# TVR: A Large-Scale Dataset for Video-Subtitle Moment Retrieval

Jie Lei, Licheng Yu, Tamara L. Berg, and Mohit Bansal

University of North Carolina at Chapel Hill
{jielei, licheng, tlberg, mbansal}@cs.unc.edu

**Abstract.** We introduce TV show Retrieval (**TVR**), a new multimodal retrieval dataset. TVR requires systems to understand both videos and their associated subtitle (dialogue) texts, making it more realistic. The dataset contains 109K queries collected on 21.8K videos from 6 TV shows of diverse genres, where each query is associated with a tight temporal window. The queries are also labeled with query types that indicate whether each of them is more related to video or subtitle or both, allowing for in-depth analysis of the dataset and the methods that built on top of it. Strict qualification and post-annotation verification tests are applied to ensure the quality of the collected data. Further, we present several baselines and a novel Cross-modal Moment Localization (**XML**) network for multimodal moment retrieval tasks. The proposed XML model uses a late fusion design with a novel Convolutional Start-End detector (**ConvSE**), surpassing baselines by a large margin and with better efficiency, providing a strong starting point for future work. We have also collected additional descriptions for each annotated moment in TVR to form a new multimodal captioning dataset with 262K captions, named TV show Caption (**TVC**).[1]

## 1 Introduction

Enormous numbers of multimodal videos (with audio and/or text) are being uploaded to the web every day. To enable users to search through these videos and find relevant moments, an efficient and accurate method for retrieval of video data is crucial. Recent works [14,9] introduced the task of Single Video Moment Retrieval (SVMR), whose goal is to retrieve a moment from a single video via a natural language query. Escorcia *et al.* [8] extended SVMR to Video Corpus Moment Retrieval (VCMR), where a system is required to retrieve the most relevant moments from a large video corpus instead of from a single video. However, these works rely on a single modality (visual) as the context source for retrieval, as existing moment retrieval datasets [14,32,9,22] are based on videos. In practice, videos are often associated with other modalities such as audio or text, e.g., subtitles for movie/TV-shows or audience discourse accompanying live

[1] Published in ECCV 2020. Both datasets are publicly available. TVR: `https://tvr.cs.unc.edu`, TVC: `https://tvr.cs.unc.edu/tvc.html`.

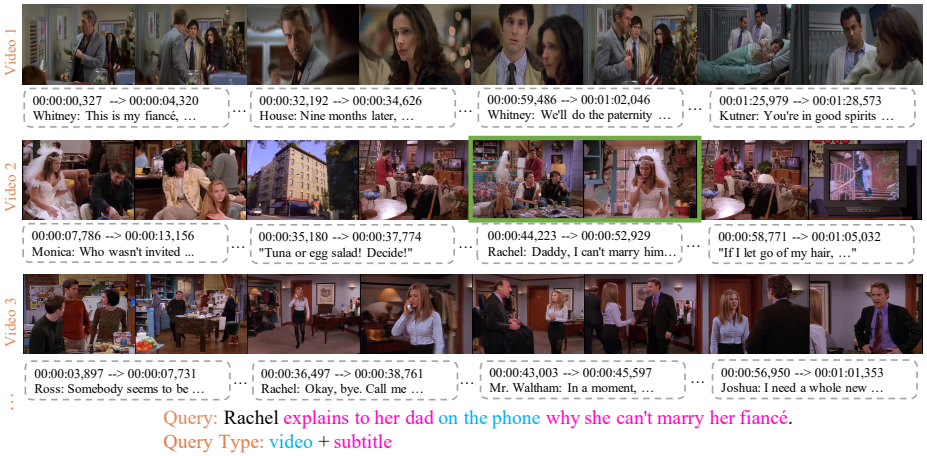


Fig. 1: A TVR example in the VCMR task. Ground truth moment is shown in *green box*. Colors in the query indicate whether the words are related to video (*blue*) or subtitle (*magenta*) or both (*black*). To better retrieve relevant moments from the video corpus, a system needs to comprehend both videos and subtitles

stream videos. These associated modalities could be equally important sources for retrieving user-relevant moments. Fig. 1 shows a query example in the VCMR task, in which both videos and subtitles are vital to the retrieval process.

Hence, to study multimodal moment retrieval with both video and text contexts, we propose a new dataset - TV show Retrieval (**TVR**). Inspired by recent works [39,21,24] that built multimodal datasets based on Movie/Cartoon/TV shows, we select TV shows as our data resource as they typically involve rich social interactions between actors, involving both activities and dialogues. During data collection, we present annotators with videos and associated subtitles to encourage them to write multimodal queries. A tight temporal timestamp is labeled for each video-query pair. We do not use predefined fixed segments (as in [14]) but choose to freely annotate the timestamps for more accurate localization. Moreover, query types are collected for each query to indicate whether it is more related to the video, the subtitle, or both, allowing deeper analyses of systems. To ensure data quality, we set up strict qualification and post-annotation quality verification tests. In total, we have collected 108,965 high-quality queries on 21,793 videos from 6 TV shows, producing the largest dataset of this kind. Compared to existing datasets [14,32,9,22], we show TVR has greater linguistic diversity (Fig. 3) and involves more actions and people in its queries (Table 2).

With the TVR dataset, we extend the moment retrieval task to a more realistic multimodal setup where both video and subtitle text need to be considered (i.e., 'Video-Subtitle Moment Retrieval'). In this paper, we focus on the corpus-level task VCMR , as SVMR can be viewed as a simplified version of VCMR in which the ground-truth video is given beforehand. Prior works [14,9,15,43,10,8] explore the moment retrieval task as a ranking problem over a predefined set

of moment proposals. These proposals are usually generated using handcrafted heuristics [14,15] or sliding windows [9,43,10,8] and are usually not temporally precise, leading to suboptimal performance. Furthermore, these methods may not be easily scaled to long videos: the number of proposals often increase quadratically with video length, making computational costs infeasible. Recent methods [11,25] adapt start-end span predictors [36,3] from the reading comprehension task to moment retrieval, by early fusion of video and language (query) features, then applying neural networks on the fused features to predict start-end probabilities. It has been shown [11] that using span predictors outperforms several proposal-based methods. Additionally, start-end predictors allow a hassle-free extension to long videos, with only linearly increased computational cost. While [11] has shown promising results in SVMR, it is not scalable to VCMR as it uses expensive early fusion operation. Consider retrieving $N$ queries in a corpus of $M$ videos, the approach in [11] requires running several layers of LSTM [17] on $M \cdot N$ early fused representations to generate the probabilities, which is computationally expensive for large values of $M$ and $N$.

To address these challenges, we propose Cross-modal Moment Localization (**XML**), a late fusion approach for VCMR. In XML, videos (or subtitles) and queries are encoded independently, thus only $M+N$ neural network operations are needed. Furthermore, videos can be pre-encoded and stored. At test time, one only needs to encode new user queries, which greatly reduces user waiting time. Late fusion then integrates video and query representations with highly optimized matrix multiplication to generate 1D query-clip similarity scores over the temporal dimension of the videos. To produce moment predictions from these similarity scores, a naive approach is to rank the aforementioned sliding window proposals with confidence scores computed as the average of the similarity scores inside each proposal region. Alternatively, one can use TAG [50] to progressively group top-scored clips. However, these methods rely on handcrafted rules and are not end-to-end trainable. Inspired by image edge detectors [38] in image processing, we propose Convolutional Start-End detector (**ConvSE**) that learns to detect start (up) and end (down) edges in the similarity signals with two trainable 1D convolution filters. Using the same backbone net, we show ConvSE has better performance than both approaches. With late fusion and ConvSE, we further show XML outperforms previous methods [14,8,11], and does this with better computational efficiency.

To summarize, our contributions are three-fold: (*i*) We introduce **TVR** dataset, a large-scale multimodal moment retrieval dataset with 109K high-quality queries of great linguistic diversity. (*ii*) We propose **XML**, an efficient approach that uses a late fusion design for the VCMR task. The core of XML is our novel **ConvSE** module which learns to detect start-end edges in 1D similarity signals with 2 convolution filters. Comprehensive experiments and analyses show XML surpasses all presented baselines by a large margin and runs with better efficiency. (*iii*) We have also collected additional descriptions for each annotated moment in TVR to form a new multimodal captioning dataset with 262K captions, named TV show Caption (**TVC**).

## 2 Related Work

The goal of natural language-based moment retrieval is to retrieve relevant moments from a single video [14,9] or from a large video corpus [8]. In the following, we present a brief overview of the community efforts on these tasks and make distinctions between existing works and ours.

**Datasets**. Several datasets have been proposed for the task, e.g., DiDeMo [14], ActivityNet Captions [22], CharadesSTA [9], and TACoS [32], where queries can be localized solely from video. TVR differs from them by requiring additional text (subtitle) information in localizing the queries. Two types of data annotation have been explored in previous works: (*i*) uniformly chunking videos into segments and letting an annotator pick one (or more) and write an unambiguous description [14]. For example, moments in DiDeMo [14] are created from fixed 5-second segments. However, such coarse temporal annotations are not well aligned with natural moments. In TVR, temporal windows are freely selected to more accurately capture important moments. (*ii*) converting a paragraph written for a whole video into separate query sentences [32,9,22]. While it is natural for people to use temporal connectives (e.g., 'first', 'then') and anaphora (e.g., pronouns) [34] in a paragraph, these words make individual sentences less suitable as retrieval queries. In comparison, the TVR annotation process encourages annotators to write queries individually without requiring the context of a paragraph. Besides, TVR also has a larger size and greater linguistic diversity, see Sec. 3.2.

**Methods**. Existing works [14,9,15,43,10,8] pose moment retrieval as ranking a predefined set of moment proposals. These proposals are typically generated with handcrafted rules [14,15] or sliding windows [9,43,10,8]. Typically, such proposals are not temporally precise and are not scalable to long videos due to high computational cost. [9,43,10] alleviate the first with a regression branch that offsets the proposals. However, they are still restricted by the coarseness of the initial proposals. Inspired by span predictors in reading comprehension [36,3] and action localization [27], we use start-end predictors to predict start-end probabilities from early fused query-video representations. Though these methods can be more flexibly applied to long videos and have shown promising performance on single video moment retrieval, the time cost of early fusion becomes unbearable when dealing with the corpus level moment retrieval problem: they require early fusing every possible query-video pair [8]. Proposal based approaches MCN [14] and CAL [8] use a late fusion design, in which the video representations can be pre-computed and stored, making the retrieval more efficient. The final moment predictions are then made by ranking the Squared Euclidean Distances between the proposals w.r.t. a given query. However, as they rely on predefined proposals, MCN and CAL still suffer from the aforementioned drawbacks, leading to less precise predictions and higher costs (especially for long videos). Recent works [48,4,49] consider word-level early fusion with the videos, which can be even more expensive. In contrast, XML uses a late fusion design with a novel Convolutional Start-End (ConvSE) detector, which produces more accurate moment predictions while reducing the computational cost.

## 3 Dataset

Our TVR dataset is built on 21,793 videos from 6 long-running TV shows across 3 genres (*sitcom*, *medical*, *crime*), provided by TVQA [24]. Videos are paired with subtitles and are on average 76.2 seconds in length. In the following, we describe how we collected TVR and provide a detailed analysis of the data.

### 3.1 Data Collection

We used Amazon Mechanical Turk (AMT) for TVR data collection. Each AMT worker was asked to write a query using information from the video and/or subtitle, then mark the start and end timestamps to define a moment that matches the written query. This query-moment pair is required to be a unique match within the given video, i.e., the query should be a referring expression [20,14] that uniquely localizes the moment. We additionally ask workers to select a query type from three types: *video-only* - queries relevant to the visual content only, *sub-only* - queries relevant to the subtitles only, and *video+sub* - queries that involve both. In our pilot study, we found workers preferred to write *sub-only* queries. A similar phenomenon was observed in TVQA [24], where people can achieve 72.88% QA accuracy by reading the subtitles only. Therefore, to ensure that we collect a balance of queries requiring one or both modalities, we split the data annotation into two rounds - *visual* round and *textual* round. For the visual round, we encourage workers to write queries related to the visual content, including both *video-only* and *video+sub* queries. For the textual round, we encourage *sub-only* and *video+sub* queries. We ensure data quality with the following strategies:[2]

**Qualification Test**. We designed a set of 12 multiple-choice questions as our qualification test and only let workers who correctly answer at least 9 questions participate in our annotation task, ensuring that workers understand our task requirements well. In total, 1,055 workers participated in the test, with a pass rate of 67%. Adding this qualification test greatly improved data quality.

