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. Additionally, 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.[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 [13,8] 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.* [7] 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 [13,25,8,19] 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 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.

[1] TVR dataset and code are publicly available: https://tvr.cs.unc.edu/. We also introduce TVC for multimodal captioning at https://tvr.cs.unc.edu/tvc.html.

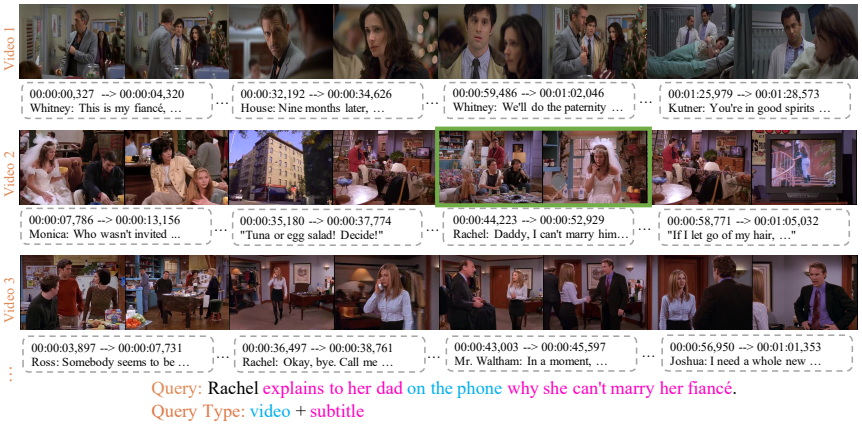

Fig. 1: A TVR example in the VCMR task. Ground truth moment is shown in *green box*. Colors in the query indicate whether the words are related to video (*blue*) or subtitle (*magenta*) or both (*black*). To better retrieve relevant moments from the video corpus, a system needs to comprehend both videos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

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 [30,18,20] 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 [13]) 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 [13,25,8,19], 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 [13,8,14,32,9,7] explore the moment retrieval task as a ranking problem over a predefined set of moment proposals. These proposals are usually generated using handcrafted heuristics [13,14] or sliding windows [8,32,9,7] 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 [10,21] adapt start-end span predictors [28,3] 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 [10] 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 [10] has shown promising results in SVMR, it is not scalable to VCMR as it uses early fusion. Consider retrieving N queries in a corpus of M videos. This requires running several layers of LSTM [15] 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 [37] to progressively group top-scored clips. However, these methods rely on handcrafted rules and are not trainable. Inspired by image edge detectors [29], 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 [13,7,10], and does this with better computational efficiency.

To summarize, our contributions are 2-fold: (*i*) We introduce **TVR** dataset, a large-scale multimodal moment retrieval dataset with 109K high-quality queries of great linguistic diversity.[2] (*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. Comprehensive experiments and analyses show XML surpasses all presented baselines by a large margin and runs with better efficiency.

2 Related Work

The goal of natural language-based moment retrieval is to retrieve relevant moments from a single video [13,8] or from a large video corpus [7]. In the following, we present a brief overview of the community efforts on these tasks and make distinctions between existing works and ours.

[2] We also collected a new multimodal captioning dataset with 262K captions, named as TV show Caption (**TVC**) `https://tvr.cs.unc.edu/tvc.html`

Datasets. Several datasets have been proposed for the task, e.g., DiDeMo [13], ActivityNet Captions [19], CharadesSTA [8], and TACoS [25], 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 [13]. For example, moments in DiDeMo [13] 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 [25,8,19]. While it is natural for people to use temporal connectives (e.g., 'first', 'then') and anaphora (e.g., pronouns) [27] 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 [13,8,14,32,9,7] pose moment retrieval as ranking a predefined set of moment proposals. These proposals are typically generated with handcrafted rules [13,14] or sliding windows [8,32,9,7]. Typically, such proposals are not temporally precise and are not scalable to long videos due to high computational cost. [8,32,9] 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 [28,3] and action localization [22], 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 [7]. Proposal based approaches MCN [13] and CAL [7] 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 [35,4,36] 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 [20]. 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 [17,13] 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 [20], 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:[3]

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

[3] We present a pipeline figure of our data collection procedure in the supplementary.

