-	29.61	24.27	20.03	24.11
2D-TAN$^\diamond$	47.59	37.29	25.32	-	-
Ours*	**61.24**	**48.29**	**36.58**	21.25	**34.63**
Ours$^\diamond$	60.54	47.69	36.33	**21.58**	34.49

$^\diamond$ The same data as *2D-TAN* are adopted.
* The same data as *VSLNet* are adopted.

5.5. Ablation Study

In this section, we conduct the ablation study for the concrete design and setting of our model.

Choice of initialization function We compare different fusion functions in the initialization process, including 1D convolution, 1D max-pooling and cross gate module. Figure 6 shows that the cross gate module always outperforms the others. We tentatively infer the reason is that pooling may blur the difference between adjacent frames thus impeding the model from making accurate predictions, and the cross gate module can furthur enhance interaction and encourage competition between neighbor frames.

Effect of signal loss In the training process, we assign the balance factor λ to 0, therefore the supervision information

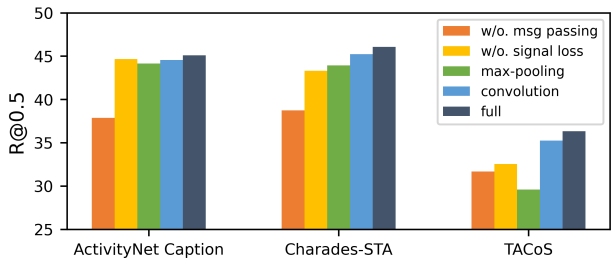


Figure 6. Ablation study of our proposed CPN method.

within the segment will not be exploited. From Figure 6, we can observe that the signal loss effectively provides enough supervision information from the inside of target segments and improve the capacity of representation learning.

Effect of message passing To verify the function of message passing, we remove this process and proceed directly with the initial tree structure. As shown in Figure 6, we find that it's quite significant to exploit context information and it shows the effectiveness of this mechanism.

5.6. Hyper-Parameter Analysis

In our model, the balance factor λ in the loss function is a significant hyper-parameter. Therefore, we furthur explore its effect to the model performance in this section. Specifically, we conduct multiple experiments by varying λ in the range of $[1, 9]$ and plot the evaluation results [1] in the Figure 7. We can clearly find that the optimum is obtained when λ is 6 for ActivityNet Caption and Charades-STA and 8 for TACoS. From this observation, we speculate the reason might be that too small λ would make the model incapable of extracting adequate information from the inside of target segments, while too large λ would cause the supervision from boundary annotations to be diluted, which makes the model confused and produce blurry or unstable results. Apart from this, some other hyper-parameter analysis can be referred to the supplementary materials.

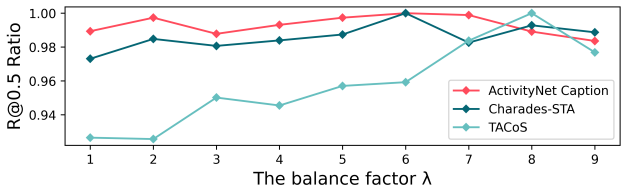
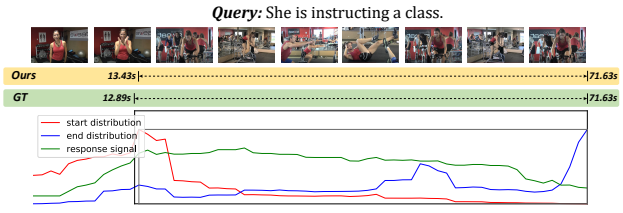


Figure 7. Effect of the balance factor λ in the multi-task loss on all three datasets.

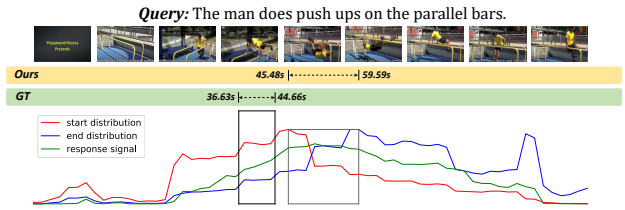
5.7. Qualitative Analysis

In order to qualitatively evaluate the performance of our CPN method, we show a success case and a failure case

[1]The values are normalized by dividing the optimal value.



