Lightweight Attentional Feature Fusion: A New Baseline for Text-to-Video Retrieval

Fan Hu[1,2*] , Aozhu Chen[1,2*], Ziyue Wang[1,2*], Fangming Zhou[1,2], Jianfeng Dong[3], and Xirong Li[1,2†]

[1] MoE Key Lab of DEKE, Renmin University of China
[2] AIMC Lab, School of Information, Renmin University of China
[3] College of Computer and Information Engineering, Zhejiang Gongshang University

Abstract. In this paper we revisit *feature fusion*, an old-fashioned topic, in the new context of text-to-video retrieval. Different from previous research that considers feature fusion only at one end, let it be video or text, we aim for feature fusion for both ends within a unified framework. We hypothesize that optimizing the convex combination of the features is preferred to modeling their correlations by computationally heavy multi-head self attention. We propose Lightweight Attentional Feature Fusion (LAFF). LAFF performs feature fusion at both early and late stages and at both video and text ends, making it a powerful method for exploiting diverse (off-the-shelf) features. The interpretability of LAFF can be used for feature selection. Extensive experiments on five public benchmark sets (MSR-VTT, MSVD, TGIF, VATEX and TRECVID AVS 2016-2020) justify LAFF as a new baseline for text-to-video retrieval.

Keywords: Text-to-video retrieval, video/text feature fusion

1 Introduction

Text-to-video retrieval is to retrieve videos *w.r.t.* to an ad-hoc textual query from many *unlabeled* videos. Both video and text have to be embedded into one or more cross-modal common spaces for text-to-video matching. The state-of-the-art tackles the task in different approaches, including novel networks for query representation learning [59, 65], multi-modal Transformers for video representation learning [3, 19], hybrid space learning for interpretable cross-modal matching [15, 60], and more recently CLIP2Video [17] that learns text and video representations in an end-to-end manner. Differently, we look into *feature fusion*, an important yet largely underexplored topic for text-to-video retrieval.

Given video/text samples represented by diverse features, feature fusion aims to answer a basic research question of *what is the optimal way to combine these features?* By optimal we mean the fusion shall maximize the retrieval performance. Meanwhile, the fusion process shall be explainable to interpret the importance of the individual features. As the use of each feature introduces extra

---

*Equal contribution.
†Corresponding author: Xirong Li (xirong@ruc.edu.cn)

computational and storage overheads, the explainability is crucial for the fusion process to be selective to balance the performance and the cost.

Feature fusion is not new by itself. In fact, the topic has been extensively studied in varied contexts such as multimedia content analysis [2, 50] and multimodal or multi-view image classification [4, 64]. These earlier efforts focus on combining hand-crafted features, because such kinds of features are known to be domain-specific, suffering from the semantic gap problem [49], and thus insufficient for content representation when used alone. While current deep learning features are already more powerful than their predecessors, no single feature appears to rule all. Dark knowledge about objects and scenes is better carried in pre-trained 2D convolutional neural networks (2D-CNNs) [48], while 3D-CNNs are more suited for representing actions and motions [20]. For text-to-video retrieval, there are some initial efforts on combining diverse deep video features, *e.g.* JE [41,42], CE [35] and MMT [19], whilst W2VV++ [28] and SEA [31] show the potential of combining different text features for better query representation. The recent CLIP series [17, 36], due to their end-to-end learning paradigm, actually lacks the ability of exploiting existing features. Therefore, even in the era of deep learning, the need for feature fusion remains strong.

Concerning approaches to feature fusion, vector concatenation is commonly used when combining features at an early stage [15, 28]. As for late fusion, multiple feature-specific common spaces are learned in parallel, with the resultant similarities combined either by averaging [31, 42], empirical weighting [41] or by Mixture of Experts (MoE) ensembles [35]. As the number of features grows, vector concatenation suffers from the curse of dimensionality, while constructing common spaces per feature lacks inter-feature interactions. Moreover, the prior works focus either on the video end or on the text end. To the best of our knowledge, no attempt is made to develop a unified learning-based approach that works for both ends in the context of text-to-video retrieval, see Tab. 1.

One might consider feature fusion by Multi-head Self-Attention (MHSA), the cornerstone of Transformers [56]. As Fig. 1(a) shows, MHSA transforms a specific feature by blending it with information from all other features, with the blending weights produced by a self-attention mechanism termed QKV. Note that the module was initially developed for NLP tasks, for which exploiting element-wise correlations is crucial for resolving semantic ambiguity. However, as video features extracted by distinct 2D-CNNs and 3D-CNNs are meant for describing the video content from different aspects, we conjecture that optimizing their combination is preferred to modeling their correlations. Moreover, the self-attention in MHSA, computed by $\text{Softmax}((\frac{QK^T}{\sqrt{d_v}})V)$, depends largely on inter-feature correlations. It thus tends to have a group effect that features related to each other will be more attended. Consequently, the related yet relatively weak features will be over-emphasized. Hence, despite its high prevalence in varied contexts, we consider MHSA suboptimal for the current task.

We propose in this paper a much simplified feature fusion block, termed Lightweight Attention Feature Fusion (LAFF), see Fig. 1(c). LAFF is generic, working for both video and text ends. Video/text features are combined in a
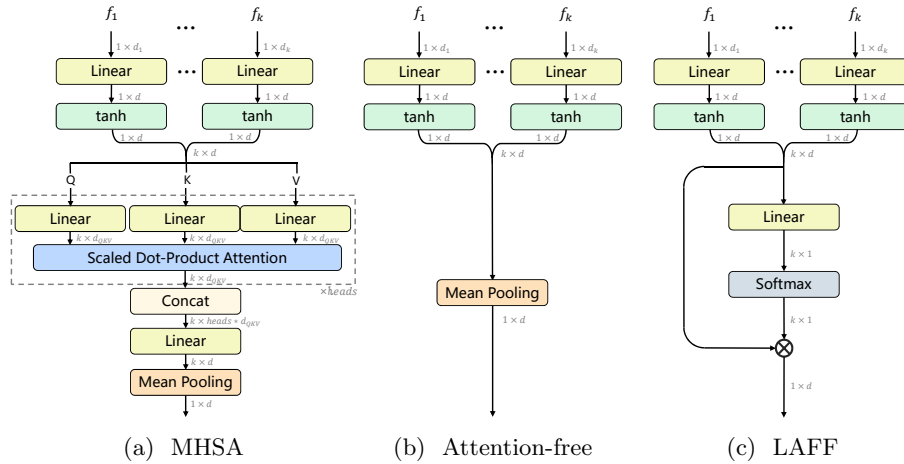
Fig. 1: **Three distinct blocks for feature fusion**: (a) Multi-Head Self-Attention (MHSA), (b) Attention-free, and (c) our proposed LAFF.

convex manner in a specific LAFF block, with the combination weights learned to optimize cross-modal text-to-video matching. Performing fusion at the feature level, LAFF can thus be viewed as an early fusion method. Meanwhile, with the multi-head trick as used in MHSA, multiple LAFFs can be deployed within a single network, with their resultant similarities combined in a late fusion manner. The ability to perform feature fusion at both early and late stages and at both video and text ends makes LAFF a powerful method for exploiting diverse, multi-level (off-the-shelf) features for text-to-video retrieval. In sum, our main contributions are as follows:

• We are the first to study both video-end and text-end feature fusion for text-to-video retrieval. Given the increasing availability of deep vision/language models for feature extraction, this paper presents an effective mean to harness such dark knowledge for tackling the task.

