Multi-modal Transformer for Video Retrieval

Valentin Gabeur[1,2], Chen Sun[2], Karteek Alahari[1], Cordelia Schmid[2]

[1] Inria*, karteek.alahari@inria.fr
[2] Google Research, {valgab,chensun,cordelias}@google.com

Abstract. The task of retrieving video content relevant to natural language queries plays a critical role in effectively handling internet-scale datasets. Most of the existing methods for this caption-to-video retrieval problem do not fully exploit cross-modal cues present in video. Furthermore, they aggregate per-frame visual features with limited or no temporal information. In this paper, we present a multi-modal transformer to jointly encode the different modalities in video, which allows each of them to attend to the others. The transformer architecture is also leveraged to encode and model the temporal information. On the natural language side, we investigate the best practices to jointly optimize the language embedding together with the multi-modal transformer. This novel framework allows us to establish state-of-the-art results for video retrieval on three datasets. More details are available at http://thoth.inrialpes.fr/research/MMT.

Keywords: video, language, retrieval, multi-modal, cross-modal, temporality, transformer, attention

1 Introduction

Video is one of the most popular forms of media due to its ability to capture dynamic events and its natural appeal to our visual and auditory senses. Online video platforms are playing a major role in promoting this form of media. However, the billions of hours of video available on such platforms are unusable if we cannot access them effectively, for example, by retrieving relevant content through queries.

In this paper, we tackle the tasks of caption-to-video and video-to-caption retrieval. In the first task of caption-to-video retrieval, we are given a query in the form of a caption (e.g., "How to build a house") and the goal is to retrieve the videos best described by it (i.e., videos explaining how to build a house). In practice, given a test set of caption-video pairs, our aim is to provide, for each caption query, a ranking of all the video candidates such that the video associated with the caption query is ranked as high as possible. On the other hand, the task of video-to-caption retrieval focuses on finding among a collection of caption candidates the ones that best describe the query video.

* Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France.

Fig. 1: When matching a text query with videos, the inherent cross-modal and temporal information in videos needs to be leveraged effectively, for example, with a video encoder that handles all the constituent modalities (appearance, audio, speech) jointly across the entire duration of the video. In this example, a video encoder will be able to distinguish between "someone walking *to*" and "someone walking *away*" only if it exploits the temporal information of events occurring in the video (red arrows). Also, in order to understand that a "motorbike failed to start", it needs to use cross-modal information (e.g., absence of noise after someone tried to start the engine, orange arrow).

A common approach for the retrieval problem is similarity learning [29], where we learn a function of two elements (a query and a candidate) that best describes their similarity. All the candidates can then be ranked according to their similarity with the query. In order to perform this ranking, the captions as well as the videos are represented in a common multi-dimensional embedding space, wherein similarities can be computed as a dot product of their corresponding representations. The critical question here is how to learn accurate representations of both caption and video to base our similarity estimation on.

The problem of learning representation of text has been extensively studied, leading to various methods [3,7,18,25,34], which can be used to encode captions. In contrast to these advances, learning effective video representation continues to be a challenge, and forms the focus of our work. This is in part due to the multimodal and temporal nature of video. Video data not only varies in terms of appearance, but also in possible motion, audio, overlaid text, speech, etc. Leveraging cross-modal relations thus forms a key to building effective video representations. As illustrated in Fig. 1, cues jointly extracted from all the constituent modalities are more informative than handling each modality independently. Hearing a motor sound right after seeing someone starting a bike tells us that the running bike is the visible one and not a background one. Another example is the case of a video of "a crowd listening to a talk", neither of the modali-

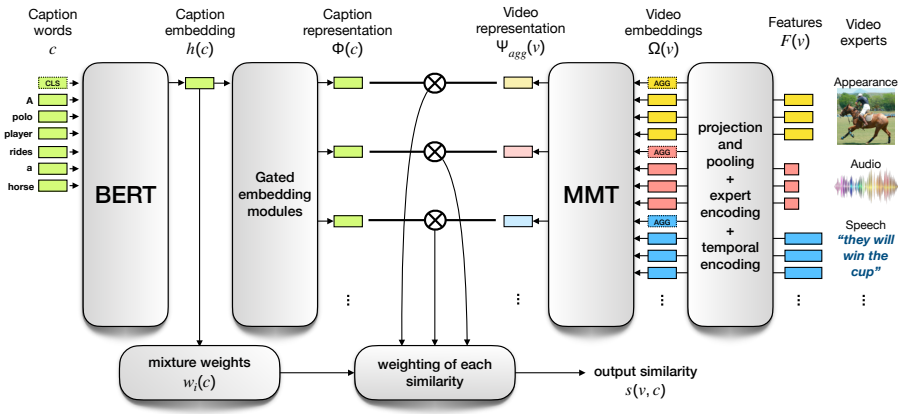

Fig. 2: Our cross-modal framework for similarity estimation. We use our Multi-modal Transformer (MMT, right) to encode video, and BERT (left) for text.

ties "appearance" or "audio" can fully describe the scene, but when processed together, higher level semantics can be obtained.

Recent work on video retrieval does not fully exploit such cross-modal high-level semantics. They either ignore the multi-modal signal [15], treat modalities separately [16], or only use a gating mechanism to modulate certain modality dimensions [14]. Another challenge in representing video is its temporality. Due to the difficulty in handling variable duration of videos, current approaches [14, 16] discard long-term temporal information by aggregating descriptors extracted at different moments in the video. We argue that this temporal information can be important to the task of video retrieval. As shown in Fig. 1, a video of "someone walking *to* an object" and "someone walking *away* from an object" will have the same representation once pooled temporally, however, the movement of the person relative to the object is potentially important in the query.

We address the temporal and multi-modal challenges posed in video data by introducing our multi-modal transformer. It performs the task of processing features extracted from different modalities at different moments in video and aggregates them in a compact representation. Building on the transformer architecture [25], our multi-modal transformer exploits the self-attention mechanism to gather valuable cross-modal and temporal cues about events occurring in a video. We integrate our multi-modal transformer in a cross-modal framework, as illustrated in Fig. 2, which leverages both captions and videos, and estimates their similarity.

Contributions. In this work, we make the following three contributions: (i) First, we introduce a novel video encoder architecture for retrieval: Our multi-modal transformer processes effectively multiple modality features extracted at different times. (ii) We thoroughly investigate different architectures for language embed-

ding, and show the superiority of the BERT model for the task of video retrieval. (iii) By leveraging our novel cross-modal framework we outperform prior state of the art for the task of video retrieval on MSRVTT [30], ActivityNet [12] and LSMDC [21] datasets. It is also the winning solution in the CVPR 2020 Video Pentathlon Challenge [4].