**Automatic Check**. During collection, we used an automatic tool checking that all required annotations (query, timestamps, etc) have been performed and each query contains at least 8 words and is not copied from the subtitle.

**Manual Check**. Additional manual check of the collected data was done in house throughout the collection process. Those disqualified queries were re-annotated and workers with disqualified queries were removed from our worker list.

**Post-Annotation Verification**. To verify the quality of the collected data, we performed a post-annotation verification experiment. We set up another AMT task where workers were required to rate the quality of the collected query-moment pairs based on *relevance*, *is the query-moment pair a unique-match*, etc. The rating was done in a *likert-scale* manner with 5 options: *strongly agree*, *agree*, *neutral*, *disagree* and *strongly disagree*. Results show that 92% of the pairs have a rating of at least *neutral*. We further analyzed the group of queries that were rated as *strongly disagree*, and found that 80% of them were still of acceptable

[2] We present a pipeline figure of our data collection procedure in Fig. 9.

Table 1: Comparison of TVR with existing moment retrieval datasets. *Q* stands for query. *Q context* indicate which modality the queries are related. *Free st-ed* indicates whether the timestamps are freely annotated. *Individual Q* means the queries are collected as individual sentences, rather than sentences in paragraphs

| Dataset | Domain | #Q/#videos | Vocab. size | Avg. Q len. | Avg. len. (s) moment/video | Q context video | Q context text | Free st-ed | Q type anno. | Individual Q |
|---|---|---|---|---|---|---|---|---|---|---|
| TACoS [32] | Cooking | 16.2K / 0.1K | 2K | 10.5 | 5.9 / 287 | ✓ | - | ✓ | - | - |
| DiDeMo [14] | Flickr | 41.2K / 10.6K | 7.6K | 8.0 | 6.5 / 29.3 | ✓ | - | - | - | ✓ |
| ActivityNet Captions [22] | Activity | 72K / 15K | 12.5K | 14.8 | 36.2 / 117.6 | ✓ | - | ✓ | - | - |
| CharadesSTA [9] | Activity | 16.1K / 6.7K | 1.3K | 7.2 | 8.1 / 30.6 | ✓ | - | ✓ | - | - |
| TVR | TV show | 109K / 21.8K | 57.1K | 13.4 | 9.1 / 76.2 | ✓ | ✓ | ✓ | ✓ | ✓ |

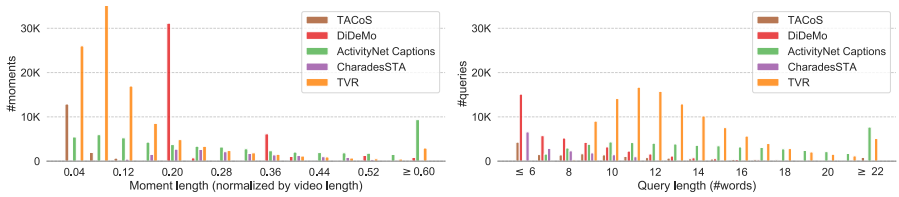


Fig. 2: Distributions of moment (*left*) and query (*right*) lengths. Compared to existing moment retrieval datasets [32,14,22,9], TVR has relatively shorter moments (normalized) and longer queries. Best viewed digitally with zoom

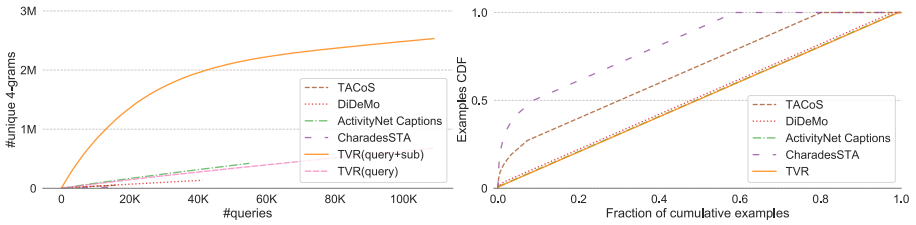


Fig. 3: *Left*: #unique 4-gram as a function of #queries. *Right*: CDF of queries ordered by frequency, to obtain this plot, we sampled 10K queries from each dataset, we consider two queries to be the same if they exact match, after tokenization and lemmatization, following [47]. Compared to existing moment retrieval datasets [32,14,22,9], TVR has greater diversity, i.e., it has more unique 4-grams and almost every TVR query is unique. Best viewed digitally with zoom

quality: e.g., slightly mismatched timestamps ($\leq$1 sec.). This verification was conducted on 3,600 query-moment pairs. Details are presented in Sec. A.1.

Given the high quality demonstrated by this verification, we did not further annotate each query, instead prioritizing collection toward adding more TVR queries, and collecting additional captions for each annotated moment to form **TVC**, a large-scale multimodal video captioning dataset with 262K captions. See details in Sec. D

Table 2: Percentage of queries that have multiple actions or involve multiple people. Statistics is based on 100 manually labeled queries from each dataset. We also show query examples, with unique person mentions underlined and actions in **bold**. Compared to existing datasets, TVR queries typically have more people and actions and require both *video* and *sub* (subtitle) context

| Dataset | #actions ≥2 (%) | #people ≥2 (%) | Query examples (*query type*) |
|---|---|---|---|
| TACoS [32] | 20 | 0 | She **rinses** the peeled carrots off in the sink. (*video*)<br>The person **removes** roots and outer leaves and **rewashes** the leek. (*video*) |
| CharadesSTA [9] | 6 | 12 | A person is **eating** food slowly. (*video*)<br>A person is **opening** the door to a bedroom. (*video*) |
| ActivityNet Caption [22] | 44 | 44 | He then **grabs** a metal mask and **positions** himself correctly on the floor. (*video*)<br>The same man **comes** back and **lifts** the weight over his head again. (*video*) |
| DiDeMo [14] | 6 | 10 | A dog **shakes** its body. (*video*)<br>A lady in a cowboy hat **claps** and **jumps** excitedly. (*video*) |
| TVR | 67 | 66 | Bert **leans** down and **gives** Amy a hug who is **standing** next to Penny. (*video*)<br>Taub **argues** with the patient that fighting in Hockey **undermines** the sport. (*sub*)<br>Chandler **points** at Joey while **describing** a woman who wants to **date** him. (*video+sub*) |

### 3.2 Data Analysis and Comparison

Table 1 shows an overview of TVR and its comparisons with existing moment retrieval datasets [32,9,22,14]. TVR contains 109K human annotated query-moment pairs on 21.8K videos, making it the largest of its kind. Moments have an average length of 9.1 seconds, and are annotated with tight start and end timestamps, enabling training and evaluating on more precise localization. Compared to existing datasets, TVR has relatively shorter (video-length normalized) moments and longer queries (Fig. 2). It also has greater linguistic diversity (Fig. 3): it has more unique 4-grams and almost every query is unique, making the textual understanding of TVR more challenging. As TVR is collected on TV shows, query-moment matching often involves understanding rich interactions between characters. Table 2 shows a comparison of the percentages of queries that involve more than one action or person across different datasets. 66% of TVR queries involve at least two people and 67% involve at least two actions, both of which are significantly higher than those of other datasets. This makes TVR an interesting testbed for studying multimodal interactions between people. Additionally, each TVR query is labeled with a query type, indicating whether this query is based on video, subtitle or both, which can be used for deeper analyses of the systems.

## 4 Cross-modal Moment Localization (XML)

In VCMR, the goal is to retrieve a moment from a large video corpus $V=\{v_i\}_{i=1}^{n}$ given a query $q_j$. Each video $v_i$ is represented as a list of consecutive short clips, i.e., $v_i=[c_{i,1}, c_{i,2}, ..., c_{i,l}]$. In TVR, each short clip is also associated with temporally aligned subtitle sentences. The retrieved moment is denoted as $v_i[t_{st}{:}t_{ed}]=[c_{i,t_{st}}, c_{i,t_{st}+1}, ..., c_{i,t_{ed}}]$. To address VCMR, we propose a hierarchical Cross-modal Moment Localization (XML) network. XML performs video retrieval

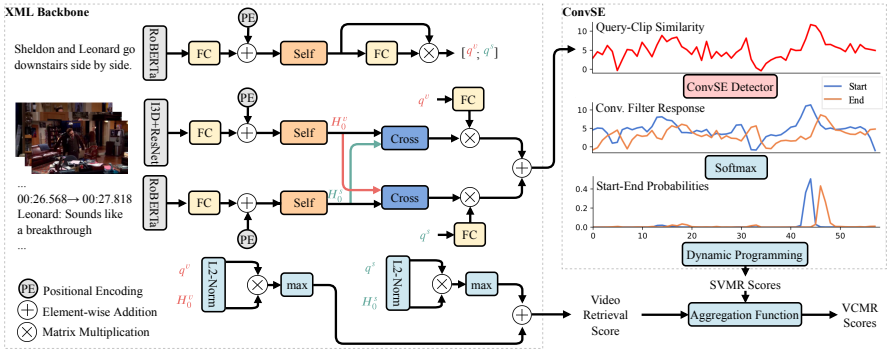


Fig. 4: Cross-modal Moment Localization (XML) model overview. *Self=Self Encoder*, *Cross=Cross Encoder*. We describe *XML Backbone* in Sec. 4.1, *ConvSE* module in Sec. 4.2 and show XML's training and inference procedure in Sec. 4.3

(VR) in its shallower layers and more fine-grained moment retrieval in its deeper layers. It uses a late fusion design with a novel Convolutional Start-End (ConvSE) detector, making the moment predictions efficient and accurate.

### 4.1 XML Backbone Network

**Input Representations**. To represent videos, we consider both appearance and motion features. For appearance, we extract 2048D ResNet-152 [13] features at 3FPS and max-pool the features every 1.5 seconds to get a clip-level feature. For motion, we extract 1024D I3D [2] features every 1.5 seconds. The ResNet-152 model is pre-trained on ImageNet [5] for image recognition, and the I3D model is pre-trained on Kinetics-600 [19] for action recognition. The final video representation is the concatenation of the two features after L2-normalization, denoted as $E^v \in \mathbb{R}^{l \times 3072}$, where $l$ is video length (#clips). We extract contextualized text features using a 12-layer RoBERTa [28]. Specifically, we first fine-tune RoBERTa using the queries and subtitle sentences in TVR train-split with MLM objective [7], then fix the parameters to extract contextualized token embeddings from its second-to-last layer [25]. For queries, we directly use the extracted token embeddings, denoted as $E^q \in \mathbb{R}^{l_q \times 768}$, where $l_q$ is query length (#words). For subtitles, we first extract token-level embeddings, then max-pool them every 1.5 seconds to get a 768D clip-level feature vector. We use a 768D zero vector if encountering no subtitle. The final subtitle embedding is denoted as $E^s \in \mathbb{R}^{l \times 768}$. The extracted features are projected into a low-dimensional space via a linear layer with ReLU [12]. We then add learned positional encoding [7] to the projected features. Without ambiguity, we reuse the symbols by denoting the processed features as $E^v \in \mathbb{R}^{l \times d}$, $E^s \in \mathbb{R}^{l \times d}$, $E^q \in \mathbb{R}^{l_q \times d}$, where $d$ is hidden size.