• We propose LAFF, a lightweight feature fusion block, capable of performing fusion at both early and late stages. Compared to MHSA, LAFF is much more compact yet more effective. Its attentional weights can also be used for selecting fewer features, with the retrieval performance mostly preserved.

• Experiments on five benchmarks, *i.e.* MSR-VTT, MSVD, TGIF, VATEX and TRECVID AVS 2016-2020, show that the LAFF-based video retrieval model (Fig. 2) compares favorably against the state-of-the-art, resulting in a strong baseline for text-to-video retrieval. Code is available at GitHub[4].

---

[4] https://github.com/ruc-aimc-lab/laff

2   Related Work

Feature fusion for text-to-video retrieval. Previous methods on feature fusion focus either on the video end or on the text end, see Tab. 1. For video-end feature fusion, earlier works often simply use vector concatenation to merge multiple features in advance to cross-modal representation learning [14]. JE [42] and its journal extension [41] have made an initial attempt to combine diverse video features by late fusion, where multiple feature-specific common spaces are learned. Multi-space similarities are either averaged [42] or linearly combined with empirical weights [41]. CE [35] also uses late fusion, but resorts to a learning based method, *i.e.* Mixture of Experts (MoE), to determine the fusion weights on the fly. MMT [19] improves over CE by first using a multi-modal Transformer to aggregate features from the frame level to the video level, and later using MoE for combining multi-space similarities. JE, CE and MMT all use a single text feature, so these methods leave text-end feature fusion untouched.

As for text-end feature fusion, W2VV++ [28] takes an early fusion approach, combining the output of three text encoders, *i.e.* bag-of-words, word2vec and GRU, by vector concatenation. By contrast, SEA [31] opts for late fusion, first building a common space per text feature and then averaging the similarities computed within the individual spaces. The more recent TEACHTEXT [11] first trains multiple CEs with different text encoders, and later combines these models by late fusion with MoE-predicted weights.

Table 1: **Taxonomy of feature fusion methods for text-to-video retrieval**. Note that feature fusion is conceptually different from multi-level feature learning, *e.g.* JPoSE [59], PIE-Net [52] and Dual Encoding [15], where new features are first computed at varied levels from a single feature input and combined later. So research in that line is excluded.

Method	Feature modality	Fusion stage	Fusion block
JE, ICMR18 [42]	Video	Late fusion	Average
JE, IJMIR19 [41]	Video	Late fusion	Manual weights
CE, BMVC19 [35]	Video	Late fusion	MoE
MMT, ECCV20 [19]	Video	Hybrid fusion	Multimodal Transformer + MoE
W2VV++, MM19 [28]	Text	Early fusion	Vector concatenation
SEA, TMM21 [31]	Text	Late fusion	Average
TEACHTEXT, ICCV21 [11]	Text	Late fusion	MoE
This work	Text/Video	Hybrid fusion	LAFF

Attentional feature fusion in other contexts. LAFF is conceptually different from context gating modules [12, 40], which aim to re-weight each dimension of a given feature vector, and thus produce weight per dimension. By contrast, as LAFF is to combine multiple features, it produces weight per feature vector. LAFF is technically inspired by attention-based multiple instance learning (MIL) [22], wherein there is a need of aggregating multiple instance-

level features into a case-level feature. However, the text-to-video retrieval task differs from MIL in the following two aspects, making the attention-based MIL not directly applicable in the new context. First, instances in a MIL setting are of the same modality, *e.g.* patches taken from the same image [22] or images from an image array [27], so the instance-level features are homogeneous and directly comparable. By contrast, the video or text features to be fused are obtained by distinct feature extractors with varied feature dimensions and are thus incompatible. Second, MIL is typically exploited in the context of a classification task, so the case-level feature is used as input to a classification layer. By contrast, our fused features are meant for cross-modal matching. Hence, a feature fusion block at one end shall be used in pair with a fusion block at the other end. The technical novelty of LAFF lies in its overall design that effectively answers the unique challenges in video/text feature fusion for text-to-video retrieval. While combining homogeneous features have been studied in other contexts [25,34,58], our paper fills the gap between diverse feature fusion and text-to-video retrieval.

3   A New Baseline

We propose trainable feature fusion for both video and text ends. Specifically, suppose we have a specific video x represented by a set of k_1 video-level features, $\{f_{v,1}(x), \ldots, f_{v,k_1}(x)\}$, and a specific textual query q represented by a set of k_2 sentence-level features $\{f_{t,1}(q), \ldots, f_{t,k_2}(q)\}$. We shall construct two feature fusion blocks to encode respectively the video and the query into their d-dimensional cross-modal embeddings $e(x)$ and $e(q)$. Their semantic similarity $s(x,q)$ is measured in terms of the two embeddings accordingly, *i.e.*

$$\begin{cases} e(x) & := fusion_v(\{f_{v,1}(x), \ldots, f_{v,k_1}(x)\}), \\ e(q) & := fusion_t(\{f_{t,1}(q), \ldots, f_{t,k_2}(q)\}), \\ s(x,q) := similarity(e(x), e(q)). \end{cases} \tag{1}$$

As such, text-to-video retrieval for the given query q is achieved by sorting all videos in a test collection in light of their $s(x,q)$ in descending order. In what follows, we describe the proposed LAFF as a unified implementation of the *fusion* blocks in Eq. (1), followed by its detailed usage for text-to-video retrieval.