2 Related work

We present previous work on language and video representation learning, as well as on visual-language retrieval.

Language representations. Earlier work on language representations include bag of words [34] and Word2Vec [18]. A limitation of these representations is capturing the sequential properties in a sentence. LSTM [7] was one of the first successful deep learning models to handle this. More recently, the transformer [25] architecture has shown impressive results for text representation by implementing a self-attention mechanism where each word (or wordpiece [27]) of the sentence can attend to all the others. The transformer architecture, consisting of self-attention layers alternatively stacked with fully-connected layers, forms the base of the popular language modeling network BERT [3]. Burns et al. [1] perform an analysis of the different word embeddings and language models (Word2Vec [18], LSTM [7], BERT [3], etc.) used in vision-language tasks. They show that the pretrained and frozen BERT model [3] performs relatively poorly compared to a LSTM or even a simpler average embedding model. In this work, we show that for video retrieval, a pretrained BERT outperforms other language models, but it needs to be finetuned.

Video representations. With a two-stream network, Simonyan et al. [22] have used complementary information from still frames and motion between frames to perform action recognition in videos. Carreira et al. [2] incorporated 3D convolutions in a two-stream network to better attend the temporal structure of the signal. S3D [28] is an alternative approach, which replaced the expensive 3D spatio-temporal convolutions by separable 2D and 1D convolutions. More recently, transformer-based methods, which leverage BERT pretraining [3], have been applied to S3D features in VideoBERT [24] and CBT [23]. While these works focus on visual signals, they have not studied how to encode the other multi-modal semantics, such as audio signals.

Visual-language retrieval. Harwath et al. [5] perform image and audio-caption retrieval by embedding audio segments and image regions in the same space and requiring high similarity between each audio segment and its corresponding image region. The method presented in [13] takes a similar approach for image-text retrieval by embedding images regions and words in a joint space. A high similarity is obtained for images that have matching words and image regions.

For videos, JSFusion [31] estimates video-caption similarity through dense pairwise comparisons between each word of the caption and each frame of the video. In this work, we instead estimate both a video embedding and a caption

embedding and then compute the similarity between them. Zhang et al. [33] perform paragraph-to-video retrieval by assuming a hierarchical decomposition of the video and paragraph. Our method do not assume that the video can be decomposed into clips that align with sentences of the caption. A recent alternative is creating separate embedding spaces for different parts of speech (e.g., noun or verb) [26]. In contrast to this method, we do not pre-process the sentences but encode them directly through BERT.

Another work [17] leverages the large number of instructional videos in the HowTo100M dataset, but does not fully exploit the temporal relations. Our work instead relies on longer segments extracted from HowTo100M videos in order to learn temporal dependencies and address the problem of misalignment between speech and visual features. Mithun et al. [19, 20] use three experts (Object, Activity and Place) to compute three corresponding text-video similarities. These experts however do not collaborate together as their respective similarities are simply summed together. A related approach [16] uses precomputed features from experts for text to video retrieval, where the overall similarity is obtained as a weighted sum of each expert's similarity. A recent extension [14] to this mixture of experts model uses a collaborative gating mechanism for modulating each expert feature according to the other experts. However, this collaborative gating mechanism only strengthens (or weakens) some dimensions of the input signal in a single step, and is therefore not able to capture high level inter-modality information. Our multi-modal transformer overcomes this limitation by attending to all available modalities over multiple self-attention layers.

3 Methodology

Our overall method relies on learning a function s to compute the similarity between two elements: text and video, as shown in Fig. 2. We then rank all the videos (or captions) in the dataset, according to their similarity with the query caption (or video) in the case of text-to-video (or video-to-text) retrieval. In other words, given a dataset of n video-caption pairs $\{(v_1, c_1), \ldots, (v_n, c_n)\}$, the goal of the learnt similarity function $s(v_i, c_j)$, between video v_i and caption c_j, is to provide a high value if $i = j$, and a low one if $i \neq j$. Estimating this similarity (described in Section 3.3) requires accurate representations for the video as well as the caption. Fig. 2 shows the two parts focused on producing these representations (presented in Sections 3.1 and 3.2 respectively) in our cross-modal framework.

3.1 Video representation

The video-level representation is computed by our proposed multi-modal transformer (MMT). MMT follows the architecture of the transformer encoder presented in [25]. It consists of stacked self-attention layers and fully collected layers. MMT's input (v) is a set of embeddings, all of the same dimension d_{model}. Each

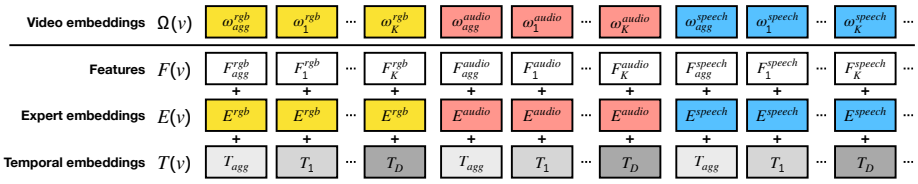

Fig. 3: Inputs to our multi-modal transformer. We combine feature semantics F, expert information E, and temporal cues T to form our video embeddings (v), which are input to MMT.

of them embeds the semantics of a feature, its modality, and the time in the video when the feature was extracted. This input is given by:

$$(v) = F(v) + E(v) + T(v); \tag{1}$$

In the following, we describe those three components.

Features F. In order to learn an effective representation from different modalities inherent in video data, we begin with video feature extractors called "experts" [14,16,19,31]. In contrast to previous methods, we learn a joint representation leveraging both cross-modal and long-term temporal relationships among the experts. We use N pretrained experts $fF^n g_{n=1}^N$. Each expert is a model trained for a particular task that is then used to extract features from video. For a video v, each expert extracts a sequence $F^n(v) = [F_1^n; ...; F_K^n]$ of K features.

The features extracted by our experts encode the semantics of the video. Each expert F^n outputs features in $\mathbb{R}^{d_n}$. In order to project the different expert features into a common dimension d_{model}, we learn N linear layers (one per expert) to project all the features into $\mathbb{R}^{d_{model}}$.