(a) Success Case



(b) Failure Case

Figure 8. Qualitative Examples on the ActivityNet Caption dataset.

on ActivityNet Caption dataset, which can be found in the supplementary Figure 8. Each case presents predicted boundary distributions and response signals [2] along with the ground-truth annotation. In the success case, although there are lots of scene changes appearing in the video, our CPN method is still available to generate quite accurate predictions, which demonstrates the comprehensive analysis capability of our model. And looking closer into the the failure case, we can find that our CPN mistake *rope traverse* and *declined pull up* for *push up*. This may be because the model misidentifies different actions as variants of a single action under the change of perspective. The detailed analysis can be found in the supplementary materials.

6. Conclusion

In this paper, we propose a novel cascaded prediction network for temporal video grounding task. Our main idea is to split the original problem into a series of cascaded subtasks and solve them sequentially. Therefore, we devise a hierarchical representation learning method to produce effective integrated features and perform decision navigation and signal decomposition on each level to address this task. The extensive experiments on large-scale datasets demonstrate the effectiveness of our CPN method.

7. Acknowledgement

This work was supported in part by the National Key R&D Program of China (Grant No. 2018AAA0100603), National Natural Science Foundation of China under Grant No. 61836002 and No. 62072397, and Zhejiang Natural Science Foundation LR19F020006.

[2]The curves are scaled along the Y-axis.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 3

[2] Piotr Bojanowski, Rémi Lajugie, Edouard Grave, Francis Bach, Ivan Laptev, Jean Ponce, and Cordelia Schmid. Weakly-supervised alignment of video with text. In *Proceedings of the IEEE international conference on computer vision*, pages 4462–4470, 2015. 2

[3] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015. 1, 6

[4] David Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, 2011. 1

[5] Jingyuan Chen, Xinpeng Chen, Lin Ma, Zequn Jie, and Tat-Seng Chua. Temporally grounding natural sentence in video. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 162–171, 2018. 1, 2, 6

[6] Zhenfang Chen, Lin Ma, Wenhan Luo, Peng Tang, and Kwan-Yee K Wong. Look closer to ground better: Weakly-supervised temporal grounding of sentence in video. *arXiv preprint arXiv:2001.09308*, 2020. 2

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 3

[8] Xuguang Duan, Wenbing Huang, Chuang Gan, Jingdong Wang, Wenwu Zhu, and Junzhou Huang. Weakly supervised dense event captioning in videos. In *Advances in Neural Information Processing Systems*, pages 3059–3069, 2018. 2

[9] 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*, pages 5267–5275, 2017. 1, 2, 6

[10] Mingfei Gao, Larry Davis, Richard Socher, and Caiming Xiong. Wslln: Weakly supervised natural language localization networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1481–1487, 2019. 2

[11] Soham Ghosh, Anuva Agarwal, Zarana Parekh, and Alexander Hauptmann. Excl: Extractive clip localization using natural language descriptions. *arXiv preprint arXiv:1904.02755*, 2019. 6

[12] Meera Hahn, Asim Kadav, James M Rehg, and Hans Peter Graf. Tripping through time: Efficient localization of activities in videos. *arXiv preprint arXiv:1904.09936*, 2019. 1, 2, 6

[13] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2758–2766, 2017. 1

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[15] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017. 6

[16] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergcn: All you need to train deeper gcns. *arXiv preprint arXiv:2006.07739*, 2020. 4

[17] Zhijie Lin, Zhou Zhao, Zhu Zhang, Qi Wang, and Huasheng Liu. Weakly-supervised video moment retrieval via semantic completion network. *arXiv preprint arXiv:1911.08199*, 2019. 2

[18] Chujie Lu, Long Chen, Chilie Tan, Xiaolin Li, and Jun Xiao. Debug: A dense bottom-up grounding approach for natural language video localization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5147–5156, 2019. 6

[19] Niluthpol Chowdhury Mithun, Sujoy Paul, and Amit K Roy-Chowdhury. Weakly supervised video moment retrieval from text queries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11592–11601, 2019. 2

[20] Jonghwan Mun, Minsu Cho, and Bohyung Han. Local-global video-text interactions for temporal grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10810–10819, 2020. 1, 2, 6

[21] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting 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. 1

[22] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 3

[23] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics*, 1:25–36, 2013. 6

[24] Cristian Rodriguez, Edison Marrese-Taylor, Fatemeh Sadat Saleh, Hongdong Li, and Stephen Gould. Proposal-free temporal moment localization of a natural-language query in video using guided attention. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2464–2473, 2020. 1, 2, 6

[25] Marcus Rohrbach, Michaela Regneri, Mykhaylo Andriluka, Sikandar Amin, Manfred Pinkal, and Bernt Schiele. Script data for attribute-based recognition of composite activities.