3.1   The LAFF Block

Without loss of generality, we are provided with a diverse set of k different features $\{f_1, \ldots, f_k\}$, sized as $d_1, \ldots, d_k$, respectively. As the features are obtained by distinct extractors and thus incompatible, we shall use a feature transformation layer to rectify the diverse features to be of the same length. To convert the i-th feature to a new d-dimensional feature, we use
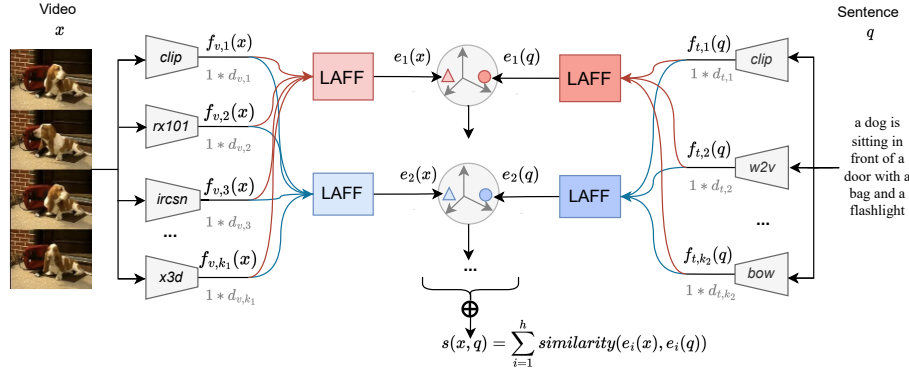
$$f'_i = \sigma(Linear_{d_i \times d}(f_i)), \tag{2}$$

Fig. 2: **Conceptual diagram of using paired LAFF blocks for text-to-video retrieval**. Given a specific video x, we employ multiple (pre-trained) feature extractors to obtain a set of k_1 video-level features $\{f_{v,1}(x), \ldots, f_{v,k_1}(x)\}$. In a similar manner we extract from a query sentence q a set of k_2 sentence-level features $\{f_{t,1}(q), \ldots, f_{t,k_2}(q)\}$. Each pair of LAFFs determines a common space. In particular, a specific pair of LAFFs, indexed by i, aggregates the video (sentence) features into a d-dimensional cross-modal feature $e_i(x)$ $(e_i(q))$, and consequently computes the video-text similarity $s_i(x, q)$ per space. The sum of the similarities from all h common spaces, $i.e.$ $\sum_{i=1}^{h} s_i(x, q)$, is used for retrieval.

where σ is a nonlinear activation function. As the output of the non-linear activation in LAFF is to calculate the cosine similarity, we use $tanh$ in this work[5]. The notation of $Linear_{d_i \times d}$ indicates a fully connected layer with an input of size d_i and an output of size of d. Each input feature has its own $Linear$, optional when d_i equals to d.

Although the transformed features $\{f_i'\}$ are now comparable, they are not equally important for representing the video/text content. We thus consider a weighted fusion, $i.e.$

$$\bar{f} = \sum_{i}^{k} a_i f_i', \tag{3}$$

with weights $\{a_1, \ldots, a_k\}$ computed by a lightweight attentional layer as follow,

$$\{a_1, \ldots, a_k\} = softmax(Linear_{d \times 1}(\{f_1', \ldots, f_k'\})). \tag{4}$$

As shown in Fig. 1(b), the Attention-free feature fusion block is a special case of LAFF when enforcing the weights in Eq. (3) to be uniform, $i.e.$ $a_i = \frac{1}{k}$. Compared to Attention-free, LAFF has d more parameters to learn, see Tab. 2.

---

[5]Other non-linear activations such as ReLU and sigmoid will make each dimension non-negative, constrain the feature space be in the first quadrant, and consequently put a lower boundary of 0 on the cosine similarity. As such, the similarity will be less discriminative than the $tanh$ counterpart.

Such a small amount of extra parameters turn out to be important for improving the effectiveness of feature fusion, as our ablation study will show. Compared with MHSA, LAFF has much fewer trainable parameters and is thus more data-efficient. Furthermore, as the attentional weights of LAFF are directly used for a convex combination of the features, LAFF is more interpretable than MHSA.

Table 2: **Complexity analysis of feature fusion blocks**, with D indicating the overall dimension of the input features and d as the dimension of the output features. FLOPs are computed based on input features of shape 8×2048.

Feature fusion block	Parameters	FLOPs (M)
MHSA	$D \times d + 4 \times d^2$	94.90
Attention-free	$D \times d$	27.78
LAFF	$D \times d + d$	27.80

3.2   Paired LAFFs for Text-to-Video Retrieval

Network Architecture We now detail the usage of LAFF for text-to-video retrieval. A straightforward solution is to substitute LAFF for the $fusion$ functions in Eq. (1). As such, we have a single configuration of how the video/text features are combined. However, due to the high complexity of the video and text contents, we hypothesize that the single configuration is suboptimal for cross-modal representation and matching. Borrowing the multi-head idea of MHSA, we consider multi-head LAFF. In particular, we deploy h pairs of LAFFs, where each pair of LAFFs jointly determine a latent common space for video-text matching. In particular, a specific pair of LAFFs, denoted as $< LAFF_{v,i}, LAFF_{t,i} >$, aggregates the video/text features into a d-dimensional cross-modal embedding vector $e_i(x)/e_i(q)$, $i.e.$

$$\begin{cases} e_i(x) = LAFF_{v,i}(x) \\ e_i(q) = LAFF_{t,i}(q) \\ s_i(x,q) = similarity(e_i(x), e_i(q)) \end{cases} \tag{5}$$

where $similarity$ is the widely used cosine similarity. Accordingly, we compute the final video-text similarity as the mean of the h individual similarities,

$$s(x,q) = \frac{1}{h} \sum_{i=1}^{h} s_i(x,q). \tag{6}$$

The overall architecture is illustrated in Fig. 2. In order to make the amount of trainable parameters invariant with respect to h, we set $d = \frac{d_0}{h}$, where d_0 is a constant empirically set to 2,048. As such, the multi-head version of LAFF is not an ensemble. We use $h = 8$, unless otherwise stated.

LAFF for multi-level feature fusion. So far we presume the features to be fused are already at the video level. In fact, for its high flexibility, LAFF can be extended with ease to a multi-level variant to deal with the situation wherein different frame-level and video-level features coexist. Fig. 3 shows this variant, which we term *LAFF-ml*. LAFF-ml works in a bottom-up manner, where a set of specific frame-level features are aggregated via a specific LAFF block to produce a video-level feature. Suppose there are two different frame-level features, *e.g. clip* and *rx101*. Each will have its own LAFF block. The (resultant) different video features are then fused via a video-level LAFF block.
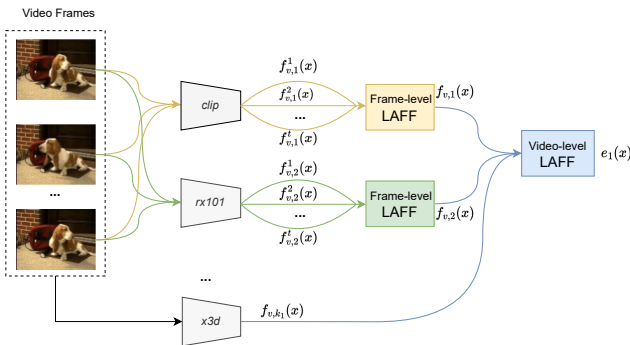

Fig. 3: **LAFF-ml for multi-level feature fusion**. Frame-level LAFF is applied per feature, *e.g. clip* or *rx101*. The outputs of the frame-level LAFF blocks are later combined (with other video-level features, *e.g. x3d*) by a video-level LAFF.