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 [25]	Cooking	16.2K / 0.1K	2K	10.5	5.9 / 287	✓	-	✓	-	-
DiDeMo [13]	Flickr	41.2K / 10.6K	7.6K	8.0	6.5 / 29.3	✓	-	-	-	✓
ActivityNet Captions [19]	Activity	72K / 15K	12.5K	14.8	36.2 / 117.6	✓	-	✓	-	-
CharadesSTA [8]	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	✓	✓	✓	✓	✓

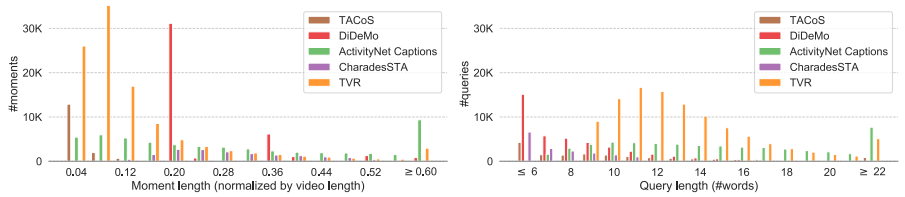

Fig. 2: Distributions of moment (*left*) and query (*right*) lengths. Compared to existing moment retrieval datasets [25,13,19,8], TVR has relatively shorter moments (normalized) and longer queries. Best viewed digitally with zoom

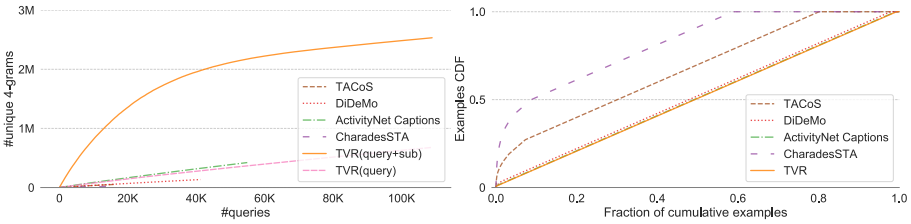

Fig. 3: *Left*: #unique 4-gram as a function of #queries. *Right*: CDF of queries ordered by frequency, to obtain this plot, we sampled 10K queries from each dataset, we consider two queries to be the same if they exact match, after tokenization and lemmatization, following [34]. Compared to existing moment retrieval datasets [25,13,19,8], TVR has greater diversity, i.e., it has more unique 4-grams and almost every TVR query is unique. Best viewed digitally with zoom

3.2 Data Analysis and Comparison

Table 1 shows an overview of TVR and its comparisons with existing moment retrieval datasets [25,8,19,13]. 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

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 [25]	20	0	She **rinses** the peeled carrots off in the sink. (*video*) The person **removes** roots and outer leaves and **rewashes** the leek. (*video*)
CharadesSTA [8]	6	12	A person is **eating** food slowly. (*video*) A person is **opening** the door to a bedroom. (*video*)
ActivityNet Caption [19]	44	44	He then **grabs** a metal mask and **positions** himself correctly on the floor. (*video*) The same man **comes** back and **lifts** the weight over his head again. (*video*)
DiDeMo [13]	6	10	A dog **shakes** its body. (*video*) 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*) Taub **argues** with the patient that fighting in Hockey **undermines** the sport. (*sub*) Chandler **points** at Joey while **describing** a woman who wants to **date** him. (*video+sub*)

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 (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 [12] features at 3FPS and max-pool the features every 1.5 seconds to get a clip-level feature.

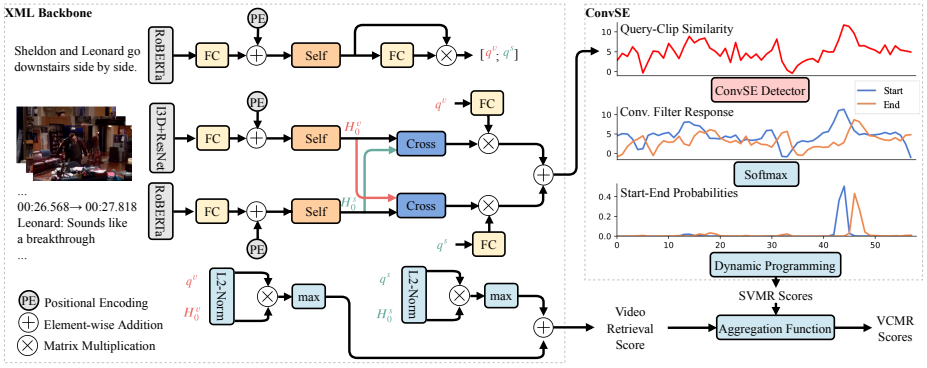

Fig. 4: Cross-modal Moment Localization (XML) model overview. *Self=Self Encoder*, *Cross=Cross Encoder*. We describe *XML Backbone* in Sec. 4.1, *ConvSE* module in Sec. 4.2 and show XML's training and inference procedure in Sec. 4.3