**Query Encoding**. As TVR queries can be related to either video or subtitle, we adopt a modular design to dynamically decompose the query into two modularized vectors. Specifically, the query feature is encoded using a *Self-Encoder*, consisting

of a self-attention [40] layer and a linear layer, with a residual [13] connection followed by layer normalization [1]. We denote the encoded query as $H^q \in \mathbb{R}^{l_q \times d}$. Then, we apply two trainable modular weight vectors $\mathbf{w}_m \in \mathbb{R}^d$, $m \in \{v, s\}$ to compute the attention scores of each query word w.r.t. the video ($v$) or subtitle ($s$). The scores are used to aggregate the information of $H^q = \{\mathbf{h}_\mathbf{r}^\mathbf{q}\}_{r=1}^{l_q}$ to generate modularized query vectors $\mathbf{q}^m \in \mathbb{R}^d$ [46]:

$$a_r^m = \frac{\exp(\mathbf{w}_m^T \mathbf{h}_r^q)}{\sum_{k=1}^{l_q} \exp(\mathbf{w}_m^T \mathbf{h}_k^q)}, \quad \mathbf{q}^m = \sum_{r=1}^{l_q} a_r^m \mathbf{h}_\mathbf{r}^\mathbf{q}, \quad \text{where } m \in \{v, s\}. \tag{1}$$

**Context Encoding**. Given the video and subtitle features $E^v$, $E^s$, we use two Self-Encoders to compute their single-modal contextualized features $H_0^v \in \mathbb{R}^{l \times d}$ and $H_0^s \in \mathbb{R}^{l \times d}$. Then, we encode their cross-modal representations via *Cross-Encoder*. which takes as input the self-modality and cross-modality features, and encodes the two via cross-attention [40] followed by a linear layer, a residual connection, a layer normalization, and another Self-Encoder. We denote the final video and subtitle representations as $H_1^v \in \mathbb{R}^{l \times d}$ and $H_1^s \in \mathbb{R}^{l \times d}$, respectively.

### 4.2 Convolutional Start-End Detector

Given $H_1^v, H_1^s$ and $\mathbf{q}^v, \mathbf{q}^s$, we compute query-clip similarity scores $S_{\text{query-clip}} \in \mathbb{R}^l$:

$$S_{\text{query-clip}} = \frac{1}{2}(H_1^v \mathbf{q}^v + H_1^s \mathbf{q}^s). \tag{2}$$

To produce moment predictions from $S_{\text{query-clip}}$, one could rank sliding window proposals with confidence scores computed as the average of scores in each proposal region, or use TAG [50] to progressively group top-scored regions. However, both methods require handcrafted rules and are not trainable. Inspired by edge detectors in image processing [38], we propose Convolutional Start-End detector (**ConvSE**) with two 1D convolution filters to learn to detect start (up) and end (down) edges in the score curves. Clips inside a semantically close span will have higher similarity to the query than those outside, naturally forming detectable edges around the span. Fig. 4 (right) and Fig. 7 show examples of the learned ConvSE filters applied to the similarity curves. Specifically, we use two trainable filters (no bias) to generate the start (st) and end (ed) scores:

$$S_{\text{st}} = \text{Conv1D}_{\text{st}}(S_{\text{query-clip}}), \quad S_{\text{ed}} = \text{Conv1D}_{\text{ed}}(S_{\text{query-clip}}). \tag{3}$$

The scores are normalized with softmax to output the probabilities $P_{\text{st}}, P_{\text{ed}} \in \mathbb{R}^l$. In Sec. 5.3, we show ConvSE outperforms the baselines and is also interpretable.

### 4.3 Training and Inference

**Video Retrieval**. Given the modularized queries $\mathbf{q}^v, \mathbf{q}^s$ and the encoded contexts $H_0^v, H_0^s$, we compute the video-level retrieval (VR) score as:

$$s^{\text{vr}} = \frac{1}{2} \sum_{m \in \{v,s\}} \max\left( \frac{H_0^m}{\|H_0^m\|} \frac{\mathbf{q}^m}{\|\mathbf{q}^m\|} \right). \tag{4}$$

This essentially computes the cosine similarity between each clip and query and picks the maximum. The final VR score is the average of the scores from the two modalities. During training, we sample two negative pairs $(q_i, v_j)$ and $(q_z, v_i)$ for each positive pair of $(q_i, v_i)$ to calculate a combined hinge loss as [46]:

$$L^{\text{vr}} = \frac{1}{n}\sum_i [\max(0, \Delta + s^{\text{vr}}(v_j|q_i) - s^{\text{vr}}(v_i|q_i)) + \max(0, \Delta + s^{\text{vr}}(v_i|q_z) - s^{\text{vr}}(v_i|q_i))]. \quad (5)$$

**Single Video Moment Retrieval**. Given the start, end probabilities $P_{\text{st}}, P_{\text{ed}}$, we define single video moment retrieval loss as:

$$L^{\text{svmr}} = -\frac{1}{n}\sum_i [\log(P_{i,\text{st}}(t^i_{\text{st}})) + \log(P_{i,\text{ed}}(t^i_{\text{ed}}))], \quad (6)$$

where $t^i_{\text{st}}$ and $t^i_{\text{ed}}$ are the ground-truth indices. At inference, predictions can be generated from the probabilities in linear time using dynamic programming [36]. The confidence score of a predicted moment $[t^{'}_{\text{st}}, t^{'}_{\text{ed}}]$ is computed as:

$$s^{\text{svmr}}(t^{'}_{\text{st}}, t^{'}_{\text{ed}}) = P_{\text{st}}(t^{'}_{\text{st}}) P_{\text{ed}}(t^{'}_{\text{ed}}),\ t^{'}_{\text{st}} \leq t^{'}_{\text{ed}}. \quad (7)$$

To use length prior, we add an additional constraint $L_{min} \leq t^{'}_{\text{ed}} - t^{'}_{\text{st}} + 1 \leq L_{max}$. For TVR, we set $L_{min}$=2 and $L_{max}$=16 for clip length 1.5 seconds.

**Video Corpus Moment Retrieval**. Our final training loss combines both: $L^{\text{vcmr}} = L^{\text{vr}} + \lambda L^{\text{svmr}}$, where the hyperparameter $\lambda$ is set as 0.01. At inference, we compute the VCMR score with the following aggregation function:

$$s^{\text{vcmr}}(v_j, t_{\text{st}}, t_{\text{ed}}|q_i) = s^{\text{svmr}}(t_{\text{st}}, t_{\text{ed}}|v_j, q_i)\exp(\alpha s^{\text{vr}}(v_j|q_i)), \quad (8)$$

where $s^{\text{vcmr}}(v_j, t_{\text{st}}, t_{\text{ed}}|q_i)$ is the retrieval score of moment $v_j[t_{\text{st}}{:}t_{\text{ed}}]$ w.r.t. the query $q_i$. The exponential term and the hyperparameter $\alpha$ are used to balance the importance of the two scores. A higher $\alpha$ encourages more moments from top retrieved videos. Empirically, we find $\alpha$=20 works well. At inference, for each query, we first retrieve the top 100 videos based on $s^{\text{vr}}$, then rank all the moments in the 100 videos by $s^{\text{vcmr}}$ to give the final predictions.

## 5 Experiments

### 5.1 Data, Metrics and Implementation Details

**Data**. TVR contains 109K queries from 21.8K videos. We split TVR into 80% *train*, 10% *val*, 5% *test-public* and 5% *test-private* splits such that videos and their associated queries appear in only one split. *test-public* will be used for a public leaderboard, *test-private* is reserved for future challenges.

Table 3: Baseline comparison on TVR *test-public* set, VCMR task. Model references: *MCN* [14], *CAL* [8], *MEE* [29], *ExCL* [11]. Results with TEF [14] feature are presented in Table 5

| Model | w/ video | w/ sub. | IoU=0.5 | | | | IoU=0.7 | | | | Runtime ↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | R@1 | R@5 | R@10 | R@100 | R@1 | R@5 | R@10 | R@100 | (seconds) |
| Chance | - | - | 0.00 | 0.02 | 0.04 | 0.33 | 0.00 | 0.00 | 0.00 | 0.07 | |
| **Proposal based Methods** | | | | | | | | | | | |
| MCN | ✓ | ✓ | 0.02 | 0.15 | 0.24 | 2.20 | 0.00 | 0.07 | 0.09 | 1.03 | - |
| CAL | ✓ | ✓ | 0.09 | 0.31 | 0.57 | 3.42 | 0.04 | 0.15 | 0.26 | 1.89 | - |
| **Retrieval + Re-ranking** | | | | | | | | | | | |
| MEE+MCN | ✓ | ✓ | 0.92 | 3.69 | 5.58 | 17.91 | 0.42 | 1.89 | 2.98 | 10.84 | 66.8 |
| MEE+CAL | ✓ | ✓ | 0.97 | 3.75 | 5.80 | 18.66 | 0.39 | 1.69 | 2.98 | 11.52 | 161.5 |
| MEE+ExCL | ✓ | ✓ | 0.92 | 2.53 | 3.60 | 6.01 | 0.33 | 1.19 | 1.73 | 2.87 | 1307.2 |
| XML | ✓ | ✓ | **7.25** | **16.24** | **21.65** | **44.44** | **3.25** | **8.71** | **12.49** | **29.51** | **25.5** |

**Metrics**. Following [8,9], we use average recall at K (R@K) over all queries as our metric. A prediction is correct if: (*i*) predicted video matches the ground truth; (*ii*) predicted span has high overlap with the ground truth where temporal intersection over union (IoU) is used to measure overlap.

**Implementation Details**. All baseline comparisons are configured to use the same hidden size as XML. We train the baselines following the original papers. We use the same features for all the models. To support retrieval using subtitle for the baselines, we add a separate subtitle stream and average the final predictions from both streams. Non-maximum suppression is not used as we do not observe consistent performance gain on the *val* set.

### 5.2 Baselines Comparison

In this section, we compare XML with baselines on TVR *test-public* set (5,445 queries and 1,089 videos). We report the runtime for top-performing methods, averaged across 3 runs on an RTX 2080Ti GPU. Time spent on data loading, preprocessing, backend model (i.e., ResNet-152, I3D, RoBERTa) feature extraction, etc, is ignored since they should be similar for all methods. We mainly focus on the VCMR task here. In Sec. B and Sec. C, we include additional experiments: (1) model performance on single video moment retrieval and video retrieval tasks; (2) computation and storage cost comparison in a 1M videos corpus; (3) Temporal Endpoint Feature (TEF) [14] model results; (4) feature and model architecture ablation studies; (5) VCMR results on DiDeMo [14] dataset, etc.

**Proposal based Methods**. MCN [14] and CAL [8] pose the moment retrieval task as a ranking problem in which all moment proposal candidates are ranked based on their squared Euclidean Distance with the queries. For VCMR, they require directly ranking all the proposals (95K in the following experiments) in the video corpus for each query, which can be costly and difficulty. In contrast, XML uses a hierarchical design that performs video retrieval in its shallow layers and moment retrieval on the retrieved videos in its deeper layers. In Table 3, XML is showing to have significantly higher performance than MCN and CAL.

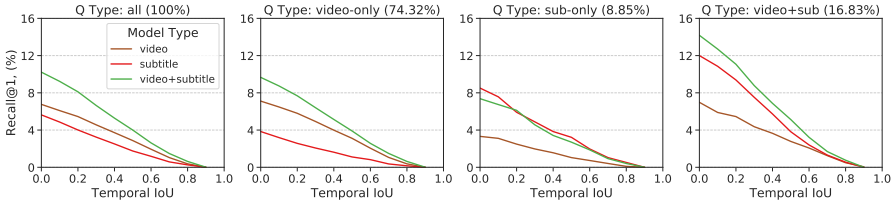


Fig. 5: Performance breakdown of XML models that use only *video*, *subtitle*, or both as inputs, by different query types (with percentage of queries shown in brackets). The performance is evaluated on TVR *val* set for VCMR

**Retrieval+Re-ranking Methods**. We also compare to methods under the retrieval+re-ranking setting [8] where we first retrieve a set of candidate videos using a given method and then re-rank the moment predictions in the candidate videos using another method. Specifically, we first use MEE [29] to retrieve 100 videos for each query as candidates. Then, we use MCN and CAL to rank all of the proposals in the candidate videos. ExCL [11] is an early fusion method designed for SVMR, with a start-end predictor. We adapt it to VCMR by combining MEE video-level scores with ExCL moment-level scores, using Eq. 8. The results are shown in Table 3. Compared to their purely proposal based counterparts (i.e., MCN and CAL), both MEE+MCN and MEE+CAL achieve significant performance gain, showing the benefit of reducing the number of proposals needed to rank (by reducing the number of videos). However, they are still far below XML as they use very coarse-grained, predefined proposals. In Sec. 5.3, we show our start-end detector performs consistently better than predefined proposals [8,50] under our XML framework. Compared to MEE+ExCL, XML achieves 9.85× performance gain (3.25 *vs.* 0.33, R@1 IoU=0.7) and 51.3× speedup (25.5s *vs.* 1307.2s). In the Sec. B.1, we show that this speedup can be even more significant (287×) when retrieving on a larger scale video corpus (1M videos) with pre-encoded video representations. This huge speedup shows the effectiveness of XML's late fusion design over ExCL's early fusion design.