Network Training Following the good practice of the previous work, we adopt as our base loss function the triplet ranking loss with hard-negative mining [16]. For a specific sentence q in a given training batch, let x_+ and x_- be videos relevant and irrelevant *w.r.t.* q, and x_-^* be the hard negative that violates the ranking constraint the most. We have

$$\begin{cases} x_-^* & = \mathrm{argmax}_{x_-} \left(s(x_-, q) - s(x_+, q) \right) \\ \mathrm{loss}(q) = \max(0, \alpha + s(x_-^*, q) - s(x_+, q)), \end{cases} \tag{7}$$

where α is a positive hyper-parameter controlling the margin of the ranking loss.

As [31] has documented, when training a cross-modal network that produces multiple similarities, combining losses per similarity gives better results than using a single loss with the combined similarity. Hence, we follow this strategy, computing $\mathrm{loss}_i(q)$, namely the loss in the i-th space by substituting s_i for s in Eq. (7). The network is trained to minimize a combined loss $\sum_{i=1}^{h} \mathrm{loss}_i(q)$.

4 Experiments

In order to evaluate the effectiveness of LAFF, we conduct a series of experiments. An ablation study is performed on MSR-VTT, a *de facto* benchmark, to evaluate LAFF in multiple aspects. We then compare our LAFF-based retrieval model with the state-of-the-art on MSR-VTT and three other popular benchmarks including MSVD, TGIF and VATEX. In order to assess our retrieval method on a much larger collection, a post-competition evaluation is conducted on the TRECVID AVS benchmark series.

4.1 Common Setups

Implementation Details. Eight video features and five text features are used, see Tab. 3. The margin α in the loss is set to 0.2 according to VSE++ [16]. We perform SGD based training, with a mini-batch size of 128 and RMSProp as the optimizer. The learning rate is initially set to 10^{-4}, decayed by a factor of 0.99 per epoch. Following [23], we half the learning rate if the validation performance does not increase in three consecutive epochs. Early stop occurs when no validation performance increase is achieved in ten consecutive epochs. The dropout rate of the *Linear* layers is set to 0.2. All experiments were done with PyTorch (1.7.1) [45] on an Nvidia GEFORCE GTX 2080Ti GPU.

Evaluation Criteria. We report three standard rank-based metrics: Recall at Rank N (R@N, N=1, 5, 10), Median rank (Med r), and mean Average Precision (mAP) for assessing the overall ranking quality.

Table 3: **Video/text features used in ablation study**. Video-level features are obtained by mean pooling over frames or segments, unless otherwise stated.

Feature	Dim.	Short description
Video features:		
rx101	2,048	ResNeXt-101 trained on the full set of ImageNet [39].
re152	2,048	ResNet-152 from the MXNet model zoo.
wsl	2,048	ResNeXt-101 pre-trained by weakly supervised learning on 940 million public images, followed by fine-tuning on ImageNet1k [37].
clip	512	CLIP (ViT-B/32) pre-trained on web images and corpus by contrastive learning [48].
c3d	2,048	C3D trained on Kinetics400 [53].
ircsn	2,048	irCSN-152 which trained by weakly supervised learning on IG-65M [20].
tf	768	TimeSformer trained on HowTo100M [7].
x3d	2,048	X3D trained on Kinetics400 [18].
Text features:		
bow	m	m-dimensional Bag-of-words feature, with m being 7,675 (MSR-VTT), 2,916 (MSVD), 3,980 (TGIF), or 10,312 (VATEX)
w2v	500	Word2Vec trained on Flickr tags [14].
gru	1,024	Mean pooling over hidden vectors of GRU trained from scratch [14].
bert	768	The base version of BERT, pre-trained on BooksCorpus and English Wikipedia [13].
clip	512	The same CLIP as used to extract video features

4.2   Ablation Study

Our ablation study is conducted on MSR-VTT [63], which has 10k videos in total, each associated with 20 captions. We adopt the official data split: 6,513 videos for training, 497 videos for validation and the remaining 2,990 images for test. In order to distinguish this data split from other customized splits, *e.g.* JSFusion [67], we term the split **MV-test3k**.

    On Combining Diverse Video/Text Features. We investigate how LAFF responds when diverse video/text features are gradually added. For the ease of lateral comparison, we include as baselines the following two models: W2VV++ [28], which simply uses vector concatenation, and SEA [31] which learns cross-modal similarities per text feature.
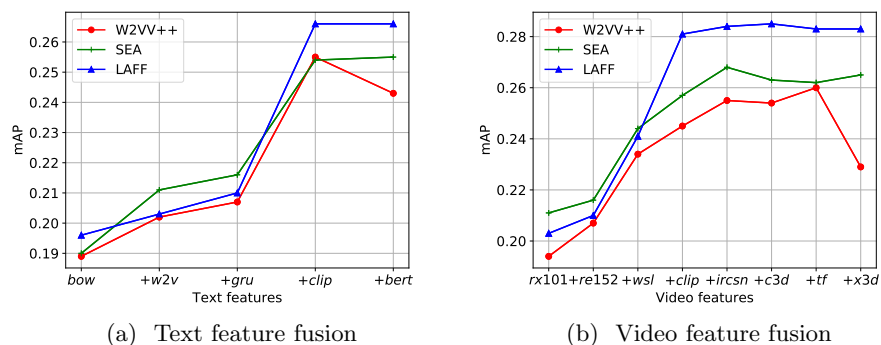

(a)   Text feature fusion   (b)   Video feature fusion

Fig. 4: **Performance curves of three distinct models, *i.e.* W2VV++, SEA and LAFF**, *w.r.t.* (a) text feature fusion, with {*rx101,re152*} as video features, and (b) video feature fusion, with {*bow,w2v,gru*} as text features. LAFF is both effective and stable for fusing diverse features. Data: MV-test3k.

    Given the many video and text features investigated in this work, a complete enumeration of video-text feature combinations is impractical. We choose to reduce the computation by only varying the features at one end, with features at the other end fixed. Fig. 4(a) shows the performance curves of W2VV++, SEA and LAFF *w.r.t.* text features, with {*rx101, re152*} as their common video features. The performance of all three models improves at the earlier steps when few features are fused. There is a noticeable drop in the performance curve of W2VV++ when *bert* is included. LAFF is more effective and more stable. Similar results can be observed from 4(b), which shows the performance curves of the three models *w.r.t.* video features. The above results justify the effectiveness of LAFF for combining diverse video/text features.