A transformer encoder produces an embedding for each of its feature inputs, resulting in several embeddings for an expert. In order to obtain a unique embedding for each expert, we define an aggregated embedding F_{agg}^n that will collect and contextualize the expert's information. We initialize this embedding with a max pooling aggregation of all the corresponding expert's features as $F_{agg}^n = maxpool(fF_k^n g_{k=1}^K)$. The sequence of input features to our video encoder then takes the form:

$$F(v) = [F_{agg}^1; F_1^1; ...; F_K^1; ...; F_{agg}^N; F_1^N; ...; F_K^N]: \tag{2}$$

Expert embeddings E. In order to process cross-modality information, our MMT needs to identify which expert it is attending to. We learn N embeddings $fE_1; ...; E_N g$ of dimension d_{model} to distinguish between embeddings of different experts. Thus, the sequence of expert embeddings to our video encoder takes the form:

$$E(v) = [E^1; E^1; ...; E^1; ...; E^N; E^N; ...; E^N]: \tag{3}$$

Temporal embeddings T. They provide temporal information about the time in the video where each feature was extracted to our multi-modal transformer. Considering videos of a maximum duration of t_{max} seconds, we learn $D = jt_{max}j$ embeddings $fT_1; :::; T_Dg$ of dimension d_{model}. Each expert feature that has been extracted in the time range $[t; t+1)$ will be temporally embedded with T_{t+1}. For example, a feature extracted at 7.4s in the video will be temporally encoded with temporal embedding T_8. We learn two additional temporal embeddings T_{agg} and T_{unk}, which encode aggregated features and unknown temporal information features (for experts whose temporal information is unknown), respectively. The sequence of temporal embeddings of our video encoder then takes the form:

$$T(v) = [T_{agg}; T_1; :::; T_D; :::; T_{agg}; T_1; :::; T_D]: \tag{4}$$

Multi-modal Transformer. The video embeddings (v) defined as the sum of features, expert and temporal embeddings in (1), as shown in Fig. 3, are input to the transformer. They are given by: $(v) = F(v) + E(v) + T(v) = [!^1_{agg}; !^1_1; :::; !^1_K; :::; !^N_{agg}; !^N_1; :::; !^N_K]$: MMT contextualises its input (v) and produces the video representation $_{agg}(v)$. As illustrated in Fig. 2, we only keep the aggregated embedding per expert. Thus, our video representation $_{agg}(v)$ consists of the output embeddings corresponding to the aggregated features, i.e.,

$$_{agg}(v) = MMT(\ (v)) = [\ ^1_{agg}; :::; \ ^N_{agg}]: \tag{5}$$

The advantage of our MMT over the state-of-the-art collaborative gating mechanism [14] is two-fold: First, the input embeddings are not simply modulated in a single step but iteratively refined through several layers featuring multiple attention heads. Second, we do not limit our video encoder with a temporally aggregated feature for each expert, but provide all the extracted features instead, along with a temporal encoding describing at what moment of the video they were extracted from. Thanks to its self-attention modules, each layer of our multi-modal transformer is able to attend to all its input embeddings, thus extracting semantics of events occurring in the video over several modalities.

3.2 Caption representation

We compute our caption representation (c) in two stages: first, we obtain an embedding $h(c)$ of the caption, and then project it with a function g into N different spaces as $= g \ h$. For the embedding function h, we use a pre-trained BERT model [3]. Specifically, we extract our single caption embedding $h(c)$ from the [CLS] output of BERT. In order to match the size of this caption representation with that of video, we learn for function g as many gated embedding modules [16] as there are video experts. Our caption representation then consists of N embeddings, represented by $(c) = f \ ^i g^N_{i=1}$.

3.3 Similarity estimation

We compute our final video-caption similarity s, as a weighted sum of each expert i's video-caption similarity $h^{\ i};\ ^{i}_{agg}i$. It is given by:

$$s(v;c) = \sum_{i=1}^{N} w_i(c)h^{\ i};\ ^{i}_{agg}i; \tag{6}$$

where $w_i(c)$ represents the weight for the ith expert. To obtain these mixture weights, we follow [16] and process our caption representation $h(c)$ through a linear layer and then perform a softmax operation, i.e.,

$$w_i(c) = \frac{e^{h(c)^{>}a_i}}{\sum_{j=1}^{N} e^{h(c)^{>}a_j}}; \tag{7}$$

where $(a_1;:::;a_N)$ are the weights of the linear layer. The intuition behind using a weighted sum is that a caption may not describe all the inherent modalities in video uniformly. For example, in the case of a video with a person in a red dress singing opera, the caption "a person in a red dress" provides no information relevant for audio. On the contrary, the caption "someone is singing" should focus on the audio modality for computing similarity. Note that $w_i(c);\ ^{i}$ and $^{i}_{agg}$ can all be precomputed offline for each caption and for each video, and therefore the retrieval operation only involves dot product operations.

3.4 Training

We train our model with the bi-directional max-margin ranking loss [10]:

$$L = \frac{1}{B}\sum_{i=1}^{B}\sum_{j\neq i}\Big[\max(0;s_{ij} \quad s_{ii}+m)+\max(0;s_{ji} \quad s_{ii}+m)\Big]; \tag{8}$$

where B is the batch size, $s_{ij} = s(v_i;c_j)$, the similarity score between video v_i and caption c_j, and m is the margin. This loss enforces the similarity for true video-caption pairs s_{ii} to be higher than the similarity of negative samples s_{ij} or s_{ji}, for all $i \neq j$, by at least m.

4 Experiments

4.1 Datasets and Metrics

HowTo100M [17]. It is composed of more than 1 million YouTube instructional videos, along with automatically-extracted speech transcriptions, which form the captions. These captions are naturally noisy, and often do not describe the visual content accurately or are temporally misaligned with it. We use this dataset only for pre-training.