In *European conference on computer vision*, pages 144–157. Springer, 2012. 6

[26] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016. 3

[27] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016. 6

[28] Yijun Song, Jingwen Wang, Lin Ma, Zhou Yu, and Jun Yu. Weakly-supervised multi-level attentional reconstruction network for grounding textual queries in videos. *arXiv preprint arXiv:2003.07048*, 2020. 2

[29] Reuben Tan, Huijuan Xu, Kate Saenko, and Bryan A Plummer. wman: Weakly-supervised moment alignment network for text-based video segment retrieval. *arXiv preprint arXiv:1909.13784*, 2019. 2

[30] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4631–4640, 2016. 1

[31] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 6

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4, 6

[33] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE international conference on computer vision*, pages 4534–4542, 2015. 1

[34] Jingwen Wang, Lin Ma, and Wenhao Jiang. Temporally grounding language queries in videos by contextual boundary-aware prediction. In *AAAI*, pages 12168–12175, 2020. 1, 2

[35] Weining Wang, Yan Huang, and Liang Wang. Language-driven temporal activity localization: A semantic matching reinforcement learning model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 334–343, 2019. 1, 2, 6

[36] Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1645–1653, 2017. 1

[37] Huijuan Xu, Kun He, Bryan A Plummer, Leonid Sigal, Stan Sclaroff, and Kate Saenko. Multilevel language and vision integration for text-to-clip retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9062–9069, 2019. 1, 2, 6

[38] Jun Xu, Tao Mei, Ting Yao, and Yong 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

[39] Ran Xu, Caiming Xiong, Wei Chen, and Jason J Corso. Jointly modeling deep video and compositional text to bridge vision and language in a unified framework. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 1

[40] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*, 2018. 3

[41] Yitian Yuan, Lin Ma, Jingwen Wang, Wei Liu, and Wenwu Zhu. Semantic conditioned dynamic modulation for temporal sentence grounding in videos. In *Advances in Neural Information Processing Systems*, pages 536–546, 2019. 1, 2

[42] Yitian Yuan, Tao Mei, and Wenwu Zhu. To find where you talk: Temporal sentence localization in video with attention based location regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9159–9166, 2019. 1, 2, 6

[43] Bowen Zhang, Hexiang Hu, and Fei Sha. Cross-modal and hierarchical modeling of video and text. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 374–390, 2018. 1

[44] Da Zhang, Xiyang Dai, Xin Wang, Yuan-Fang Wang, and Larry S Davis. Man: Moment alignment network for natural language moment retrieval via iterative graph adjustment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1247–1257, 2019. 1, 2

[45] Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. Span-based localizing network for natural language video localization. *arXiv preprint arXiv:2004.13931*, 2020. 1, 2, 6

[46] Songyang Zhang, Houwen Peng, Jianlong Fu, and Jiebo Luo. Learning 2d temporal adjacent networks for moment localization with natural language. *arXiv preprint arXiv:1912.03590*, 2019. 1, 2, 6

[47] Songyang Zhang, Houwen Peng, Jianlong Fu, and Jiebo Luo. Learning 2d temporal adjacent networks for moment localization with natural language. *arXiv preprint arXiv:1912.03590*, 2019. 6

[48] Zhu Zhang, Zhijie Lin, Zhou Zhao, and Zhenxin Xiao. Cross-modal interaction networks for query-based moment retrieval in videos. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 655–664, 2019. 1, 2, 6

[49] Zhu Zhang, Zhou Zhao, Zhijie Lin, Jingkuan Song, and Xiaofei He. Open-ended long-form video question answering via hierarchical convolutional self-attention networks. *arXiv preprint arXiv:1906.12158*, 2019. 1