    Comparing Feature Fusion Blocks. We compare the three feature fusion blocks by replacing LAFF in Fig. 2 with MHSA and Attention-free, respectively. For a more fair comparison, we also apply the multi-loss trick on

MHSA by optimizing losses for different heads, denoted as MHSA(multi-loss). Moreover, we include as a baseline method that uses the simple feature concatenation strategy, as previously adopted in W2VV++ [28]. The performance of text-to-video retrieval with specific feature fusion blocks is reported in Tab. 4. LAFF performs the best, followed by Attention-free, the concatenation baseline and MHSA. Attention-free, while being extremely simple, is more effective than MHSA for combining the increasing amounts of text features, with its mAP increases from 0.264, 0.321 to 0.326. The superior performance of LAFF against Attention-free (0.358 *versus* 0.326) justifies the necessity of the attentional layer.

Table 4: **Comparing feature fusion blocks**. The simple feature concatenation used by W2VV++ is taken as a baseline. Numbers in parentheses are relative improvements against this baseline. Video features: all. Data: MV-test3k.

Text features	Fusion block	R1	R5	R10	Medr	mAP
bow, w2v, gru	Baseline	14.0	35.9	47.7	12	0.249
	MHSA	11.7	31.9	43.4	15	0.219 (12.0%↓)
	MHSA(mulit-loss)	11.1	30.1	41.1	18	0.207 (16.8%↓)
	Attention-free	15.4	37.8	49.7	11	0.264 (6.0%↑)
	LAFF	**16.0**	**39.5**	**51.4**	**10**	**0.276** (10.8%↑)
bow, w2v, gru, clip	Baseline	19.2	43.5	55.3	8	0.310
	MHSA	18.8	43.0	54.6	8	0.305 (1.6%↓)
	MHSA(mulit-loss)	18.7	43.1	54.5	8	0.305 (1.6%↓)
	Attention-free	20.5	44.8	56.2	7	0.321 (3.5%↑)
	LAFF	**23.7**	**49.1**	**60.6**	**6**	**0.358** (15.5%↑)
bow, w2v, gru, clip, bert	Baseline	14.5	35.0	46.1	13	0.247
	MHSA	17.9	41.6	53.3	9	0.294 (19.0%↑)
	MHSA (mulit-loss)	19.0	43.4	54.9	8	0.306 (23.9%↑)
	Attention-free	20.9	45.3	56.9	7	0.326 (32.0%↑)
	LAFF	**23.8**	**49.0**	**60.3**	**6**	**0.358** (44.9%↑)

LAFF Weights for Model Interpretability and Feature Selection. Fig. 5 visualizes the LAFF weights of videos and their associated captions selected from the MV-test3k test set. We observe that 3D-CNN features receive more weight when the video content contains more motions, see Fig. 5(b). For each feature, its weight averaged over samples reflects its contribution to the retrieval performance. The weights of text features in descending order are *clip* (64.3%), *bow* (15.7%), *gru* (9.5%), *w2v* (6.5%), *bert* (4.0%). For video feaetures, the order is *clip* (38.0%), *x3d* (16.8%), *ircsn* (13.3%), *tf* (10.9%), *rx101* (7.0%), *wsl*(6.6%) , *c3d* (5.1%), *re152* (1.4%). We re-train our model with the top-3 ranked video / text features. Compared to the full setup (mAP of 0.358), the reduced model obtains mAP of 0.353, meaning a relatively small performance loss of 1.4%. Hence, the LAFF weights are helpful for feature selection.

Combined Loss *versus* Single Loss. As Tab. 5 shows, LAFF trained with the combined loss produces a relative improvement of over 10% in terms of mAP, when compared to its single-loss counterpart.

Hillary Clinton gives a speech on race.

A girl does a cartwheel on her front lawn.

Two male commentators discussing a football game.

People are doing martial arts kicks.

■ bow ■ w2v ■ rnn ■ clip ■ bert   ■ rx101 ■ re152 ■ wsl ■ clip ■ c3d ■ ircsn ■ tf ■ x3d
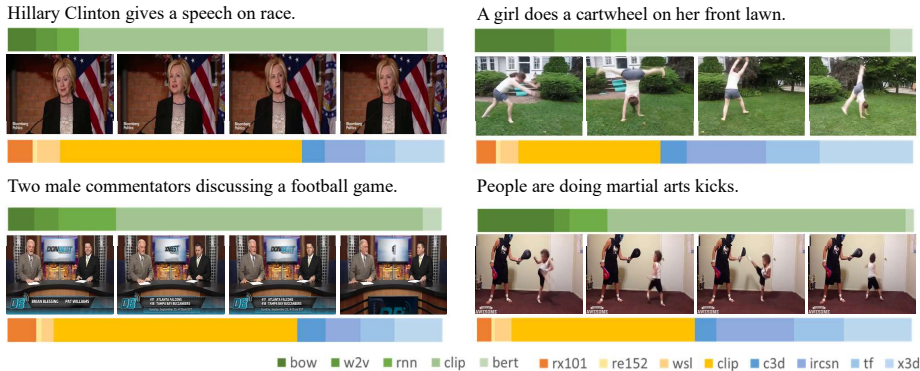
Fig. 5: **Visualization of LAFF weights per feature**, with samples from the MV-test3k test set. Green, brown, and blue mean text features, 2D video features and 3D video features, respectively. Best viewed in color.

Table 5: **Combined loss *versus* single loss**. Video features: all.

Text features	Loss	R1	R5	R10	Med r	mAP
bow, w2v, gru	Single	14.1	36.2	47.8	12	0.250
	Combined	**16.0**	**39.5**	**51.4**	**10**	**0.276** (10.4%↑)
bow, w2v, gru, clip	Single	20.6	45.0	56.5	7	0.324
	Combined	**23.7**	**49.1**	**60.6**	**6**	**0.358** (10.5%↑)

Table 6: **Effect of the number of common spaces** h. Features: all.

h	R1	R5	R10	Med r	mAP
1	23.1	48.3	59.9	6	0.352
2	23.1	48.3	60.1	6	0.352
4	23.5	49.1	60.4	6	0.356
8	**23.7**	**49.1**	**60.6**	6	**0.358**
16	23.5	48.8	60.4	6	0.356

The Effect of the Number of Common Spaces. Concerning the number of common spaces h, we try different values, *i.e.* {1, 2, 4, 8, 16}. As shown in Tab. 6, the performance improves as h increases, with the peak performance reached at $h = 8$. While using a larger h is beneficial, the relatively small gap between LAFF(h=1) and LAFF(h=8) suggests that the good performance of LAFF-based video retrieval is largely contributed by the LAFF block rather than the multi-space similarity.