MSRVTT [30]. This dataset is composed of 10K YouTube videos, collected using 257 queries from a commercial video search engine. Each video is 10 to 30s long, and is paired with 20 natural sentences describing it, obtained from Amazon Mechanical Turk workers. We use this dataset for training from scratch and also for fine-tuning. We report results on the train/test splits introduced in [31] that uses 9000 videos for training and 1000 for test. We refer to this split as "1k-A". We also report results on the train/test split in [16] that we refer to as "1k-B". Unless otherwise specified, our MSRVTT results are with "1k-A".
ActivityNet Captions [12]. It consists of 20K YouTube videos temporally annotated with sentence descriptions. We follow the approach of [33], where all the descriptions of a video are concatenated to form a paragraph. The training set has 10009 videos. We evaluate our video-paragraph retrieval on the "val1" split (4917 videos). We use ActivityNet for training from scratch and fine-tuning.
LSMDC [21]. It contains 118,081 short video clips (4–5s) extracted from 202 movies. Each clip is annotated with a caption, extracted from either the movie script or the audio description. The test set is composed of 1000 videos, from movies not present in the training set. We use LSMDC for training from scratch and also fine-tuning.
Metrics. We evaluate the performance of our model with standard retrieval metrics: recall at rank N (R@N, higher is better), median rank (MdR, lower is better) and mean rank (MnR, lower is better). For each metric, we report the mean and the standard deviation over experiments with 3 random seeds. In the main paper, we only report recall@5, median and mean ranks, and refer the reader to the supplementary material for additional metrics.

4.2 Implementation details

Pre-trained experts. Recall that our video encoder uses pre-trained experts models for extracting features from each video modality. We use the following seven experts. **Motion** features are extracted from S3D [28] trained on the Kinetics action recognition dataset. **Audio** features are extracted using VGGish model [6] trained on YT8M. **Scene** embeddings are extracted from DenseNet-161 [9] trained for image classification on the Places365 dataset [35]. **OCR** features are obtained in three stages. Overlaid text is first detected using the pixel link text detection model. The detected boxes are then passed through a text recognition model trained on the Synth90K dataset. Finally, each character sequence is encoded with word2vec [18] embeddings. **Face** features are extracted in two stages. An SSD face detector is used to extract bounding boxes, which are then passed through a ResNet50 trained for face classification on the VGGFace2 dataset. **Speech** transcripts are extracted using the Google Cloud Speech to Text API, with the language set to English. The detected words are then encoded with word2vec. **Appearance** features are extracted from the final global average pooling layer of SENet-154 [8] trained for classification on ImageNet. For scene, OCR, face, speech and appearance, we use the features publicly released by [14], and compute the other features ourselves.

Training. For each dataset, we run a grid search on the corresponding validation set to estimate the hyperparameters. We use the Adam optimizer for all our experiments, and set the margin of the bidirectional max-margin ranking loss to 0.05. We also freeze our pre-trained expert models.

When pre-training on HowTo100M, we use a batch size of 64 video-caption pairs, an initial learning rate of 5e-5, which we decay by a multiplicative factor 0.98 every 10K optimisation steps, and train for 2 million steps. Given the long duration of most of the HowTo100M videos, we randomly sample 100 consecutive words in the caption, and keep 100 consecutive seconds of video data, closest in time to the selected words.

When training from scratch or finetuning on MSRVTT or LSMDC, we use a batch size of 32 video-caption pairs, an initial learning rate of 5e-5, which we decay by a multiplicative factor 0.95 every 1K optimisation steps. We train for 50K steps. We use the same settings when training from scratch or finetuning on ActivityNet, except for 0.90 as the multiplicative factor.

To compute our caption representation $h(c)$, we use the "BERT-base-cased" checkpoint of the BERT model and finetune it with a dropout probability of 10%. To compute our video representation $_{agg}(v)$, we use MMT with 4 layers and 4 attention heads, a dropout probability of 10%, a hidden size d_{model} of 512, and an intermediate size of 3072.

For datasets with short videos (MSRVTT and LSMDC), we use all the 7 experts and limit video input to 30 features per expert, and BERT input to the first 30 wordpieces. For datasets containing longer videos (HowTo100M and ActivityNet), we only use motion and audio experts, and limit our video input to 100 features per expert and our BERT input to the first 100 wordpieces. In cases where an expert is unavailable for a given video, e.g., no speech was detected, we set the aggregated feature F^n_{agg} to a zero vector. We refer the reader to the supplementary material for a study of the model complexity.

4.3 Ablation studies and comparisons

We will first show the advantage of pretraining our model on a large-scale, uncurated dataset. We will then perform ablations on the architecture used for our language and video encoders. Finally, we will present the relative importance of the pretrained experts used in this work, and compare with related methods.

Pretraining. Table 1 shows the advantage of pretraining on HowTo100M, before finetuning on the target dataset (MSRVTT in this case). We also evaluated the impact of pretraining on ActivityNet and LSMDC; see Table 5 and Table 6.

Language encoder. We evaluated several architectures for caption representation, as shown in Table 2. Similar to the observation made in [1], we obtain poor results from a frozen, pretrained BERT. Using the [CLS] output from a pretrained and frozen BERT model is in fact the worst result. We suppose this is because the output was not trained for caption representation, but for a very different task: next sentence prediction. Finetuning BERT greatly improves performance; it is the best result. We also compare with GrOVLE [1] embeddings, frozen or finetuned, aggregated with a max-pooling operation or a 1-layer LSTM

Table 1: Advantage of pretraining on HowTo100M then finetuning on MSRVTT. Impact of removing the stop words. Performance reported on MSRVTT.

		Text ! Video		
Method	Caption	R@5″	MdR#	MnR#
Pretraining without finetuning (zero-shot setting)	all words	6.9	160.0	240.2
	w/o stop words	**14.4**	**66.0**	**148.1**
Training from scratch on MSRVTT	all words	**54.0** $_{0:2}$	**4.0** $_{0:0}$	**26.7** $_{0:9}$
	w/o stop words	50.0 $_{0:6}$	5.3 $_{0:5}$	28.5 $_{0:9}$
Pretraining then finetuning on MSRVTT	all words	**57.1** $_{1:0}$	**4.0** $_{0:0}$	**24.0** $_{0:8}$
	w/o stop words	55.0 $_{0:7}$	4.3 $_{0:5}$	24.3 $_{0:3}$

Table 2: Comparison of different architectures for caption embedding when training from scratch on MSRVTT.