### 5.3 Model Analysis

**Video vs. Subtitle**. In Fig. 5, we compare to XML variants that use only video or subtitle. We observe that the full *video+subtitle* model has better overall performance than single modality models (*video* and *subtitle*), demonstrating that both modalities are useful. We also see that a model trained on one modality does not perform well on the queries tagged by another modality, e.g., the *video* model performs much worse on *sub-only* queries compared to the *subtitle* model.

**ConvSE: Comparison and Analysis**. To produce moment predictions from the query-clip similarity signals, we proposed ConvSE that learns to detect start (up) and end (down) edges in the 1D similarity signals. To show its effectiveness, we compare ConvSE with two baselines under our XML backbone network: (1) sliding window, where we rank proposals generated by multi-scale sliding windows,

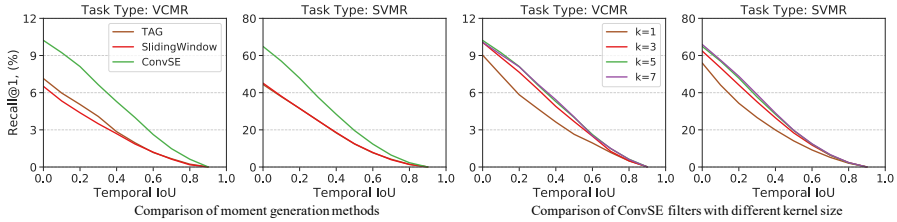


Fig. 6: ConvSE Analysis. *Left*: comparison of moment generation methods. *Right*: comparison of ConvSE filters with different kernel sizes ($k$)

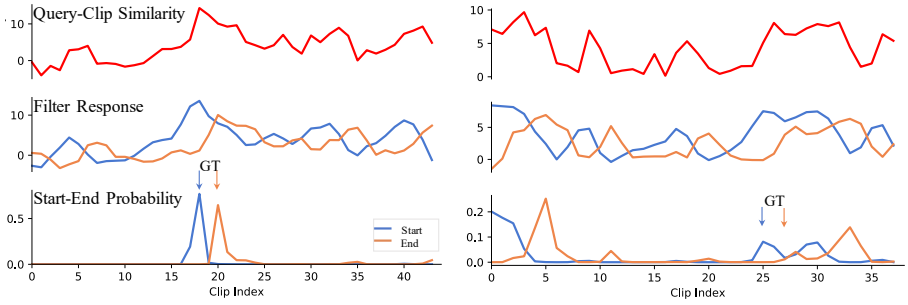


Fig. 7: Examples of learned ConvSE filters applying on query-clip similarity scores. Ground truth span is indicated by the two arrows labeled by *GT*. Note the two filters output stronger responses on the up (*Start*) and down (*End*) edges

with proposal confidence scores calculated as the average of scores inside each proposal region. On average, it produces 87 proposals per video. The proposals used here are the same as the ones used for MCN and CAL in our previous experiments; (2) TAG [50] that progressively groups top-scored clips with the classical watershed algorithm [33]. Since these two methods do not produce start-end probabilities, we cannot train the model with the objective in Eq. 6. Thus, we directly optimize the query-clip similarity scores in Eq.2 with Binary Cross Entropy loss: we assign a label of 1 if the clip falls into the ground-truth region, 0 otherwise. While both sliding window and TAG approaches rely on handcrafted rules, ConvSE **learns from data**. We show in Fig. 6 (left), under the same XML backbone network, ConvSE has consistent better performance across all IoU thresholds on both VCMR and SVMR tasks.

In Fig. 6 (right), we vary the kernel size ($k$) of ConvSE filters. While the performance is reasonable when $k$=3, 5 or 7, we observe a significant performance drop at $k$=1. In this case, the filters essentially degrade to scaling factors on the scores. This comparison demonstrates that neighboring information is important. Fig. 7 shows examples of using the **learned convolution filters**: the filters output stronger responses to the up (*Start*) and down (*End*) edges of the score


Fig. 8: XML prediction examples for VCMR, on TVR *val* set. We show top-3 retrieved moments for each query. *Top row* shows modular attention scores for query words. *Left column* shows a correct prediction, *right column* shows a failure. Text inside *dashed boxes* is the subtitles associated with the predicted moments. *Orange box* shows the predictions, *green bar* shows the ground truth

curves and thus detect them. Interestingly, the learned weights $\text{Conv1D}_{\text{st}}$ and $\text{Conv1D}_{\text{ed}}$ in Fig. 7 are similar to the edge detectors in image processing [38].

**Qualitative Analysis**. Fig. 8 shows XML example predictions on the TVR *val* set. In the top row, we also show the query word attention scores for video and subtitle, respectively. Fig. 8 (left) shows a correct prediction. The top-2 moments are from the same video and are both correct. The third moment is retrieved from a different video. While incorrect, it is still relevant as it also happens in a 'restaurant'. Fig. 8 (right) shows a failure. It is worth noting that the false moments are very close to the correct prediction with minor differences ('on the shoulder' *vs.* 'around the shoulder'). Besides, it is also interesting to see which words are important for video or subtitle. For example, the words 'waitress', 'restaurant', 'menu' and 'shoulder' get the most weight for video; while the words 'Rachel', 'menu', 'Barney', 'Ted' have higher attention scores for subtitle.

## 6 Conclusion

In this work, we present TVR, a large-scale dataset designed for multimodal moment retrieval tasks. Detailed analyses show TVR is of high quality and is more challenging than previous datasets. We also propose Cross-modal Moment Localization (XML), an efficient model suitable for the VCMR task.

**Acknowledgements:** We thank the reviewers for their helpful feedback. This research is supported by NSF Award #1562098, DARPA MCS Grant #N66001-19-2-

4031, DARPA KAIROS Grant #FA8750-19-2-1004, ARO-YIP Award #W911NF-18-1-0336, and Google Focused Research Award.

# A Additional TVR Data Details

## A.1 Data Collection

**TVR Data Collection Procedure.** In Fig. 9 we show an overview of TVR data collection procedure. For details of each step, please refer to both Sec. 3.1 and the rest of this section.

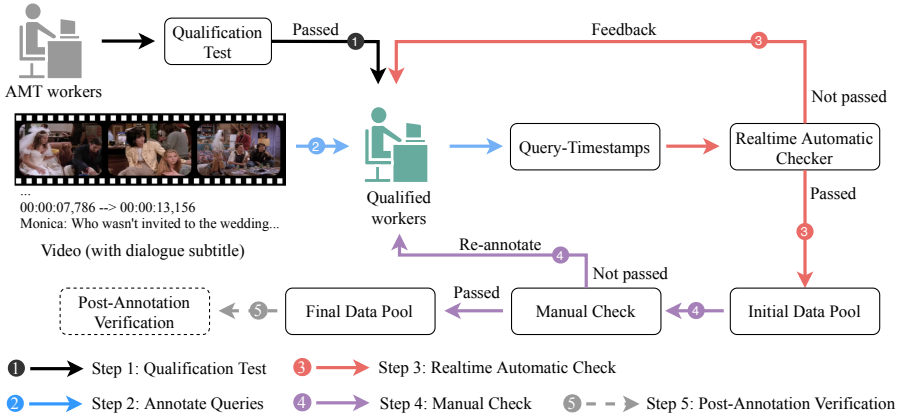


Fig. 9: TVR data collection procedure

**Qualification Test**. We designed a qualification test with 12 multiple-choice questions and only let workers who correctly answer at least 9 questions participate in our annotation task, ensuring that workers understand our task requirements well. In total, 1,055 workers participated in the test, with a pass rate of 67%. Adding this qualification test greatly improved data quality. In Fig. 10, we show a question from our qualification test. This particular question is designed to make sure the annotators write relevant and correct descriptions (queries).

**Post-Annotation Verification**. To verify the quality of the collected data, we performed a post-annotation verification experiment. We set up another AMT task where workers were required to rate the quality of the collected query-moment pairs based on *relevance*, *is the query-moment pair a unique-match*, etc. The rating was done in a *likert-scale* manner with 5 options: *strongly agree*, *agree*, *neutral*, *disagree* and *strongly disagree*, as is shown in Fig. 11. Results show that 92% of the pairs have a rating of at least *neutral*. This verification was conducted on 3,600 query-moment pairs. Detailed rating distribution is shown in Fig. 12. We further analyzed the group of queries that were rated as *strongly disagree*, and

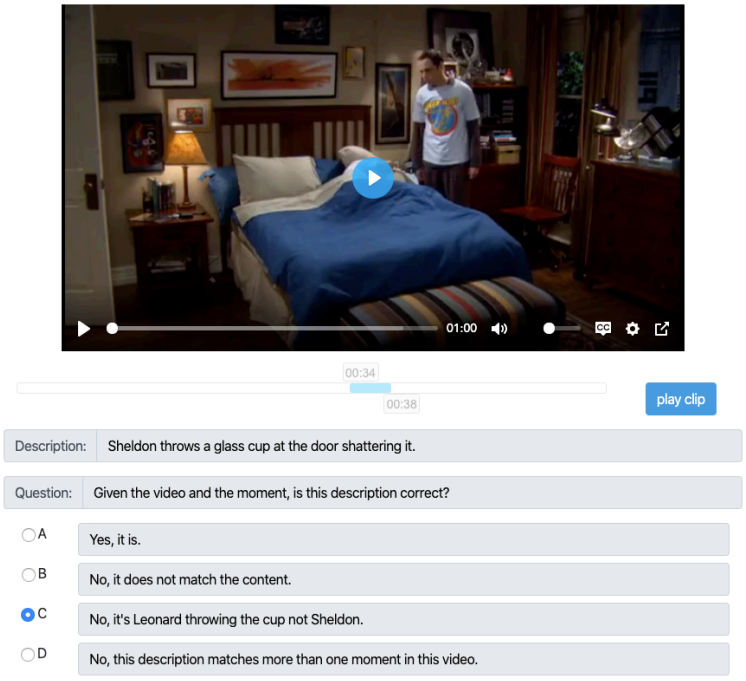


Fig. 10: Example question from our qualification test

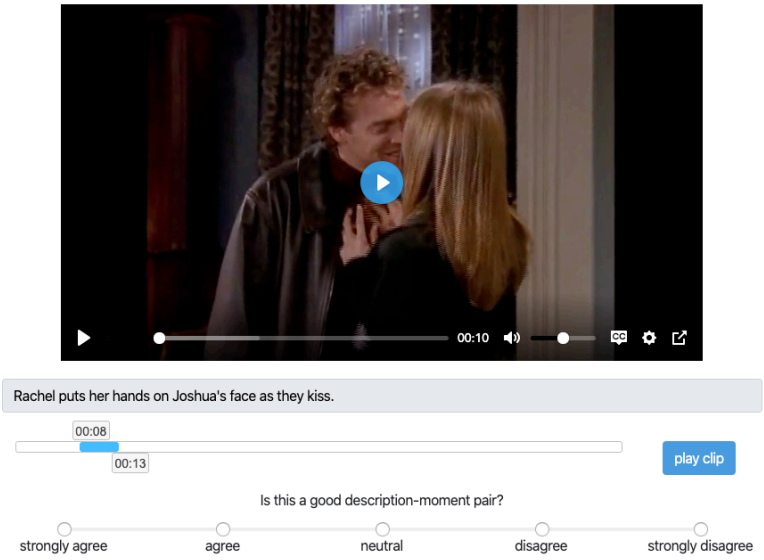


Fig. 11: Post-Annotation quality rating interface

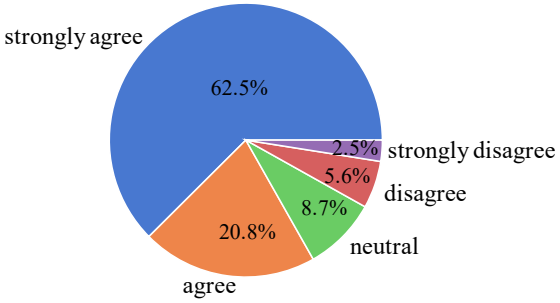


Fig. 12: Distribution of quality rating on 3,600 query-moment pairs. 92% of the pairs have a rating of at least *neutral*

Table 4: Data Statistics for each TV show. BBT=*The Big Bang Theory*, HIMYM=*How I Met You Mother*, Grey=*Grey's Anatomy*, Epi=Episode, Sea.=Season

| Show | Genre | #Sea. | #Epi. | #Clip | #Query |
|---|---|---|---|---|---|
| BBT | sitcom | 10 | 220 | 4,198 | 20,990 |
| Friends | sitcom | 10 | 226 | 5,337 | 26,685 |
| HIMYM | sitcom | 5 | 72 | 1,512 | 7,560 |
| Grey | medical | 3 | 58 | 1,427 | 7,135 |
| House | medical | 8 | 176 | 4,621 | 23,105 |
| Castle | crime | 8 | 173 | 4,698 | 23,490 |
| Total | — | 44 | 925 | 21,793 | 108,965 |

found that 80% of them were still of acceptable quality: e.g., slightly mismatched timestamps ($\leq$1 sec.). For the group of queries that were rated as *disagree*, this number is 90%. This verification demonstrates the high quality of the data.