To reveal how different are the embedding spaces to each other, we compute the Jaccard index between the top-5 video retrieval results of the individual spaces *w.r.t.* a specific query caption. The inter-space Jaccard index is lower than 0.5, suggesting sufficient divergence. Nevertheless, whether videos/captions have been separated along different axes needs further investigation.

4.3   Comparison with SOTA on Video Description Datasets

Datasets. We further include MSVD [9], TGIF [32] and VATEX [57]. For MSVD and TGIF, we follow their official data splits. For VATEX, we follow the data split as used in HGR [10]. As for MSR-VTT, in addition to the official MV-test3k split, we also report performance on another popular data split [67], with 9k videos for training and 1k for test. We term this split *MV-test1k*.

Baselines. We compare with the SOTA that uses the same data splits as we have mentioned. In particular, the following published models are included:

JE [41], W2VV++ [28], CE [35], TCE [65], HGR [10], SEA [31], MMT [19], DE [15], SSB [46], SSML [1], CLIP [47], CILP-FRL [8], and CLIP2Video (two-tower version[6]) [17]. In addition, we finetune CLIP per dataset, termed as CLIP-FT. The video/text feature extracted by CLIP-FT is denoted as *clip-ft*.

Note that the video/text features used in the above models vary. For a head-to-head comparison, we re-train LAFF, LAFF-ml, JE, W2V++, SEA and MMT using the same set of selected features[7]. Other baselines are not re-run, as they cannot handle the diverse video/text features without proper re-engineering. Since the fusion weights in JE have to be manually specified, we tried with three choices, *i.e.* JE with uniform weights, JE (0.8 for *clip-ft*) which assigns a much larger weight of 0.8 to the best *clip-ft* feature (the remains features have equal weights) and JE (0.9 for *clip-ft*). When comparing with CLIP2Video, we let LAFF include the global video/text features extracted by CLIP2Video to see whether LAFF can flexibly harness new and more powerful features.

Results. The performance of the different models on the multiple benchmarks is summarized in Tab. 7. Note that due to the inclusion of the better *clip-ft* feature, the performance is better than that reported in the ablation study. The baselines (JE, W2VV++ , SEA and MMT) get even worse results than using a single feature (*clip-ft*). The result suggests that one cannot take for granted that adding better features will yield better performance, and an intellectual design of feature fusion is needed. The proposed LAFF consistently performs the best on all the test sets. LAFF-ml outperforms LAFF, which shows that flexible use of LAFF in multiple levels can further improve performance.

Table 7: **Comparison with the state-of-the-art on four benchmark datasets**, *i.e.* MSR-VTT (MV-test3k/MV-test1k), MSVD, TGIF and VATEX. Baselines which use video/text features different from ours and thus not directly comparable are provided in the supplement.

Model	MV-test3k			MV-test1k			MSVD			TGIF			VATEX		
	R1	*R5*	*R10*	*R1*	*R5*	*R10*	*R1*	*R5*	*R10*	*R1*	*R5*	*R10*	*R1*	*R5*	*R10*
CLIP-FT (*this paper*)	27.7	53.0	64.2	39.7	67.8	78.4	44.6	74.7	84.1	21.5	40.6	49.9	53.3	87.5	94.0
The same video and text feature as ours															
JE [41] (uniform weights)	21.2	46.5	58.4	36.0	65.9	76.4	35.9	71.0	81.8	18.7	37.5	47.1	50.2	88.7	95.4
JE (0.8 for *clip-ft*)	26.1	51.7	63.3	41.2	73.2	82.5	39.4	69.9	79.4	21.7	41.3	50.9	54.1	89.0	95.0
JE (0.9 for *clip-ft*)	25.9	51.4	63.0	40.9	72.7	82.1	38.8	69.7	78.9	21.3	40.9	50.3	53.5	88.3	94.6
W2VV++ [28]	23.0	49.0	60.7	39.4	68.1	78.1	37.8	71.0	81.6	22.0	42.8	52.7	55.8	91.2	96.0
SEA [31]	19.9	44.3	56.5	37.2	67.1	78.3	34.5	68.8	80.5	16.4	33.6	42.5	52.4	90.2	95.9
MMT [19]	24.9	50.5	62.0	39.5	68.3	78.3	40.6	72.0	81.7	22.1	42.2	51.7	54.4	89.2	95.0
LAFF	28.0	53.8	64.9	42.2	70.7	**81.2**	45.2	75.8	84.3	24.1	44.7	54.3	57.7	91.3	95.9
LAFF-ml	**29.1**	**54.9**	**65.8**	**42.6**	**71.8**	81.0	**45.4**	**76.0**	**84.6**	**24.5**	**45.0**	**54.5**	**59.1**	**91.7**	**96.3**
Comparison with arXiv SOTA															
CLIP2Video [17]	n.a	n.a	n.a	44.5	71.3	80.6	44.7	74.8	83.7	n.a	n.a	n.a	54.8	89.1	95.1
LAFF	n.a	n.a	n.a	**45.8**	**71.5**	**82.0**	**45.4**	**75.5**	**84.1**	n.a	n.a	n.a	**58.3**	**91.7**	**96.3**

---

[6]We prefer the two-tower version to the single-tower version, as the latter has to compute video and text embeddings online, making it not scalable for real applications.

[7]Video features: *clip-ft*, *x3d*, *ircsn* and *tf*. Text features: *clip-ft*, *bow*, *w2v* and *gru*.

4.4   Comparison with SOTA on TRECVID AVS 2016-2020

Setup. The test collection for TRECVID AVS 2016-2018 (TV16/TV17/TV18) is IACC.3 [44] with 335,944 video clips. The test collection for TRECVID AVS 2019–2020 (TV19/TV20) is V3C1 [6] with 1,082,659 video clips. We use the official metric, *i.e.* inferred Average Precision (infAP) [66].

Baselines. Due to the prominent performance of CLIP-FT and CLIP2Video as shown in section 4.3, we again compare with the two models. Since the top-3 ranked solutions of the AVS evaluation naturally reflect the state-of-the-art, we include them as well. CLIP2Video was trained on the MV-test1k split. So for a fair comparison, we train LAFF and CLIP-FT on the same split.