			Text ! Video		
Word embeddings		Aggregation	R@5″	MdR#	MnR#
GrOVLE	frozen	maxpool	31.8 $_{0:4}$	14.7 $_{0:5}$	63.1 $_{1:3}$
		LSTM	36.4 $_{0:8}$	10.3 $_{0:9}$	44.2 $_{0:1}$
	finetuned	maxpool	34.6 $_{0:1}$	12.0 $_{0:0}$	52.3 $_{0:8}$
		LSTM	40.3 $_{0:5}$	8.7 $_{0:5}$	38.1 $_{0:7}$
BERT	frozen	maxpool	39.4 $_{0:8}$	9.7 $_{0:5}$	46.5 $_{0:2}$
		LSTM	36.4 $_{1:8}$	10.7 $_{0:5}$	42.2 $_{0:6}$
	finetuned	maxpool	44.2 $_{1:2}$	7.3 $_{0:5}$	35.6 $_{0:4}$
		LSTM	40.1 $_{1:0}$	8.7 $_{0:5}$	37.4 $_{0:5}$
	frozen	BERT-frozen	17.1 $_{0:2}$	34.7 $_{1:2}$	98.8 $_{0:8}$
	finetuned	BERT-finetuned	**54.0** $_{0:2}$	**4.0** $_{0:0}$	**26.7** $_{0:9}$

and a fully-connected layer. We show that pretrained BERT embeddings aggregated by a max-pooling operation perform better than GrOVLE embeddings processed by a LSTM (best results from [1] for the text-to-clip task).

We also analysed the impact of removing stop words from the captions in Table 1. In a zero-shot setting, i.e., trained on HowTo100M, evaluated on MSRVTT without finetuning, removing the stop words helps generalize, by bridging the domain gap—HowTo100M speech is very different from MSRVTT captions. This approach was adopted in [15]. However, we observe that when finetuning, it is better to keep all the words as they contribute to the semantics of the caption.

Video encoder. We evaluated the influence of different architectures for computing video embeddings on the MSRVTT 1k-A test split.

In Table 3a, we evaluate variants of our encoder architecture and its input. Similar to [16], we experiment with directly computing the caption-video similarities on each max-pooled expert features, i.e., no video encoder (NONE in the table). We compare this with the collaborative gating architecture (COLL) [14] and our MMT variant using only the aggregated features as input. For the first two variants without MMT, we adopt the approach of [16] to deal with missing modalities by re-weighting $w_i(c)$. We also show the superior performance of our multi-modal transformer in contextualising the different modality embeddings compared to the collaborative gating approach. We argue that our MMT is able to extract cross-modal information in a multi-stage architecture compared to collaborative gating, which is limited to modulating the input embeddings. Ta-

Table 3: Ablation studies on the video encoder of our framework with MSRVTT. **(a) Influence of the architecture and input.** With max-pooled features as input, we compare our transformer architecture (MMT) with the variant not using an encoder (NONE) and the one with Collaborative Gating [14] (COLL). We also show that MMT can attend to all extracted features, as detailed in the text. **(b) Importance of initializing F^n_{agg} features.** We compare zero-vector initialisation, mean pooling and max pooling of the expert features. **(c) Influence of the size of the multi-modal transformer.** We compare different values for number-of-layers number-of-attention-heads.

(a) Encoder architecture and input

		Text $\longrightarrow$ *Video*		
Encoder	Input	R@5↑	MdR↓	MnR↓
NONE	max pool	$50.9_{\pm 1.5}$	$5.3_{\pm 0.5}$	$28.6_{\pm 0.5}$
COLL	max pool	$51.3_{\pm 0.8}$	$5.0_{\pm 0.0}$	$29.5_{\pm 1.8}$
MMT	max pool	$52.5_{\pm 0.7}$	$5.0_{\pm 0.0}$	$27.2_{\pm 0.7}$
MMT	shuffled feats	$53.3_{\pm 0.2}$	$5.0_{\pm 0.0}$	$27.4_{\pm 0.7}$
MMT	ordered feats	$\mathbf{54.0}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$

(b) F^n_{agg} initialisation

	Text $\longrightarrow$ *Video*		
F^n_{agg} init	R@5↑	MdR↓	MnR↓
zero	$50.2_{\pm 0.9}$	$5.7_{\pm 0.5}$	$28.5_{\pm 1.3}$
mean pool	$\mathbf{54.2}_{\pm 0.3}$	$5.0_{\pm 0.0}$	$27.1_{\pm 0.9}$
max pool	$54.0_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$

(c) Model size

		Text $\longrightarrow$ *Video*		
Layers	Heads	R@5↑	MdR↓	MnR↓
2	2	$53.2_{\pm 0.4}$	$5.0_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.4}$
4	4	$\mathbf{54.0}_{\pm 0.2}$	$\mathbf{4.0}_{\pm 0.0}$	$\mathbf{26.7}_{\pm 0.9}$
8	8	$53.9_{\pm 0.3}$	$4.7_{\pm 0.5}$	$\mathbf{26.7}_{\pm 0.7}$

ble 3a also highlights the advantage of providing MMT with **all** the extracted features, instead of only aggregated ones. Temporally aggregating each expert's features ignores information about multiple events occurring in a same video (see the last three rows). As shown by the influence of ordered and randomly shuffled features on the performance, MMT has the capacity to make sense of the relative ordering of events in a video.

Table 3b shows the importance of initialising the expert aggregation feature F^n_{agg}. Since the output of our video encoder is extracted from the "agg" columns, it is important to initialise them with an appropriate representation of the experts' features. The transformer being a residual network architecture, initializing F^n_{agg} input embeddings with a zero vector leads to a low performance. Initializing with max pooling aggregation of each expert performs better than mean pooling. Finally, we analyze the impact of the size of our multi-modal transformer model in Table 3c. A model with 4 layers and 4 attention heads outperforms both a smaller model (2 layers and 2 attention heads) and a larger model (8 layers and 8 attention heads).

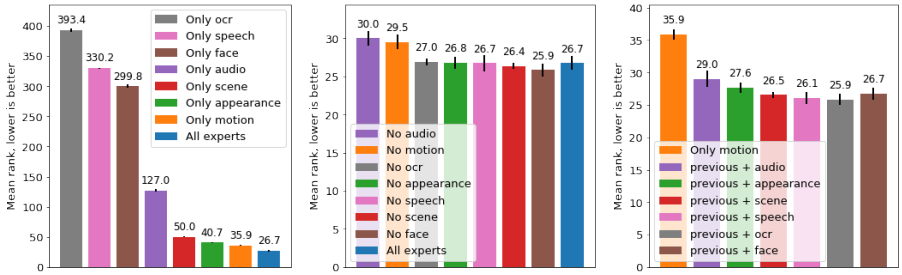

Fig. 4: MSRVTT performance (mean rank; lower is better) after training from scratch, when using only one expert (left), when using all experts but one (middle), when gradually adding experts by greedy search (right).