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 [16] 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 [23]. Specifically, we first fine-tune RoBERTa using the queries and subtitle sentences in TVR train-split with MLM objective [6], then fix the parameters to extract contextualized token embeddings from its second-to-last layer [21]. 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 [11]. We then add learned positional encoding [6] 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 [31] layer and a linear layer, with a residual [12] 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 $\mathbf{q}^m \in \mathbb{R}^d$ [33]:

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

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 [31] 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 [37] to progressively group top-scored regions. However, both methods require handcrafted rules and are not trainable. Inspired by edge detectors in image processing [29], 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 [33]:

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

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_{st} and t^i_{ed} are the ground-truth indices. At inference, predictions can be generated from the probabilities in linear time using dynamic programming [28]. 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 λ 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 α are used to balance the importance of the two scores. A higher α encourages more moments from top retrieved videos. Empirically, we find α=20 works well. At inference, for each query, we first retrieve the top 100 videos based on s^{vr}, then rank all the moments in the 100 videos by s^{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.

Metrics. Following [7,8], 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

Table 3: Baseline comparison on TVR *test-public* set, VCMR task. Model references: *MCN* [13], *CAL* [7], *MEE* [24], *ExCL* [10]

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**

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. The time spent on data loading, pre-processing, 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 the supplementary file, we include the following 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) [13] model results; (4) feature and model ablation studies; (5) VCMR results on DiDeMo [13] dataset, etc.

Proposal based Methods. MCN [13] and CAL [7] 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.

Retrieval+Re-ranking Methods. We also compare to methods under the retrieval+re-ranking setting [7] where we first retrieve a set of candidate videos

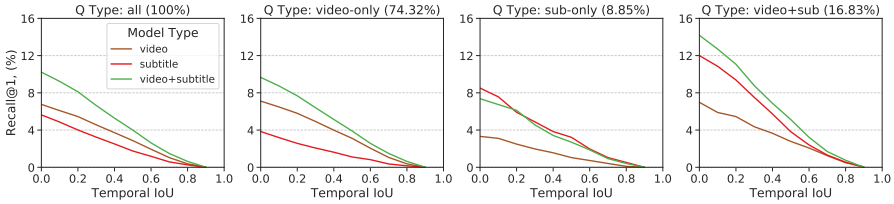

Fig. 5: Performance breakdown of XML models that use only *video*, *subtitle*, or both as inputs, by different query types (with percentage of queries shown in brackets). The performance is evaluated on TVR *val* set for VCMR

using a given method and then re-rank the moment predictions in the candidate videos using another method. Specifically, we first use MEE [24] 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 [10] 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 [7,37] 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 supplementary file, 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, with proposal confidence scores calculated as the average of scores inside each proposal region. On average, it produces 87 proposals per video. The proposals

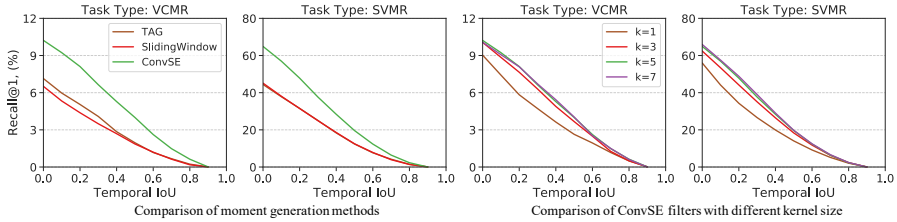

Fig. 6: ConvSE Analysis. *Left*: comparison of moment generation methods. *Right*: comparison of ConvSE filters with different kernel sizes (k)