Table 8: **State-of-the-art performance on TRECVID AVS 2016–2020.**

Model	TV16	TV17	TV18	TV19	TV20	MEAN
Rank 1	0.054 [24]	0.206 [51]	0.121 [26]	0.163 [62]	**0.354** [68]	n.a
Rank 2	0.051 [38]	0.159 [54]	0.087 [21]	0.160 [29]	0.269 [30]	n.a
Rank 3	0.040 [33]	0.120 [43]	0.082 [5]	0.123 [55]	0.229 [61]	n.a
CLIP2Video	0.176	0.229	0.114	0.176	0.207	0.180
CLIP-FT	0.191	0.215	0.105	0.147	0.203	0.172
LAFF	0.211	0.285	0.137	**0.192**	0.265	**0.218**
LAFF-ml	**0.222**	**0.290**	**0.147**	0.181	0.245	0.217

Results. As shown in Tab. 8, LAFF performs the best on TV16–TV19. Note that the top performer of TV20 was trained on the joint set of MSR-VTT, TGIF and VATEX. Re-training LAFF on this larger dataset results in infAP of 0.358, marginally better than the top performer. We also conduct a case study on TV20, which shows that LAFF outperforms the CLIP series for action related queries with a large margin, see supplementary materials. We attribute this result to the fact that LAFF integrates 3D-CNN features (*ircsn*, *c3d* and *tf*), which were designed to capture action and motion information in the video content.

5   Conclusions

For video retrieval by text, we propose LAFF, an extremely simple feature fusion block. LAFF is more effective than Multi-head Self-Attention, yet with much fewer parameters. Moreover, the attentional weights produced by LAFF can be used to explain the contribution of the individual video/text features for cross-modal matching. Consequently, the weights can be used for feature selection for building a more compact video retrieval model. Our LAFF-based video retrieval model surpasses the state-of-the-art on MSR-VTT, MSVD, TGIF, VATEX and TRECVID AVS 2016-2020. Given the increasing availability of (deep) video/text features, we believe our work opens up a promising avenue for further research.

References

1. Amrani, E., Ben-Ari, R., Rotman, D., Bronstein, A.: Noise estimation using density estimation for self-supervised multimodal learning. In: AAAI (2021)
2. Atrey, P.K., Hossain, M.A., El Saddik, A., Kankanhalli, M.S.: Multimodal fusion for multimedia analysis: A survey. Multimedia Systems **16**, 345–379 (2010)
3. Bain, M., Nagrani, A., Varol, G., Zisserman, A.: Frozen in time: A joint video and image encoder for end-to-end retrieval. In: ICCV (2021)
4. Baltrušaitis, T., Ahuja, C., Morency, L.P.: Multimodal machine learning: A survey and taxonomy. TPAMI **41**(2), 423–443 (2018)
5. Bastan, M., Shi, X., Gu, J., Heng, Z., Zhuo, C., Sng, D., Kot, A.: NTU ROSE lab at TRECVID 2018: Ad-hoc video search and video to text. In: TRECVID (2018)
6. Berns, F., Rossetto, L., Schoeffmann, K., Beecks, C., Awad, G.: V3C1 dataset: An evaluation of content characteristics. In: ICMR (2019)
7. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: ICML (2021)
8. Chen, A., Hu, F., Wang, Z., Zhou, F., Li, X.: What matters for ad-hoc video search? a large-scale evaluation on TRECVID. In: ICCV Workshop on ViRal (2021)
9. Chen, D., Dolan, W.: Collecting highly parallel data for paraphrase evaluation. In: CVPR (2011)
10. Chen, S., Zhao, Y., Jin, Q., Wu, Q.: Fine-grained video-text retrieval with hierarchical graph reasoning. In: CVPR (2020)
11. Croitoru, I., Bogolin, S.V., Leordeanu, M., Jin, H., Zisserman, A., Albanie, S., Liu, Y.: TEACHTEXT: Crossmodal generalized distillation for text-video retrieval. In: ICCV (2021)
12. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: ICML (2017)
13. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
14. Dong, J., Li, X., Snoek, C.G.: Predicting visual features from text for image and video caption retrieval. TMM **20**(12), 3377–3388 (2018)
15. Dong, J., Li, X., Xu, C., Yang, X., Yang, G., Wang, X.: Dual encoding for video retrieval by text. TPAMI (2021)
16. Faghri, F., Fleet, D.J., Kiros, J.R., Fidler, S.: VSE++: improving visual-semantic embeddings with hard negatives. In: BMVC (2018)
17. Fang, H., Xiong, P., Xu, L., Chen, Y.: CLIP2Video: Mastering video-text retrieval via image clip. arXiv preprint arXiv:2106.11097 (2021)
18. Feichtenhofer, C.: X3d: Expanding architectures for efficient video recognition. In: CVPR (2020)
19. Gabeur, V., Sun, C., Alahari, K., Schmid, C.: Multi-modal transformer for video retrieval. In: ECCV (2020)
20. Ghadiyaram, D., Tran, D., Mahajan, D.: Large-scale weakly-supervised pre-training for video action recognition. In: CVPR (2019)
21. Huang, P.Y., Liang, J., Vaibhav, V., Chang, X., Hauptmann, A.: Informedia@TRECVID 2018: Ad-hoc video search with discrete and continuous representations. In: TRECVID (2018)
22. Ilse, M., Tomczak, J., Welling, M.: Attention-based deep multiple instance learning. In: ICML (2018)
23. Joulin, A., van der Maaten, L., Jabri, A., Vasilache, N.: Learning visual features from large weakly supervised data. In: ECCV (2016)

24. Le, D.D., Phan, S., Nguyen, V.T., Renoust, B., Nguyen, T.A., Hoang, V.N., Ngo, T.D., Tran, M.T., Watanabe, Y., Klinkigt, M., Hiroke, A., Duong, Duc A. Miyao, Y., Satoh, S.: NII-HITACHI-UIT at TRECVID 2016. In: TRECVID (2016)

25. Li, H., Chen, J., Hu, R., Yu, M., Chen, H., Xu, Z.: Action recognition using visual attention with reinforcement learning. In: MMM (2019)

26. Li, X., Dong, J., Xu, C., Cao, J., Wang, X., Yang, G.: Renmin University of China and Zhejiang Gongshang University at TRECVID 2018: Deep Cross-Modal Embeddings for Video-Text Retrieval. In: TRECVID (2018)

27. Li, X., Wan, W., Zhou, Y., Zhao, J., Wei, Q., Rong, J., Zhou, P., Xu, L., Lang, L., Liu, Y., Niu, C., Ding, D., Jin, X.: Deep multiple instance learning with spatial attention for rop case classification, instance selection and abnormality localization. In: ICPR (2020)

28. Li, X., Xu, C., Yang, G., Chen, Z., Dong, J.: W2VV++: Fully deep learning for ad-hoc video search. In: ACMMM (2019)

29. Li, X., Ye, J., Xu, C., Yun, S., Zhang, L., Wang, X., Qian, R., Dong, J.: Renmin University of China and Zhejiang Gongshang University at TRECVID 2019: Learn to search and describe videos. In: TRECVID (2019)