Comparison of the di erent experts. In Figure 4, we show an ablation study when training our model on MSRVTT using only one expert (left), using all experts but one (middle), or gradually adding experts by greedy search (right). In the case of using only one expert, we note that the motion expert provides the best results. We attribute the poor performance of OCR, speech and face to the fact that they are absent from many videos, thus resulting in a zero vector input to our video encoder. While the scene expert shows a decent performance, if used alone, it does not contribute when used along other experts, perhaps due to the semantics it encodes being captured already by other experts like appearance or motion. On the contrary, the audio expert alone does not provide a good performance, but it contributes the most when used in conjunction with the others, most likely due to the complementary cues it provides, compared to the other experts.

Comparison to prior state of the art. We compare our method on three datasets: MSRVTT (Table 4), ActivityNet (Table 5) and LSMDC (Table 6). While MSRVTT and LSMDC contain short video-caption pairs (average video duration of 13s for MSRVTT, one-sentence captions), ActivityNet contains much longer videos (several minutes) and each video is captioned with multiple sentences. We consider the concatenation of all these sentences as the caption. We show that our method obtains state-of-the-art results on all the three datasets. The gains obtained through MMT's long term temporal encoding are particularly noticeable on the long videos of ActivityNet.

5 Summary

We introduced multi-modal transformer, a transformer-based architecture capable of attending multiple features extracted at different moments, and from different modalities in video. This leverages both temporal and cross-modal cues, which are crucial for accurate video representation. We incorporate this video encoder along with a caption encoder in a cross-modal framework to perform

Table 4: Retrieval performance on the MSRVTT dataset. 1k-A and 1k-B denote test sets of 1000 randomly sampled caption-video pairs used in [31] and [16] resp.

		Text ! Video			*Video ! Text*		
Method	Split	R@5"	MdR#	MnR#	R@5"	MdR#	MnR#
Random baseline	1k-A	0.5	500.0	500.0	0.5	500.0	500.0
JSFusion [31]	1k-A	31.2	13	-	-	-	-
HT [17]	1k-A	35.0	12	-	-	-	-
CE [14]	1k-A	48.8 $_{0:6}$	6.0 $_{0:0}$	28.2 $_{0:8}$	50.3 $_{0:5}$	5.3 $_{0:6}$	25.1 $_{0:8}$
Ours	1k-A	**54.0** $_{0:2}$	**4.0** $_{0:0}$	**26.7** $_{0:9}$	**56.0** $_{0:9}$	**4.0** $_{0:0}$	**23.6** $_{1:0}$
HT-pretrained [17]	1k-A	40.2	9	-	-	-	-
Ours-pretrained	1k-A	**57.1** $_{1:0}$	**4.0** $_{0:0}$	**24.0** $_{0:8}$	**57.5** $_{0:6}$	**3.7** $_{0:5}$	**21.3** $_{0:6}$
Random baseline	1k-B	0.5	500.0	500.0	0.5	500.0	500.0
MEE [16]	1k-B	37.9	10.0	-	-	-	-
JPose [26]	1k-B	38.1	9	-	41.3	8.7	-
MEE-COCO [16]	1k-B	39.2	9.0	-	-	-	-
CE [14]	1k-B	46.0 $_{0:4}$	7.0 $_{0:0}$	35.3 $_{1:1}$	46.0 $_{0:5}$	6.5 $_{0:5}$	30.6 $_{1:2}$
Ours	1k-B	**49.1** $_{0:4}$	**6.0** $_{0:0}$	**29.5** $_{1:6}$	**49.4** $_{0:4}$	**6.0** $_{0:0}$	**24.5** $_{1:8}$

Table 5: Retrieval performance on the ActivityNet dataset.

	Text ! Video			*Video ! Text*		
Method	R@5"	MdR#	MnR#	R@5"	MdR#	MnR#
Random baseline	0.1	2458.5	2458.5	0.1	2458.5	2458.5
FSE [33]	44.8 $_{0:4}$	7	-	43.1 $_{1:1}$	7	-
CE [14]	47.7 $_{0:6}$	6.0 $_{0:0}$	23.1 $_{0:5}$	46.6 $_{0:7}$	6.0 $_{0:0}$	24.4 $_{0:5}$
HSE [33]	49.3	-	-	48.1	-	-
Ours	**54.2** $_{1:0}$	**5.0** $_{0:0}$	**20.8** $_{0:4}$	**54.8** $_{0:4}$	**4.3** $_{0:5}$	**21.2** $_{0:5}$
Ours-pretrained	**61.4** $_{0:2}$	**3.3** $_{0:5}$	**16.0** $_{0:4}$	**61.1** $_{0:2}$	**4.0** $_{0:0}$	**17.1** $_{0:5}$

Table 6: Retrieval performance on the LSMDC dataset.

	Text ! Video			*Video ! Text*		
Method	R@5"	MdR#	MnR#	R@5"	MdR#	MnR#
Random baseline	0.5	500.0	500.0	0.5	500.0	500.0
CT-SAN [32]	16.3	46	-	-	-	-
JSFusion [31]	21.2	36	-	-	-	-
CCA [11] (rep. by [16])	21.7	33	-	-	-	-
MEE [16]	25.1	27	-	-	-	-
MEE-COCO [16]	25.6	27	-	-	-	-
CE [14]	26.9 $_{1:1}$	25.3 $_{3:1}$	-	-	-	-
Ours	**29.2** $_{0:8}$	**21.0** $_{1:4}$	**76.3** $_{1:9}$	**29.3** $_{1:1}$	**22.5** $_{0:4}$	**77.1** $_{2:6}$
Ours-pretrained	**29.9** $_{0:7}$	**19.3** $_{0:2}$	**75.0** $_{1:2}$	**28.6** $_{0:3}$	**20.0** $_{0:0}$	**76.0** $_{0:8}$

caption-video matching and obtain state-of-the-art results for video retrieval. As future work, we would like to improve temporal encoding for video and text.

Acknowledgments. We thank the authors of [14] for sharing their codebase and features, and Samuel Albanie, in particular, for his help with implementation details. This work was supported in part by the ANR project AVENUE.