### A.2 Data Analysis

**Statistics by TV Show**. TVR is built on 21,793 videos (provided by TVQA [24]) from 6 long-running TV shows: *The Big Bang Theory*, *Friends*, *How I Met You Mother*, *Grey's Anatomy*, *House*, *Castle*. Table 4 shows detailed statistics.

**Moments and Queries**. Fig. 13 (*left*) shows TVR moment length distribution. The majority of the moments are relatively short, with an average length of 9.1 secs. As a comparison, the average length of the videos is 76.2 secs. Fig. 13 (*right*) shows the video-length normalized moment center distributions. More moments are located at the beginning of the videos. A similar phenomenon was observed in DiDeMo [14]. Fig. 14 shows TVR query type distribution, around 91% of the queries need video context, while 26% of the queries need subtitle context.

**Frequent Words in Queries**. In Fig. 15 we show frequent nouns (*left*) and verbs (*right*) in TVR queries. The words are lemmatized, stop words are removed.

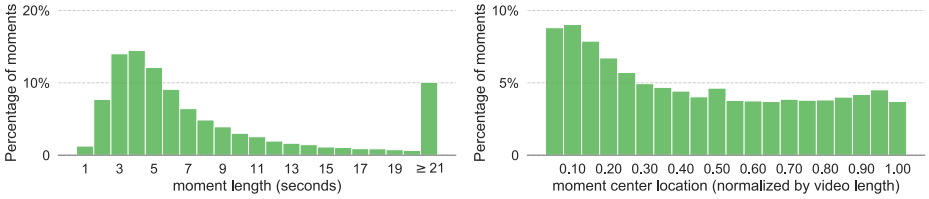


Fig. 13: Distribution of TVR moment lengths (*left*) and moment center locations (*right*)

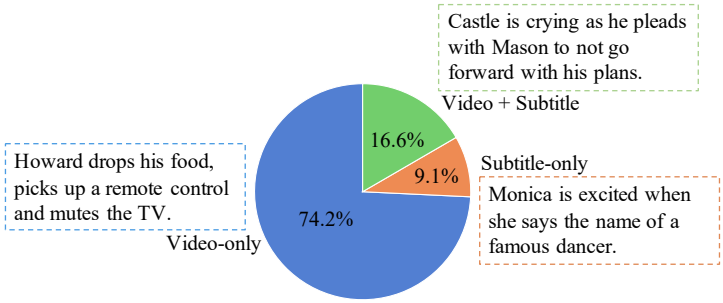


Fig. 14: Distribution of query types based on reasoning type. Text inside *dashed boxes* are query examples for each query type

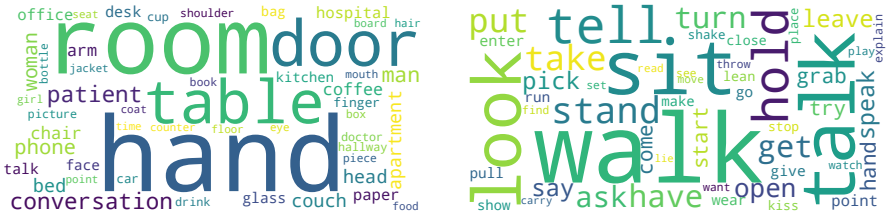


Fig. 15: TVR query word clouds for nouns (*left*) and verbs (*right*)

We notice that TVR covers a wide range of common objects/scenes and actions, while also has many genre-specific words such as 'patient' and 'hospital'.

**Video Comparison**. TVR videos are from 6 TV shows of 3 different genres, covering a diverse set of objects/scenes/activities. In Fig. 19, we compare TVR videos with videos from existing datasets [32,9,22,14]. Each TVR video typically has more visual diversity, i.e., more camera viewpoints, activities and people, etc.

Table 5: Baseline comparison on TVR *test-public* set, VCMR task. Model references: *MCN* [14], *CAL* [8], *MEE* [29], *ExCL* [11]. This table includes models trained with Temporal Endpoint Feature (TEF) [14]

| Model | w/ video | w/ sub. | IoU=0.5 | | | | IoU=0.7 | | | | Runtime ↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | R@1 | R@5 | R@10 | R@100 | R@1 | R@5 | R@10 | R@100 | (seconds) |
| Chance | - | - | 0.00 | 0.02 | 0.04 | 0.33 | 0.00 | 0.00 | 0.00 | 0.07 | - |
| Frequency | - | - | 0.06 | 0.07 | 0.11 | 0.28 | 0.02 | 0.04 | 0.06 | 0.11 | - |
| **Proposal based Methods** | | | | | | | | | | | |
| TEF-only | - | - | 0.00 | 0.09 | 0.15 | 0.79 | 0.00 | 0.07 | 0.09 | 0.48 | - |
| MCN | ✓ | ✓ | 0.02 | 0.15 | 0.24 | 2.20 | 0.00 | 0.07 | 0.09 | 1.03 | - |
| MCN (TEF) | ✓ | ✓ | 0.04 | 0.11 | 0.17 | 1.84 | 0.02 | 0.06 | 0.07 | 1.10 | - |
| CAL | ✓ | ✓ | 0.09 | 0.31 | 0.57 | 3.42 | 0.04 | 0.15 | 0.26 | 1.89 | - |
| CAL (TEF) | ✓ | ✓ | 0.04 | 0.17 | 0.31 | 2.48 | 0.02 | 0.15 | 0.22 | 1.30 | - |
| **Retrieval + Re-ranking** | | | | | | | | | | | |
| MEE+MCN | ✓ | ✓ | 0.92 | 3.69 | 5.58 | 17.91 | 0.42 | 1.89 | 2.98 | 10.84 | - |
| MEE+MCN (TEF) | ✓ | ✓ | 1.36 | 3.89 | 5.79 | 19.34 | 0.62 | 2.04 | 3.21 | 11.66 | 66.8 |
| MEE+CAL | ✓ | ✓ | 0.97 | 3.75 | 5.80 | 18.66 | 0.39 | 1.69 | 2.98 | 11.52 | - |
| MEE+CAL (TEF) | ✓ | ✓ | 1.23 | 4.00 | 6.52 | 20.07 | 0.66 | 1.93 | 3.09 | 12.03 | 161.5 |
| MEE+ExCL | ✓ | ✓ | 0.92 | 2.53 | 3.60 | 6.01 | 0.33 | 1.19 | 1.73 | 2.87 | - |
| MEE+ExCL (TEF) | ✓ | ✓ | 1.01 | 2.50 | 3.60 | 5.77 | 0.40 | 1.21 | 1.73 | 2.96 | 1307.2 |
| XML (sw) | ✓ | ✓ | 3.82 | 10.38 | 14.20 | 35.89 | 1.91 | 5.25 | 8.12 | 23.47 | - |
| XML | ✓ | ✓ | **7.25** | **16.24** | **21.65** | **44.44** | **3.25** | **8.71** | **12.49** | **29.51** | - |
| XML (TEF) | ✓ | ✓ | **7.88** | **16.53** | **21.84** | **45.51** | **3.32** | **9.46** | **13.41** | **30.52** | **25.5** |

# B Additional TVR Experiments

## B.1 More VCMR Experiments

**Frequency Baseline**. Following prior works [14,8], we first discretize the video-length normalized start-end points, then use moments with most frequent start-end points as predictions. For video retrieval, we randomly sample videos from the dataset. The results of this baseline is presented in Table 5. We observe this baseline has slightly better performance than chance, we hypothesize it is mainly caused by the fact that the annotators tend to annotate the first few seconds of the video [14], as we have shown in Fig. 13 (*Right*).

**Models Trained with TEF**. It is shown in [14,8] that adding Temporal Endpoint Feature (TEF) [14] improves models' performance in moment retrieval tasks. In Table 5, we compare models trained with TEF. In most cases, adding TEF increases models' performance, which suggests there exists a certain degree of bias in the proposed dataset. This phenomenon is also observed by recent works [14,8] in various moment retrieval datasets, i.e., DiDeMo [14], CharadesSTA [9] and ActivityNet Captions [22]. We attribute this phenomenon into two aspects: (1)*moment distribution bias* - the moments are not evenly distributed over the video, e.g., in TVR and DiDeMo [14], there are more moments appear at the beginning of the video. (2)*language timestamp correlation bias* - some query words are highly indicative of the potential temporal location of the queries, e.g., temporal connectives like 'first' strongly indicate the associated query might be located around the beginning of the video and pronouns like 'He' may suggest this query should not be placed at the beginning of the video as people would usually not

Table 6: Model architecture ablation on TVR *val* set, VCMR task. Our full XML model in the last row is configured with transformer encoder and modular query. All models use both videos and subtitles

| Model | IoU=0.7 | | | |
|---|---|---|---|---|
| | R@1 | R@5 | R@10 | R@100 |
| **Self-Encoder Type** | | | | |
| XML (LSTM) | 2.12 | 4.97 | 6.86 | 18.06 |
| XML (CNN) | 2.45 | 5.53 | 7.77 | 19.88 |
| **Modular Query** | | | | |
| XML (No modular query) | 2.46 | 5.87 | 8.56 | 22.00 |
| XML | **2.62** | **6.39** | **9.05** | **22.47** |

use pronouns when they first mention someone. The second bias commonly exists in datasets that are built by converting paragraphs into separate sentences, i.e., CharadesSTA [9], TACoS [32] and ActivityNet Captions [22]. TVR avoids this bias by explicitly ask annotators to write queries as individual sentences without requiring the context of a paragraph.

**XML with Sliding Windows**. In Sec. 5.3, we compared XML variants with different proposal generation strategies. In Table 5, we further compare XML (sw, sliding window) with MCN/CAL models. For details of this variant, see Sec. 5.3. Compared to the best baseline (MEE+CAL), using the same set of sliding window proposals, we observe XML (sw) still perform much better (3.82 *vs.* 0.97, R@1 IoU=0.7). We hypothesize that the lower performance of MCN/CAL models compared to XML is mainly caused by the difficulties of training and ranking with a large pool of proposal candidates (1.5M proposals for TVR *train*). Both MCN and CAL are trained with a ranking objective, which relies on informative negatives to learn effectively. However, effective negative sampling in such a large pool of candidates can be challenging. In comparison, XML breaks the video corpus level moment retrieval problem into two sub-problems: video-level and moment-level retrieval. At video-level retrieval, XML performs ranking within a small set of videos (17.4K), which eases the aforementioned issue. At moment-level, XML (sliding window) utilizes Binary Cross Entropy to maximize the similarity scores of each ground-truth clip, eliminating the need for manually designing a negative sampling strategy.

**Model Architecture**. Table 6 presents a model architecture ablation. We first compare with different self-encoder architectures, replacing our transformer style encoder with a bidirectional LSTM encoder [24] or a CNN encoder [45,25]. We observe worse performance after the change and attribute this performance drop to the ineffectiveness of LSTMs and CNNs to capture long-term dependencies [16,40]. Next, we compare XML with a variant that uses a single max-pooled query instead of two modularized queries. Across all metrics, XML performs better than the variant without modular queries, showing the importance of considering different query representations in matching the context from different modalities.