30. Li, X., Zhou, F., Chen, A.: Renmin University of China at TRECVID 2020: Sentence Encoder Assembly for Ad-hoc Video Search. In: TRECVID (2020)

31. Li, X., Zhou, F., Xu, C., Ji, J., Yang, G.: SEA: Sentence encoder assembly for video retrieval by textual queries. TMM **23**, 4351–4362 (2021)

32. Li, Y., Song, Y., Cao, L., Tetreault, J.R., Goldberg, L., Jaimes, A., Luo, J.: TGIF: A new dataset and benchmark on animated gif description. In: CVPR (2015)

33. Liang, J., Chen, J., Huang, P., Li, X., Jiang, L., Lan, Z., Pan, P., Fan, H., Jin, Q., Sun, J., Chen, Y.: Informedia @ TRECVID 2016. In: TRECVID (2016)

34. Liu, M., Chen, X., Zhang, Y., Li, Y., Rehg, J.M.: Attention distillation for learning video representations. In: BMVC (2020)

35. Liu, Y., Albanie, S., Nagrani, A., Zisserman, A.: Use what you have: Video retrieval using representations from collaborative experts. In: BMVC (2019)

36. Luo, H., Ji, L., Zhong, M., Chen, Y., Lei, W., Duan, N., Li, T.: CLIP4Clip: An empirical study of clip for end to end video clip retrieval. arXiv preprint arXiv:2104.08860 (2021)

37. Mahajan, D., Girshick, R.B., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., van der Maaten, L.: Exploring the limits of weakly supervised pretraining. In: ECCV (2018)

38. Markatopoulou, F., Moumtzidou, A., Galanopoulos, D., Mironidis, T., Kaltsa, V., Ioannidou, A., Symeonidis, S., Avgerinakis, K., Andreadis, S., Gialampoukidis, I., Vrochidis, S., Briassouli, A., Mezaris, V., Kompatsiaris, I., Patras, I.: ITI-CERTH participation in TRECVID 2016. In: TRECVID (2016)

39. Mettes, P., Koelma, D.C., Snoek, C.G.M.: Shuffled ImageNet banks for video event detection and search. TOMM **16**(2), 1–21 (2020)

40. Miech, A., Laptev, I., Sivic, J.: Learning a text-video embedding from incomplete and heterogeneous data. arXiv (2018)

41. Mithun, N.C., Li, J., Metze, F., Roy-Chowdhury, A.K.: Joint embeddings with multimodal cues for video-text retrieval. IJMIR pp. 3–18 (2019)

42. 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)

43. Nguyen, P.A., Li, Q., Cheng, Z.Q., Lu, Y.J., Zhang, H., Wu, X., Ngo, C.W.: Vireo @ TRECVID 2017: Video-to-text, ad-hoc video search and video hyperlinking. In: TRECVID (2017)

44. Over, P., Awad, G., Smeaton, A.F., Foley, C., Lanagan, J.: Creating a web-scale video collection for research. In: The 1st Workshop on Web-scale Multimedia Corpus (2009)
45. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)
46. Patrick, M., Huang, P., Asano, Y.M., Metze, F., Hauptmann, A.G., Henriques, J.F., Vedaldi, A.: Support-set bottlenecks for video-text representation learning. In: ICLR (2021)
47. Portillo-Quintero, J.A., Ortiz-Bayliss, J.C., Terashima-Marín, H.: A straightforward framework for video retrieval using CLIP. In: MCPR (2021)
48. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML (2021)
49. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., Jain, R.C.: Content-based image retrieval at the end of the early years. TPAMI $\mathbf{22}$(12), 1349–1380 (2000)
50. Snoek, C.G.M., Worring, M.: Multimodal video indexing: A review of the state-of-the-art. Multimedia Tools Appl. $\mathbf{25}$(1), 5–35 (2005)
51. Snoek, C.G., Li, X., Xu, C., Koelma, D.C.: University of Amsterdam and Renmin university at TRECVID 2017: Searching video, detecting events and describing video. In: TRECVID (2017)
52. Song, Y., Soleymani, M.: Polysemous visual-semantic embedding for cross-modal retrieval. In: CVPR (2019)
53. Tran, D., Bourdev, L.D., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV (2015)
54. Ueki, K., Hirakawa, K., Kikuchi, K., Ogawa, T., Kobayashi, T.: Waseda_Meisei at TRECVID 2017: Ad-hoc video search. In: TRECVID (2017)
55. Ueki, Kazuya an Hori, T., Kobayashi, T.: Waseda_Meisei_SoftBank at TRECVID 2019: Ad-hoc video search. In: TRECVID (2019)
56. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
57. Wang, X., Wu, J., Chen, J., Li, L., Wang, Y.F., Wang, W.Y.: VATEX: A large-scale, high-quality multilingual dataset for video-and-language research. In: ICCV (2019)
58. Woo, S., Park, J., Lee, J., Kweon, I.S.: CBAM: Convolutional block attention module. In: ECCV (2018)
59. Wray, M., Larlus, D., Csurka, G., Damen, D.: Fine-grained action retrieval through multiple parts-of-speech embeddings. In: ICCV (2019)
60. Wu, J., Ngo, C.W.: Interpretable embedding for ad-hoc video search. In: ACMMM (2020)
61. Wu, J., Nguyen, P.A., Ngo, C.W.: Vireo@ TRECVID 2020 ad-hoc video search. In: TRECVID (2020)
62. Wu, X., Chen, D., He, Y., Xue, H., Song, M., Mao, F.: Hybrid sequence encoder for text based video retrieval. In: TRECVID (2019)
63. Xu, J., Mei, T., Yao, T., Rui, Y.: MSR-VTT: A large video description dataset for bridging video and language. In: CVPR (2016)
64. Xue, X., Nie, F., Wang, S., Chang, X., Stantic, B., Yao, M.: Multi-view correlated feature learning by uncovering shared component. In: AAAI (2017)

65. Yang, X., Dong, J., Cao, Y., Wang, X., Wang, M., Chua, T.S.: Tree-augmented cross-modal encoding for complex-query video retrieval. In: SIGIR (2020)
66. Yilmaz, E., Aslam, J.A.: Estimating average precision with incomplete and imperfect judgments. In: CIKM (2006)
67. Yu, Y., Kim, J., Kim, G.: A joint sequence fusion model for video question answering and retrieval. In: ECCV (2018)
68. Zhao, Y., Song, Y., Chen, S., Jin, Q.: RUC_AIM3 at TRECVID 2020: Ad-hoc video search & video to text description. In: TRECVID (2020)