References

1. Burns, A., Tan, R., Saenko, K., Sclaroff, S., Plummer, B.A.: Language Features Matter: Effective language representations for vision-language tasks. In: ICCV (2019)
2. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: CVPR (2017)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
4. Gabeur, V., Sun, C., Alahari, K., Schmid, C.: CVPR 2020 video pentathlon challenge: Multi-modal transformer for video retrieval. In: CVPR Video Pentathlon Workshop (2020)
5. Harwath, D., Recasens, A., Surís, D., Chuang, G., Torralba, A., Glass, J.: Jointly discovering visual objects and spoken words from raw sensory input. In: ECCV (2018)
6. Hershey, S., Chaudhuri, S., Ellis, D.P.W., Gemmeke, J.F., Jansen, A., Moore, C., Plakal, M., Platt, D., Saurous, R.A., Seybold, B., Slaney, M., Weiss, R., Wilson, K.: CNN architectures for large-scale audio classification. In: ICASSP (2017)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8) (Nov 1997)
8. Hu, J., Shen, L., Albanie, S., Sun, G., Wu, E.: Squeeze-and-excitation networks. IEEE Trans. Pattern Analysis and Machine Intelligence (2019)
9. Huang, G., Liu, Z., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2016)
10. Karpathy, A., Joulin, A., Fei-Fei, L.: Deep fragment embeddings for bidirectional image sentence mapping. In: NIPS (2014)
11. Klein, B., Lev, G., Sadeh, G., Wolf, L.: Associating neural word embeddings with deep image representations using fisher vectors. In: CVPR (2015)
12. Krishna, R., Hata, K., Ren, F., Fei-Fei, L., Niebles, J.C.: Dense-captioning events in videos. In: ICCV (2017)
13. Lee, K.H., Chen, X., Hua, G., Hu, H., He, X.: Stacked cross attention for image-text matching. ECCV (2018)
14. Liu, Y., Albanie, S., Nagrani, A., Zisserman, A.: Use what you have: Video retrieval using representations from collaborative experts. ArXiv **abs/1907.13487** (2019)
15. Miech, A., Alayrac, J.B., Smaira, L., Laptev, I., Sivic, J., Zisserman, A.: End-to-End Learning of Visual Representations from Uncurated Instructional Videos. arXiv e-prints arXiv:1912.06430 (Dec 2019)
16. Miech, A., Laptev, I., Sivic, J.: Learning a text-video embedding from incomplete and heterogeneous data. ArXiv **abs/1804.02516** (2018)
17. Miech, A., Zhukov, D., Alayrac, J.B., Tapaswi, M., Laptev, I., Sivic, J.: HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In: ICCV (2019)
18. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR (2013)
19. Mithun, N.C., Li, J., Metze, F., Roy-Chowdhury, A.K.: Learning joint embedding with multimodal cues for cross-modal video-text retrieval. In: ICMR (2018)
20. Mithun, N.C., Li, J., Metze, F., Roy-Chowdhury, A.K.: Joint embeddings with multimodal cues for video-text retrieval. IJMIR (2019)
21. Rohrbach, A., Rohrbach, M., Tandon, N., Schiele, B.: A dataset for movie description. In: CVPR (2015)

22. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NIPS (2014)
23. Sun, C., Baradel, F., Murphy, K., Schmid, C.: Learning video representations using contrastive bidirectional transformer. arXiv 1906.05743 (2019)
24. Sun, C., Myers, A., Vondrick, C., Murphy, K., Schmid, C.: Videobert: A joint model for video and language representation learning. In: ICCV (2019)
25. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
26. Wray, M., Larlus, D., Csurka, G., Damen, D.: Fine-grained action retrieval through multiple parts-of-speech embeddings. In: ICCV (2019)
27. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G.S., Hughes, M., Dean, J.: Google's neural machine translation system: Bridging the gap between human and machine translation. ArXiv 1609.08144 (2016)
28. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: ECCV (2017)
29. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning, with application to clustering with side-information. In: NIPS (2002)
30. Xu, J., Mei, T., Yao, T., Rui, Y.: MSR-VTT: A large video description dataset for bridging video and language. In: CVPR (2016)
31. Yu, Y., Kim, J., Kim, G.: A joint sequence fusion model for video question answering and retrieval. In: ECCV (2018)
32. Yu, Y., Ko, H., Choi, J., Kim, G.: End-to-end concept word detection for video captioning, retrieval, and question answering. CVPR (2017)
33. Zhang, B., Hu, H., Sha, F.: Cross-modal and hierarchical modeling of video and text. In: ECCV (2018)
34. Zhang, Y., Jin, R., Zhou, Z.H.: Understanding bag-of-words model: a statistical framework. International Journal of Machine Learning and Cybernetics **1**, 43–52 (2010)
35. Zhou, B., Lapedriza, À., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE Trans. Pattern Analysis and Machine Intelligence **40**, 1452–1464 (2018)

A Supplementary material

A.1 Model complexity

Number of parameters. As shown below, using multiple modalities does not impact the number of parameters significantly. Interestingly, majority of the parameters correspond to the BERT caption encoding module. We also note that the difference in the video encoder comes from the projections. The number of parameters of a transformer encoder is independent of the number of input embeddings, as are the parameters of a CNN from the image size.

Our cross-modal architecture using 7 modalities has: 133.3M parameters, including caption encoder: 112.9M, video encoder: 20.4M (Projections: 3.3M, MMT: 17.1M). Our cross-modal architecture using 2 modalities has: 127.3M parameters, including caption encoder: 109.6M (decrease compared to 7 modalities due to using less gated embedding modules), video encoder: 17.7M (Projections: 0.6M, MMT: 17.1M).

Training and inference times. Training our full cross-modal architecture from scratch on MSRVTT takes about 4 hours on a single V100 16GB GPU.

If we replace our multi-modal transformer by collaborative gating [14], we reduce the number of parameters from 133.3M to 123.9M. However, the gain in inference time is minimal, from 1.1s to 0.8s, and is negligible compared to feature extraction, as detailed below.

Inference time for 1k videos and 1k text queries from MSRVTT on a single V100 GPU is as follows: approximately 3000s to extract features of 7 experts on 1k videos (480s just for S3D motion features), 1.1s to process videos with MMT, 0.9s to process 1k captions with BERT+gated embedding modules, 0.05s to compute similarities and rank the video candidates for the 1k queries.

A.2 Results on additional metrics

Here, we report our results for the additional metrics R@1, R@10, R@50. Table 7 complements the results reported for the MSRVTT [30] dataset in Table 4 of the main paper. Similarly, Table 8 and Table 9 report the additional evaluations for Table 5 and Table 6 of the main paper on ActivityNet [12] and LSMDC [21] datasets respectively. We observe that the results on these additional metrics are in line with the conclusions of the main paper.

Table 7: Retrieval performance on the MSRVTT dataset. 1k-A and 1k-B denote test sets of 1000 randomly sampled caption-video pairs used in [31] and [16] resp.