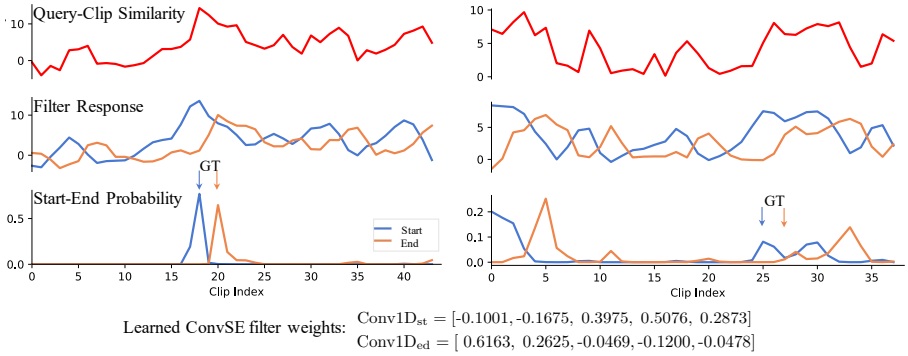

Fig. 7: Examples of learned ConvSE filters applying on query-clip similarity scores. Ground truth span is indicated by the two arrows labeled by *GT*. Note the two filters output stronger responses on the up (*Start*) and down (*End*) edges

used here are the same as the ones used for MCN and CAL in our previous experiments; (2) TAG [37] that progressively groups top-scored clips with the classical watershed algorithm [26]. 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 curves and thus detect them. Interestingly, the learned weights Conv1D$_{st}$ and Conv1D$_{ed}$ in Fig. 7 are similar to the edge detectors in image processing [29].

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

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.

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. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
7. 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)
8. Gao, J., Sun, C., Yang, Z., Nevatia, R.: Tall: Temporal activity localization via language query. In: ICCV (2017)
9. Ge, R., Gao, J., Chen, K., Nevatia, R.: Mac: Mining activity concepts for language-based temporal localization. In: WACV (2019)
10. Ghosh, S., Agarwal, A., Parekh, Z., Hauptmann, A.: Excl: Extractive clip localization using natural language descriptions. In: NAACL (2019)
11. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS (2011)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
13. Hendricks, L.A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., Russell, B.: Localizing moments in video with natural language. In: ICCV (2017)
14. Hendricks, L.A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., Russell, B.: Localizing moments in video with temporal language. In: EMNLP (2018)
15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (1997)
16. 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)
17. Kazemzadeh, S., Ordonez, V., Matten, M., Berg, T.: Referitgame: Referring to objects in photographs of natural scenes. In: EMNLP (2014)
18. Kim, K.M., Heo, M.O., Choi, S.H., Zhang, B.T.: Deepstory: Video story qa by deep embedded memory networks. In: IJCAI (2017)
19. Krishna, R., Hata, K., Ren, F., Fei-Fei, L., Niebles, J.C.: Dense-captioning events in videos. In: ICCV (2017)
20. Lei, J., Yu, L., Bansal, M., Berg, T.L.: Tvqa: Localized, compositional video question answering. In: EMNLP (2018)
21. Lei, J., Yu, L., Berg, T.L., Bansal, M.: Tvqa+: Spatio-temporal grounding for video question answering. In: ACL (2020)
22. Lin, T., Zhao, X., Su, H., Wang, C., Yang, M.: Bsn: Boundary sensitive network for temporal action proposal generation. In: ECCV (2018)
23. 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)

24. Miech, A., Laptev, I., Sivic, J.: Learning a text-video embedding from incomplete and heterogeneous data. arXiv preprint arXiv:1804.02516 (2018)
25. Regneri, M., Rohrbach, M., Wetzel, D., Thater, S., Schiele, B., Pinkal, M.: Grounding action descriptions in videos. TACL (2013)
26. Roerdink, J.B., Meijster, A.: The watershed transform: Definitions, algorithms and parallelization strategies. Fundamenta informaticae (2000)
27. 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)
28. Seo, M., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. In: ICLR (2017)
29. Szeliski, R.: Computer vision: algorithms and applications. Springer Science & Business Media (2010)
30. Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., Fidler, S.: Movieqa: Understanding stories in movies through question-answering. In: CVPR (2016)
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
32. 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)
33. 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)
34. Zellers, R., Bisk, Y., Farhadi, A., Choi, Y.: From recognition to cognition: Visual commonsense reasoning. In: CVPR (2019)
35. 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)
36. Zhang, Z., Lin, Z., Zhao, Z., Xiao, Z.: Cross-modal interaction networks for query-based moment retrieval in videos. In: SIGIR (2019)
37. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., Lin, D.: Temporal action detection with structured segment networks. In: ICCV (2017)