Table 7: Feature ablation on TVR *val* set, VCMR task. All models use both videos and subtitles

| Model | IoU=0.7 | | | |
|---|---|---|---|---|
| | R@1 | R@5 | R@10 | R@100 |
| XML (ResNet) | 2.28 | 5.40 | 7.33 | 20.28 |
| XML (I3D) | 2.22 | 5.75 | 8.37 | 21.20 |
| XML (ResNet+I3D) | **2.62** | **6.39** | **9.05** | **22.47** |

Table 8: VCMR on 1M videos with 100 queries. TVR *test-public* set results are included as reference.Model references: *MCN* [14], *CAL* [8], *MEE* [29], *ExCL* [11]

| Model | IoU=0.7 | | Search 100 queries in 1M videos ↓ | | |
|---|---|---|---|---|---|
| | R@1 | R@5 | feat time (s) | feat size (GB) | retrieval time (s) |
| **Retrieval + Re-ranking** | | | | | |
| MEE+MCN | 0.42 | 1.89 | 131 | 326 | 0.090 |
| MEE+CAL | 0.39 | 1.69 | 841 | 2,235 | 0.166 |
| MEE+ExCL | 0.33 | 1.19 | - | - | 1.435 |
| XML | **3.25** | **8.71** | **29** | **76** | **0.005** |

**Feature Ablation**. We tested XML model with different visual features, the results are shown in Table 7. The model that uses both static appearance features (ResNet [13]) and action features (I3D [2]) outperforms models using only one of the features, demonstrating the importance of recognizing both the objects and the actions in the VCMR task.

**Retrieval Efficiency in 1M Videos**. We consider Video Corpus Moment Retrieval in a video corpus containing 1M videos with 100 queries. Following [8], we conduct this experiment in a simulated setting with each video containing 20 clips with max moment length of 14 clips. Each query containing 15 words. We report the following metrics: (1) feature encoding time (*feat time*) - measures the time for encoding the context (video and subtitle) features offline. (2) encoded feature size (*feat size*) - measures the disk space needed to store the encoded context features. (3) retrieval time (*retrieval time*) - measures the time needed to retrieve relevant moments for 100 new queries. It includes time for encoding the queries and performing approximate nearest neighbor search [18] or matrix multiplication. The time spent on data loading, pre-processing, feature extraction on backend models (i.e., ResNet-152, I3D, RoBERTa) are not considered as they should be similar if not the same for all the methods. Note that the *retrieval time* here is different from the *runtime* in Table 5, which additional includes *feat time*. We do not report *feat time* and *feat size* for ExCL [11] as it does not have the ability to pre-encode the features - its context encoding depends on the input queries. This experiment was conducted on an RTX 2080Ti GPU and an Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz × 40, with PyTorch [31] and FAISS [18].

The results are shown in Table 8. Our XML model is more efficient than all the baselines. Compared to the best baseline methods MEE+MCN, XML is 18×

Table 9: Impact of #retrieved videos on TVR *val* set, VCMR task.

| Model | #retrieved videos | IoU=0.5 | | | | IoU=0.7 | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | R@1 | R@5 | R@10 | R@100 | R@1 | R@5 | R@10 | R@100 |
| XML | 10 | 5.29 | 11.82 | 15.83 | 31.05 | 2.62 | 6.54 | 9.14 | 21.19 |
| | 50 | 5.29 | 11.74 | 15.92 | 35.95 | 2.63 | 6.40 | 9.07 | 22.55 |
| | 100 | 5.28 | 11.73 | 15.90 | 36.16 | 2.62 | 6.39 | 9.05 | 22.47 |
| | 200 | 5.28 | 11.73 | 15.90 | 36.20 | 2.62 | 6.39 | 9.05 | 22.46 |

Table 10: SVMR results on TVR *val* set. Model references: *MCN* [14], *CAL* [8], *MEE* [29], *ExCL* [11]. We show top-2 scores in each column in bold

| Model | w/ video | w/ sub. | IoU=0.5 | | IoU=0.7 | |
|---|---|---|---|---|---|---|
| | | | R@1 | R@5 | R@1 | R@5 |
| Chance | - | - | 3.24 | 12.79 | 0.94 | 4.41 |
| Moment Frequency | - | - | 7.72 | 18.93 | 4.19 | 12.27 |
| TEF-only | - | - | 9.63 | 24.86 | 5.14 | 14.92 |
| MCN | ✓ | ✓ | 13.08 | 39.61 | 5.06 | 20.37 |
| MCN (TEF) | ✓ | ✓ | 16.86 | 40.55 | 7.96 | 21.45 |
| CAL | ✓ | ✓ | 12.07 | 39.52 | 4.68 | 20.17 |
| CAL (TEF) | ✓ | ✓ | 17.61 | 42.08 | 8.07 | 21.40 |
| ExCL | ✓ | ✓ | **31.34** | 47.40 | **14.19** | 28.01 |
| ExCL (TEF) | ✓ | ✓ | 31.31 | 48.54 | **14.34** | 28.89 |
| XML | ✓ | ✓ | 30.75 | **51.20** | 13.41 | **31.11** |
| XML (TEF) | ✓ | ✓ | **31.43** | **51.66** | 13.89 | **31.11** |

faster in retrieval, 4.5× faster in feature encoding and needs 77% less disk space to store the encoded features. Besides, it also has 7.7× higher performance (3.25 *vs.* 0.42, IoU=0.7, R@1, on TVR *test-public* set). Note that MEE+ExCL has very poor *retrieval time* performance (287× slower than XML), as it requires early fusion of context and query features. In comparison, the other 3 methods are able to pre-encode the context features and only perform lightweight query encoding and highly optimized nearest neighbor search or matrix multiplication to obtain the moment predictions.

**Impact of #Retrieved Videos**. In previous experiments, we fix the number of videos retrieved by XML to be 100 for corpus level moment retrieval experiments. To study the impact of this hyperparameter, we perform experiments when #videos $\in [10, 50, 100, 200]$, the results are shown in Table 9. Overall, we notice XML is not sensitive to the number of retrieved videos in terms of R@1, R@5 and R@10 (IoU=0.5, 0.7) in the tested range. When we focus on R@100, IoU=0.5, we find that using more videos helps improve the retrieval performance.

### B.2 SVMR and Video Retrieval Experiments

**Single Video Moment Retrieval**. Table 10 shows the Single Video Moment Retrieval (SVMR) results on TVR *val* set. The goal of the task is to retrieve

Table 11: Video retrieval results on TVR *val* set. Model references: *MCN* [14], *CAL* [8], *MEE* [29]

| Model | w/ video | w/ sub. | R@1 | R@5 | R@10 | R@100 |
|---|---|---|---|---|---|---|
| Chance | - | - | 0.03 | 0.22 | 0.47 | 4.61 |
| MCN | ✓ | ✓ | 0.05 | 0.38 | 0.66 | 3.59 |
| MCN (TEF) | ✓ | ✓ | 0.07 | 0.28 | 0.51 | 3.93 |
| CAL | ✓ | ✓ | 0.28 | 1.02 | 1.68 | 8.55 |
| CAL (TEF) | ✓ | ✓ | 0.06 | 0.34 | 0.63 | 5.26 |
| MEE | ✓ | ✓ | 7.56 | 20.78 | 29.88 | 73.07 |
| XML | ✓ | ✓ | **16.54** | **38.11** | **50.41** | **88.22** |
| XML (TEF) | ✓ | ✓ | **16.08** | **37.92** | **50.38** | **88.62** |

relevant moments from a single video rather than from a video corpus as in VCMR. We observe XML achieves comparable performance with the state-of-the-art method ExCL [11]. However, note that XML significantly outperforms ExCL on the VCMR task with higher efficiency, as stated in Sec. 5.2 and Sec. B.1. We also noticed that adding TEF has minimal impact on the performance of XML and ExCL, while greatly improves MCN's and CAL's performance. This is not surprising as XML and ExCL directly model the complete video where the temporal information could be acquired, while MCN and CAL break the video into separate proposals where the temporal information is lost in the process.

**Video Retrieval**. Table 11 shows the Video Retrieval results on TVR *val* set. The goal of the task is to retrieve relevant videos from a large corpus. As MCN and CAL do not perform whole-video retrieval, we approximate their video retrieval predictions using the videos associated with the top-retrieved moments, as in [8]. MCN and CAL models perform rather poor (>50x lower performance than XML, R@1) on the video retrieval task, we summarize some possible reasons here: (1) MCN and CAL's video retrieval results are only an approximation as they are trained to differentiate moments rather than videos; (2) they need to rank a large number of proposals (187K proposals in TVR *val* set), which has many drawbacks, e.g., inefficient negative sampling in training. MEE gets less than half of XML's performance as it uses global pooled context features instead of more fine-grained local context features as XML.

### B.3 More Qualitative Examples

We show more qualitative examples from our XML model in Fig. 20 and Fig. 21. We show top-3 predictions for the VCMR task, as well as associated predictions (with ConvSE filter responses) for the SVMR task.

## C TVR DiDeMo Experiments

To show the effectiveness of XML for the VCMR task, we also tested it on the popular moment retrieval dataset DiDeMo [14]. Different from TVR experiments,

Table 12: VCMR results on DiDeMo [14] *test* set. Model references: *MCN* [14], *CAL* [8], *MEE* [29]. This table includes models trained with Temporal Endpoint Feature (TEF) [14]. We show top scores in each column in bold

| Model | w/ video | IoU=0.5 | | | IoU=0.7 | | |
|---|---|---|---|---|---|---|---|
| | | R@1 | R@10 | R@100 | R@1 | R@10 | R@100 |
| Chance | - | 0.00 | 0.10 | 1.99 | 0.00 | 0.02 | 0.64 |
| Frequency | - | 0.02 | 0.22 | 2.34 | 0.02 | 0.17 | 1.99 |
| **Proposal based Methods** | | | | | | | |
| TEF-only | - | 0.05 | 0.32 | 2.58 | 0.03 | 0.27 | 2.12 |
| MCN (TEF) | ✓ | 0.88 | 5.16 | 26.23 | 0.58 | 4.12 | 21.03 |
| CAL (TEF) | ✓ | 0.97 | 6.15 | 28.06 | 0.66 | 4.69 | 22.89 |
| **Retrieval + Re-ranking** | | | | | | | |
| MEE+MCN (TEF) | ✓ | 0.53 | 3.00 | 6.52 | 0.46 | 2.64 | 6.37 |
| MCN+MCN (TEF) | ✓ | 0.92 | 4.83 | 17.50 | 0.64 | 3.67 | 13.12 |
| CAL+CAL (TEF) | ✓ | 1.07 | 6.45 | 22.60 | 0.72 | 4.86 | 17.60 |
| CAL+CAL (TEF,re-train) | ✓ | 1.29 | 6.71 | 22.51 | 0.85 | 4.95 | 17.73 |
| Approx. CAL+CAL (TEF,re-train) | ✓ | 1.27 | 6.39 | 15.82 | 0.80 | 4.95 | 11.59 |
| XML (TEF) | ✓ | **2.26** | **10.42** | **34.49** | **1.59** | **6.71** | **25.44** |

we only use ResNet features for DiDeMo. Besides, we also switch off the subtitle stream as DiDeMo has only video context. The results are shown in Table 12. The baseline results are directly taken from [8]. We observe XML outperforms all the baseline methods on DiDeMo dataset by a large margin, showing XML is able to generalize well to datasets where only video is available.

## D TVC Dataset and Experiments

After the TVR data collection, we extended TVR by collecting extra descriptions for each annotated moment. This dataset, named TV show Captions (**TVC**), is a large-scale multimodal video captioning dataset. Fig. 16 shows two TVC examples. Similar to TVR, the TVC task requires systems to gather information from both video and subtitle to generate relevant descriptions. In the following, we present a brief analysis and initial baselines for TVC.

### D.1 Data Collection and Analysis

To promote better coverage of the video (subtitle) content, we encourage annotators to write descriptions that are of different types from existing ones, e.g., we encourage annotators to write *video-only* and *video+sub* type descriptions if there already exists a *sub-only* description. For each moment in the TVR training set, we collect one extra description, together with the original description forms the TVC training set with 2 descriptions for each moment. For each moment in TVR val/test sets, we collect 4 extra descriptions as the TVC val/test sets. The original val/test descriptions in TVR are not used to ensure data integrity. Details regarding data split are presented in Sec. E.