		Text ! Video					*Video ! Text*				
Method	Split	R@1"	R@5"	R@10"	MdR#	MnR#	R@1"	R@5"	R@10"	MdR#	MnR#
Random baseline	1k-A	0.1	0.5	1.0	500.0	500.0	0.1	0.5	1.0	500.0	500.0
JSFusion [31]	1k-A	10.2	31.2	43.2	13	-	-	-	-	-	-
HT [17]	1k-A	12.1	35.0	48.0	12	-	-	-	-	-	-
CE [14]	1k-A	20.9 $_{1:2}$	48.8 $_{0:6}$	62.4 $_{0:8}$	6.0 $_{0:0}$	28.2 $_{0:8}$	20.6 $_{0:6}$	50.3 $_{0:5}$	64.0 $_{0:2}$	5.3 $_{0:6}$	25.1 $_{0:8}$
Ours	1k-A	**24.6** $_{0:4}$	**54.0** $_{0:2}$	**67.1** $_{0:5}$	**4.0** $_{0:0}$	**26.7** $_{0:9}$	**24.4** $_{0:5}$	**56.0** $_{0:9}$	**67.8** $_{0:3}$	**4.0** $_{0:0}$	**23.6** $_{1:0}$
HT-pretrained [17]	1k-A	14.9	40.2	52.8	9	-	-	-	-	-	-
Ours-pretrained	1k-A	**26.6** $_{1:0}$	**57.1** $_{1:0}$	**69.6** $_{0:2}$	**4.0** $_{0:0}$	**24.0** $_{0:8}$	**27.0** $_{0:6}$	**57.5** $_{0:6}$	**69.7** $_{0:8}$	**3.7** $_{0:5}$	**21.3** $_{0:6}$
Random baseline	1k-B	0.1	0.5	1.0	500.0	500.0	0.1	0.5	1.0	500.0	500.0
MEE [16]	1k-B	13.6	37.9	51.0	10.0	-	-	-	-	-	-
JPose [26]	1k-B	14.3	38.1	53.0	9	-	16.4	41.3	54.4	8.7	-
MEE-COCO [16]	1k-B	14.2	39.2	53.8	9.0	-	-	-	-	-	-
CE [14]	1k-B	18.2 $_{0:7}$	46.0 $_{0:4}$	60.7 $_{0:2}$	7.0 $_{0:0}$	35.3 $_{1:1}$	18.0 $_{0:8}$	46.0 $_{0:5}$	60.3 $_{0:5}$	6.5 $_{0:5}$	30.6 $_{1:2}$
Ours	1k-B	**20.3** $_{0:5}$	**49.1** $_{0:4}$	**63.9** $_{0:5}$	**6.0** $_{0:0}$	**29.5** $_{1:6}$	**21.1** $_{0:4}$	**49.4** $_{0:4}$	**63.2** $_{0:4}$	**6.0** $_{0:0}$	**24.5** $_{1:8}$

Table 8: Retrieval performance on the ActivityNet dataset.

	Text ! Video					*Video ! Text*				
Method	R@1"	R@5"	R@50"	MdR#	MnR#	R@1"	R@5"	R@50"	MdR#	MnR#
Random baseline	0.02	0.1	1.02	2458.5	2458.5	0.02	0.1	1.02	2458.5	2458.5
FSE [33]	18.2 $_{0:2}$	44.8 $_{0:4}$	89.1 $_{0:3}$	7	-	16.7 $_{0:8}$	43.1 $_{1:1}$	88.4 $_{0:3}$	7	-
CE [14]	18.2 $_{0:3}$	47.7 $_{0:6}$	91.4 $_{0:4}$	6.0 $_{0:0}$	23.1 $_{0:5}$	17.7 $_{0:6}$	46.6 $_{0:7}$	90.9 $_{0:2}$	6.0 $_{0:0}$	24.4 $_{0:5}$
HSE [33]	20.5	49.3	-	-	-	18.7	48.1	-	-	-
Ours	**22.7** $_{0:2}$	**54.2** $_{1:0}$	**93.2** $_{0:4}$	**5.0** $_{0:0}$	**20.8** $_{0:4}$	**22.9** $_{0:8}$	**54.8** $_{0:4}$	**93.1** $_{0:2}$	**4.3** $_{0:5}$	**21.2** $_{0:5}$
Ours-pretrained	**28.7** $_{0:2}$	**61.4** $_{0:2}$	**94.5** $_{0:0}$	**3.3** $_{0:5}$	**16.0** $_{0:4}$	**28.9** $_{0:2}$	**61.1** $_{0:2}$	**94.3** $_{0:4}$	**4.0** $_{0:0}$	**17.1** $_{0:5}$

Table 9: Retrieval performance on the LSMDC dataset.

	Text ! Video					*Video ! Text*				
Method	R@1"	R@5"	R@10"	MdR#	MnR#	R@1"	R@5"	R@10"	MdR#	MnR#
Random baseline	0.1	0.5	1.0	500.0	500.0	0.1	0.5	1.0	500.0	500.0
CT-SAN [32]	5.1	16.3	25.2	46	-	-	-	-	-	-
JSFusion [31]	9.1	21.2	34.1	36	-	-	-	-	-	-
CCA [11] (rep. by [16])	7.5	21.7	31.0	33	-	-	-	-	-	-
MEE [16]	9.3	25.1	33.4	27	-	-	-	-	-	-
MEE-COCO [16]	10.1	25.6	34.6	27	-	-	-	-	-	-
CE [14]	11.2 $_{0:4}$	26.9 $_{1:1}$	34.8 $_{2:0}$	25.3 $_{3:1}$	-	-	-	-	-	-
Ours	**13.2** $_{0:4}$	**29.2** $_{0:8}$	**38.8** $_{0:9}$	**21.0** $_{1:4}$	**76.3** $_{1:9}$	**12.1** $_{0:1}$	**29.3** $_{1:1}$	**37.9** $_{1:1}$	**22.5** $_{0:4}$	**77.1** $_{2:6}$
Ours-pretrained	**12.9** $_{0:1}$	**29.9** $_{0:7}$	**40.1** $_{0:8}$	**19.3** $_{0:2}$	**75.0** $_{1:2}$	**12.3** $_{0:2}$	**28.6** $_{0:3}$	**38.9** $_{0:8}$	**20.0** $_{0:0}$	**76.0** $_{0:8}$