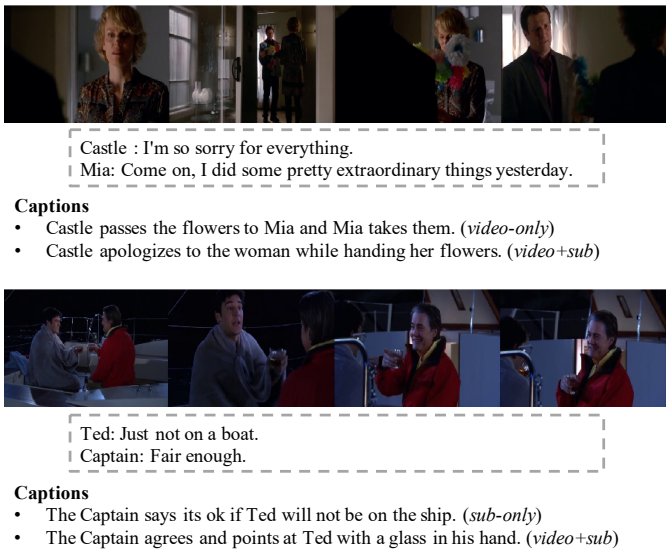


Fig. 16: TVC caption description examples. Each caption description is followed by a *description type tag*. Text inside *dashed boxes* is the subtitles associated with the moments. For brevity, here we only show sampled frames from the moments

Table 13: Comparison of TVC with existing video captioning datasets. *Desc. context* = Description context, it indicates which modality the descriptions are related to

| **Dataset** | **Domain** | **#Moment** | **#Desc.** | **#Desc. per moment** | **Desc. context** video | text | **Desc. type anno.** |
|---|---|---|---|---|---|---|---|
| TACoS-MLevel [34] | Cooking | 25K | 75K | 3 | ✓ | - | - |
| YouCook II [51] | Cooking | 15.4K | 15.4K | 11 | ✓ | - | - |
| ANetCap [22] | Activity | 100K | 100K | 1 | ✓ | - | - |
| Charades [37] | Indoor | 10K | 27.8K | 2-3 | ✓ | - | - |
| VATEX [42] | Activity | 41K | 826K | 20 | ✓ | - | - |
| LSMDC [35] | Movie | 128K | 128K | 1 | ✓ | - | - |
| MST-VTT [44] | Open | 10k | 200k | 20 | ✓ | - | - |
| TVC | TV show | 108K | 262K | 2-4 | ✓ | ✓ | ✓ |

Table 13 gives an overview of TVC and its comparison with recent video captioning datasets. In total, TVC contains 262K descriptions paired with 108K moments. TVC is unique as its captions may also describe dialogues/subtitles while the captions in the other datasets are only describing the visual content. TVC also has a description type annotation, which can be used for model training and analysis. Fig. 17 compares the description type distribution between TVR and TVC. As we encouraged annotators to write different types of descriptions, the description type distribution is more balanced in TVC compared to that of TVR. As TVC is built on top of TVR, it shares many properties of TVR, e.g.,

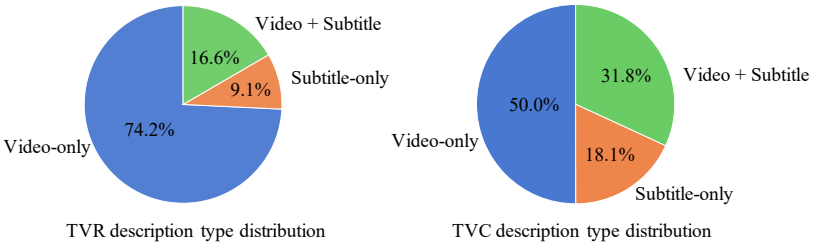


Fig. 17: Description type distributions of TVR and TVC

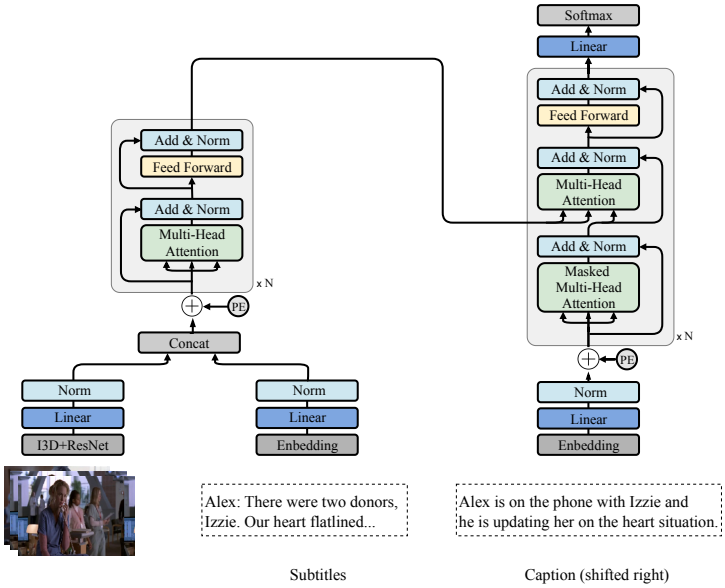


Fig. 18: Overview of the MultiModal Transformer (MMT) model for the TVC task. *PE* stands for Positional Encoding

great linguistic diversity, rich inter-human interactions, more actions and people in a single description, etc. See Sec. 3 for more details.

### D.2 Multimodal Transformer

To provide a strong initial baseline for the TVC multimodal video captioning task, we designed a MultiModal Transformer (**MMT**) captioning model which follows the classical encoder-decoder transformer architecture [40]. It takes both video and subtitle as encoder inputs to generate the captions from the decoder. Fig. 18 gives an overview of the designed model.

**Input Representation**. We use the concatenation of I3D [2] feature and ResNet-152 [13] feature to represent videos. The features are pre-processed in the same

Table 14: Model comparison on TVC *test-public* set, with different input context

| Model | B@4 | METEOR | Rouge-L | CIDEr-D |
|---|---|---|---|---|
| MMT (sub) | 6.33 | 13.92 | 7.73 | 33.76 |
| MMT (video) | 9.98 | 15.23 | 30.44 | 36.07 |
| MMT (video+sub) | **10.87** | **16.91** | **32.81** | **45.38** |

Table 15: Feature ablation on TVC *val* set. All the models use both videos and subtitles

| Model | B@4 | METEOR | Rouge-L | CIDEr-D |
|---|---|---|---|---|
| MMT (ResNet) | 9.92 | 16.24 | 31.76 | 43.94 |
| MMT (I3D) | 10.25 | 16.48 | 31.98 | 43.70 |
| MMT (ResNet+I3D) | **10.53** | **16.61** | **32.35** | **44.39** |

way as our XML model for the TVR task, as in Sec. 4.1. To represent subtitles, we use trainable 300D word embeddings. Next, we project raw video features and subtitle word features into a common embedding space using linear layers and layernorm [1] layers. The projected video embedding $E^v \in \mathcal{R}^{l_v \times d}$ and subtitle embedding $E^s \in \mathcal{R}^{l_s \times d}$ are then concatenated at length dimension [23] as the input to the encoder: $E^{ctx} = [E^v; E^s]$, where $E^{ctx} \in \mathcal{R}^{(l_v+l_s)\times d}$ stands for the context embedding, $d$ is hidden size.

**Encoder and Decoder**. Both the encoder and decoder follows the standard design [40] with 2 layers, i.e., $N$=2. The decoder access encoder outputs at each layer with a multi-head attention [40]. We refer readers to [40] for a more detailed explanation of the model architecture.

**Training and Inference**. We train the model using Maximum Likelihood Estimation (MLE), i.e., we maximize the likelihood of generating the ground truth words. At inference, we use greedy decoding instead of beam search as it performs better in our experiments.

### D.3 Experiments

We use the same video split for TVC as in TVR, see Sec. E for more details. We report numbers on standard metrics, inlcuding BLEU@4 [30], METEOR [6], Rouge-L [26], CIDEr-D [41]. We first compare MMT models with different input modalities. The results are shown in Table 14. Across all metrics, the model with both videos and subtitles performs better than the models with only one of them, which shows both videos and subtitles are important for describing the moments. Next, we compare models with different visual features. The results are shown in Table 15. Models with both appearance features (ResNet-152 [13]) and motion feature (I3D [2]) performs better than only using one of them.

Table 16: TVR data split detail

| Split | #queries | #moments | #videos |
|---|---|---|---|
| *train* | 87,175 | 87,175 | 17,435 |
| *val* | 10,895 | 10,895 | 2,179 |
| *test-public* | 5,445 | 5,445 | 1,089 |
| *test-private* | 5,450 | 5,450 | 1,090 |
| *total* | 108,965 | 108,965 | 21,793 |

Table 17: TVC data split detail

| Split | #desc. | #moments | #videos | #desc./moment |
|---|---|---|---|---|
| *train* | 174,350 | 86,603 | 17,435 | 2 |
| *val* | 43,580 | 10,481 | 2,179 | 4 |
| *test-public* | 21,780 | 5,420 | 1,089 | 4 |
| *test-private* | 21,800 | 5,422 | 1,090 | 4 |
| *total* | 261,510 | 107,926 | 21,793 | - |

### D.4 Qualitative Examples

We show qualitative examples of MMT in Fig. 22, with generated captions by the three MMT models trained with different input context.

## E Data Release and Public Leaderboards

Both TVR and TVC are publicly available at their websites: `https://tvr.cs.unc.edu`, `https://tvr.cs.unc.edu/tvc.html`. With the datasets, we host public leaderboards to better compare the systems. In the following, we describe data split and usage in detail.

We split TVR into 80% *train*, 10% *val*, 5% *test-public* and 5% *test-private* such that videos and their associated queries appear in only one split. This setup is the same as TVQA [24]. Details of the splits are presented in Table 16. *test-public* will be used for a public leaderboard, *test-private* is reserved for future challenges. *val* set should only be used for parameter tuning, it should not be used in the training process, including but not limited to pre-train the language features.

TVC follows the same data split as TVR, but with a different number of descriptions per moment, i.e., each of the training moments are paired with 2 descriptions while each of the moments in other splits are paired with 4 descriptions. Details are presented in Table 17. The rules on split usage are also the same as TVR.


Fig. 19: Comparison of TVR with existing moment retrieval datasets [32,9,22,14]. Ground truth moment is shown in *green box*. TVR videos are typically more diverse, containing more camera viewpoints, activities and people, etc.

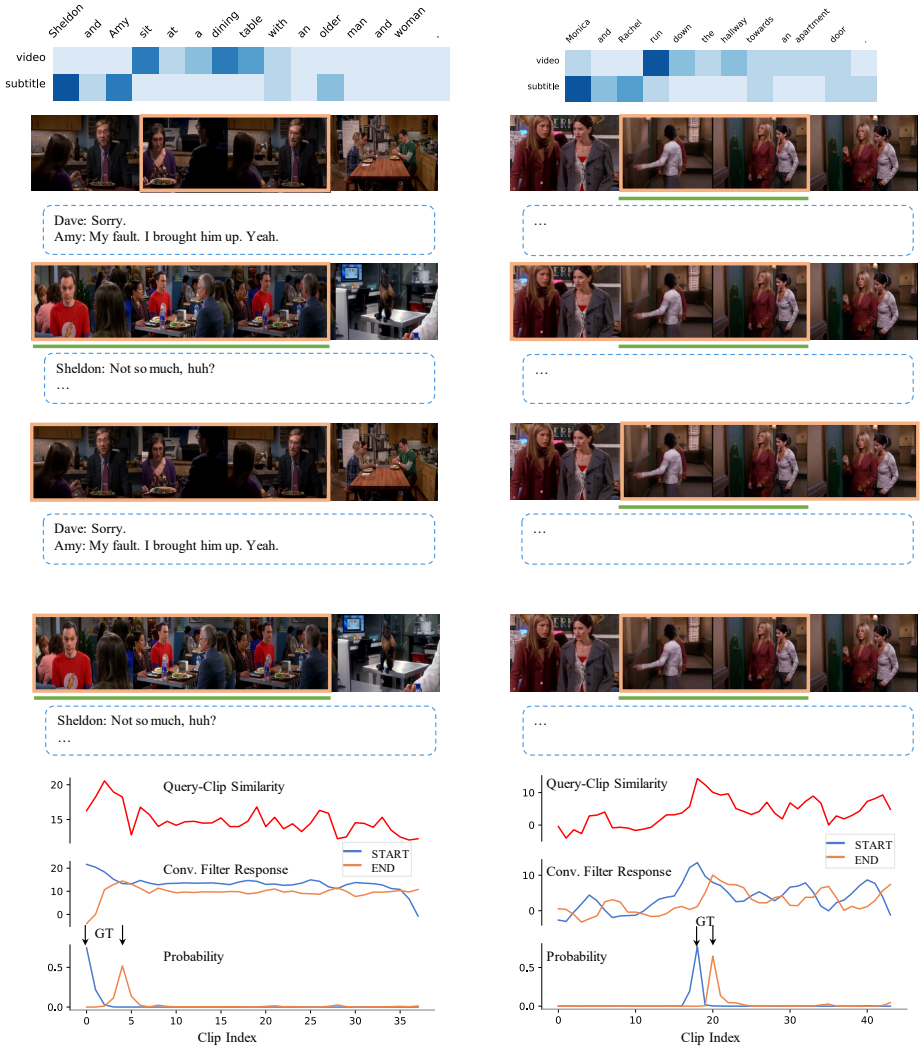


Fig. 20: Qualitative examples of XML. We show top-3 retrieved moments for VCMR (*top*) and SVMR results (*bottom*, with convolution filter responses) for each query. Text inside *dashed boxes* is the subtitles with the predicted moments. *Orange box* shows the predictions, *green bar* shows the ground truth. Best viewed in color


Fig. 21: Qualitative examples of XML. We show top-3 retrieved moments for VCMR (*top*) and SVMR results (*bottom*, with convolution filter responses) for each query. Text inside *dashed boxes* is the subtitles with the predicted moments. *Orange box* shows the predictions, *green bar* shows the ground truth. Best viewed in color

Beckett: No, Castle, I'm talking about my life. I don't know what to do about my life...

**Ground-Truth Captions**
- Beckett rolls over to lay on her back. (*video-only*)
- Beckett is visibly worried and speaks to Castle while they are in bed. (*video-only*)
- Castle and Beckett discuss her vacation when they're in bed. (*video-text*)
- Beckett is talking to Castle about losing her job. (*text-only*)

**Generated Captions:**
- Beckett tells Castle that she is going to be honest. (model: *sub*)
- Beckett and Castle are in bed and they are in bed. (model: *video*)
- Beckett and Castle are in bed together, and Beckett is sleeping in bed. (model: *video + sub*)

Cameron: or should we just start running a thousand different tox screens? …

**Ground-Truth Captions**
- House grabs a file and opens it up. (*video-only*)
- House picks a file, place it on a table then opens it. (*video-only*)
- House opens a file and says he won't read it. (*video-text*)
- House jokes about tox screens when Foreman suggests it's something different. (*video-text*)

**Generated Captions:**
- House and Cameron are having a conversation with each other. (model: *sub*)
- House puts a red box on the table and takes a red coffee cup. (model: *video*)
- House picks up a red mug and takes it off. (model: *video + sub*)

Sheldon: ...with Adamantium like Wolverine.
Penny: Are they working on that?...

**Ground-Truth Captions**
- Sheldon holds out a large pile of cash with his right hand in front of Sheldon. (*video-only*)
- Sheldon is holding something in his hand out to Penny. (*video-only*)
- Astonished, Penny makes a question, to which Sheldon gives a serious answer while presenting her with money. (*video-text*)
- Penny questions Sheldon as to whether somebody is trying something. (*text-only*)

**Generated Captions:**
- Sheldon asks penny if she is feeling like a certain way. (model: *sub*)
- Sheldon tells penny that she is not sure. (model: *video*)
- Sheldon is standing in front of penny as he speaks to her. (model: *video + sub*)

Rachel: You've been here for two months now. And your boss is required to hand in a performance evaluation.

**Ground-Truth Captions**
- Tag prepares to leave before being reeled back in by Rachel. (*video-only*)
- As Tag is leaving Rachel tells him about his evaluation because of how long he has been there. (*video-text*)
- Tag is surprised to learn that Rachel will evaluate him. (*video-text*)
- Rachel tells Tag that he will put his performance into his evaluation. (*text-only*)

**Generated Captions:**
- Tag tells Rachel that he has been in his office. (model: *sub*)
- Rachel walks into the office and picks up a book. (model: *video*)
- Tag walks into Rachel's office and hands her a file. (model: *video + sub*)

Alexis: Once we realized Mandy hadn't really cheated on him.

**Ground-Truth Captions**
- Alexis runs her hand through her hair when Castle is looking at her. (*video-only*)
- Alexis fixes her hair as she speaks to Castle beside her. (*video-only*)
- Alexis rubs her hair when Castle is looking at her. (*video-only*)
- The girl adjust her hair while Castle stares at her. (*video-only*)

**Generated Captions:**
- Alexis and castle walk into the room together. (model: *sub*)
- Beckett and castle are talking to each other. (model: *video*)
- Alexis and castle stand in front of each other as they stand in front of each other. (model: *video + sub*)

Chandler: We could trade later.
Monica: Yeah, I'm good…

**Ground-Truth Captions**
- Monica and Chandler touch their babies together in the hospital. (*video-only*)
- Monica is trying to reposition the baby in her arms. (*video-only*)
- Chandler and Monica hold their babies close by then decide to keep them. (*video-text*)
- Monica and Chandler try to figure out how to swap holding babies. (*video-only*)

**Generated Captions:**
- Chandler and Monica walk into the apartment and chandler closes the door. (model: *sub*)
- Monica and Chandler are holding baby Emma as they are in the baby room. (model: *video*)
- Monica hands Chandler a towel and he takes it and then chandler picks it up. (model: *video + sub*)

Fig. 22: Qualitative comparison of MMT. Text inside *dashed boxes* is the subtitles associated with the moments. Each ground-truth caption description is followed by a description type tag. We show comparison among models trained with only videos (*video*), subtitles (*sub*), or both (*video + sub*)

## References


1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
2. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017)
3. Chen, D., Fisch, A., Weston, J., Bordes, A.: Reading wikipedia to answer open-domain questions. In: ACL (2017)
4. Chen, J., Chen, X., Ma, L., Jie, Z., Chua, T.S.: Temporally grounding natural sentence in video. In: EMNLP (2018)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
6. Denkowski, M., Lavie, A.: Meteor universal: Language specific translation evaluation for any target language. In: Proceedings of the ninth workshop on statistical machine translation (2014)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
8. Escorcia, V., Soldan, M., Sivic, J., Ghanem, B., Russell, B.: Temporal localization of moments in video collections with natural language. arXiv preprint arXiv:1907.12763 (2019)
9. Gao, J., Sun, C., Yang, Z., Nevatia, R.: Tall: Temporal activity localization via language query. In: ICCV (2017)
10. Ge, R., Gao, J., Chen, K., Nevatia, R.: Mac: Mining activity concepts for language-based temporal localization. In: WACV (2019)
11. Ghosh, S., Agarwal, A., Parekh, Z., Hauptmann, A.: Excl: Extractive clip localization using natural language descriptions. In: NAACL (2019)
12. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS (2011)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
14. Hendricks, L.A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., Russell, B.: Localizing moments in video with natural language. In: ICCV (2017)
15. Hendricks, L.A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., Russell, B.: Localizing moments in video with temporal language. In: EMNLP (2018)
16. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al.: Gradient flow in recurrent nets: the difficulty of learning long-term dependencies (2001)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (1997)
18. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with gpus. IEEE Transactions on Big Data (2019)
19. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
20. Kazemzadeh, S., Ordonez, V., Matten, M., Berg, T.: Referitgame: Referring to objects in photographs of natural scenes. In: EMNLP (2014)
21. Kim, K.M., Heo, M.O., Choi, S.H., Zhang, B.T.: Deepstory: Video story qa by deep embedded memory networks. In: IJCAI (2017)
22. Krishna, R., Hata, K., Ren, F., Fei-Fei, L., Niebles, J.C.: Dense-captioning events in videos. In: ICCV (2017)

23. Lei, J., Wang, L., Shen, Y., Yu, D., Berg, T.L., Bansal, M.: Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning. In: ACL (2020)
24. Lei, J., Yu, L., Bansal, M., Berg, T.L.: Tvqa: Localized, compositional video question answering. In: EMNLP (2018)
25. Lei, J., Yu, L., Berg, T.L., Bansal, M.: Tvqa+: Spatio-temporal grounding for video question answering. In: ACL (2020)
26. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: ACL (2004)
27. Lin, T., Zhao, X., Su, H., Wang, C., Yang, M.: Bsn: Boundary sensitive network for temporal action proposal generation. In: ECCV (2018)
28. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
29. Miech, A., Laptev, I., Sivic, J.: Learning a text-video embedding from incomplete and heterogeneous data. arXiv preprint arXiv:1804.02516 (2018)
30. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: ACL (2002)
31. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: NIPS Autodiff Workshop (2017)
32. Regneri, M., Rohrbach, M., Wetzel, D., Thater, S., Schiele, B., Pinkal, M.: Grounding action descriptions in videos. TACL (2013)
33. Roerdink, J.B., Meijster, A.: The watershed transform: Definitions, algorithms and parallelization strategies. Fundamenta informaticae (2000)
34. Rohrbach, A., Rohrbach, M., Qiu, W., Friedrich, A., Pinkal, M., Schiele, B.: Coherent multi-sentence video description with variable level of detail. In: GCPR (2014)
35. Rohrbach, A., Torabi, A., Rohrbach, M., Tandon, N., Pal, C., Larochelle, H., Courville, A., Schiele, B.: Movie description. IJCV (2017)
36. Seo, M., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. In: ICLR (2017)
37. Sigurdsson, G.A., Varol, G., Wang, X., Farhadi, A., Laptev, I., Gupta, A.: Hollywood in homes: Crowdsourcing data collection for activity understanding. In: ECCV (2016)
38. Szeliski, R.: Computer vision: algorithms and applications. Springer Science & Business Media (2010)
39. Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., Fidler, S.: Movieqa: Understanding stories in movies through question-answering. In: CVPR (2016)
40. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
41. Vedantam, R., Lawrence Zitnick, C., Parikh, D.: Cider: Consensus-based image description evaluation. In: CVPR (2015)
42. Wang, X., Wu, J., Chen, J., Li, L., Wang, Y.F., Wang, W.Y.: Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In: ICCV (2019)
43. Xu, H., He, K., Plummer, B.A., Sigal, L., Sclaroff, S., Saenko, K.: Multilevel language and vision integration for text-to-clip retrieval. In: AAAI (2019)
44. Xu, J., Mei, T., Yao, T., Rui, Y.: Msr-vtt: A large video description dataset for bridging video and language. In: CVPR (2016)
45. Yu, A.W., Dohan, D., Luong, M.T., Zhao, R., Chen, K., Norouzi, M., Le, Q.V.: Qanet: Combining local convolution with global self-attention for reading comprehension. In: ICLR (2018)

46. Yu, L., Lin, Z., Shen, X., Yang, J., Lu, X., Bansal, M., Berg, T.L.: Mattnet: Modular attention network for referring expression comprehension. In: CVPR (2018)
47. Zellers, R., Bisk, Y., Farhadi, A., Choi, Y.: From recognition to cognition: Visual commonsense reasoning. In: CVPR (2019)
48. Zhang, D., Dai, X., Wang, X., fang Wang, Y., Davis, L.S.: Man: Moment alignment network for natural language moment retrieval via iterative graph adjustment. In: CVPR (2018)
49. Zhang, Z., Lin, Z., Zhao, Z., Xiao, Z.: Cross-modal interaction networks for query-based moment retrieval in videos. In: SIGIR (2019)
50. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., Lin, D.: Temporal action detection with structured segment networks. In: ICCV (2017)
51. Zhou, L., Xu, C., Corso, J.J.: Towards automatic learning of procedures from web instructional videos. In: AAAI (2018)