Are All Combinations Equal? Combining Textual and Visual Features with Multiple Space Learning for Text-Based Video Retrieval

Damianos Galanopoulos[(✉)] and Vasileios Mezaris

CERTH-ITI, 6th Km Charilaou-Thermi Road, P.O. BOX,
60361 Thessaloniki, Greece
{dgalanop,bmezaris}@iti.gr

Abstract. In this paper we tackle the cross-modal video retrieval problem and, more specifically, we focus on text-to-video retrieval. We investigate how to optimally combine multiple diverse textual and visual features into feature pairs that lead to generating multiple joint feature spaces, which encode text-video pairs into comparable representations. To learn these representations our proposed network architecture is trained by following a multiple space learning procedure. Moreover, at the retrieval stage, we introduce additional softmax operations for revising the inferred query-video similarities. Extensive experiments in several setups based on three large-scale datasets (IACC.3, V3C1, and MSR-VTT) lead to conclusions on how to best combine text-visual features and document the performance of the proposed network. (Source code is made publicly available at: https://github.com/bmezaris/TextToVideoRetrieval-TtimesV)

Keywords: Text-based video search · Cross-modal video retrieval · Feature encoders · Multiple space learning

1 Introduction

Cross-modal information retrieval refers to the task where queries from one or more modalities (e.g., text, audio etc.) are used to retrieve items from a different modality (e.g., images or videos). This paper focuses on text-video retrieval, a key sub-task of cross-modal retrieval. The text-video retrieval task aims to retrieve unlabeled videos using only textual descriptions as input. This supports real-life scenarios, such as a human user searching for a video he/she remembers having viewed in the past, e.g., "*I remember a video where a dog and a cat were laying down in front of a fireplace*", or, searching for a video never seen before, again by expressing their information needs in natural language, e.g., "*I would like to find a video where some kids are playing basketball in an open field*".

To perform text-video retrieval, typically the videos or video parts, along with the textual queries, need to be embedded into a joint latent feature space.

Early approaches to this task [16,26] tried to annotate both modalities with a set of pre-defined visual concepts, and retrieval was performed by comparing these annotations. With the rise of deep neural networks over the past years, the community turned to them. Although various DNN architectures have been proposed to this end, their general strategy is the same: encode text and video into one or more joint latent feature spaces where text-video similarities can be calculated.
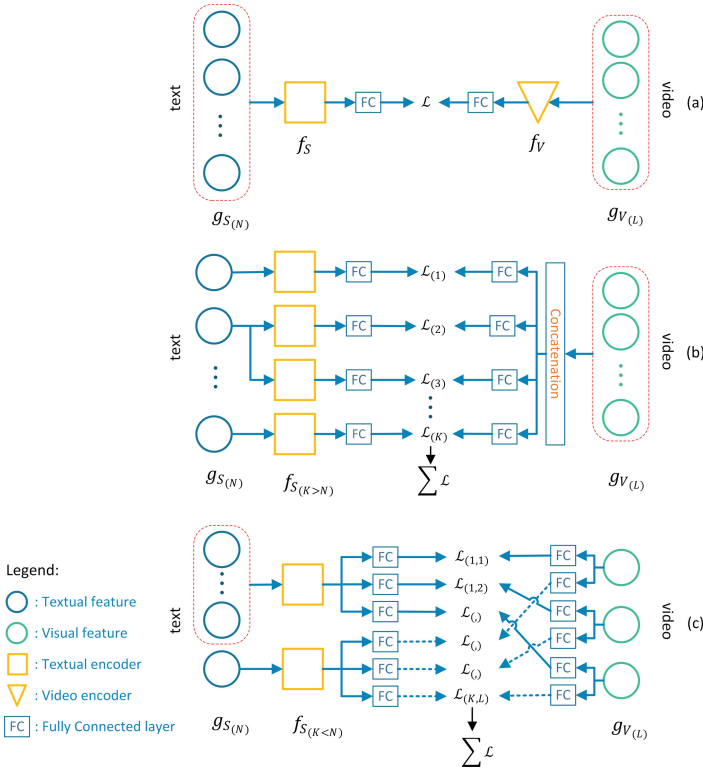
Fig. 1. Illustration of training different network architectures for dealing with the text-video retrieval problem using video-caption pairs. In all illustrations, $\mathcal{L}$ stands for the loss function. (a) All features are fed into one encoder per modality, (b) Every textual feature is used as input to a different encoder (or to more than one encoders), while visual features are simply concatenated, and (c) the proposed T×V approach, where various textual and visual features are selectively combined to create different joint spaces

State-of-the-art cross-modal video retrieval approaches utilize textual information by exploiting several textual features $g_s(\cdot)$ – extracted with the help of already-trained deep networks or non-trainable extractors – and encoding them through one or more trainable textual encoders (which are trained end-to-end as part of the overall cross-modal network training). The simple but widely used Bag-of-Words (bow) feature is often combined with embedding-based features

such as Word2vec [28] and Bert [7]. Typically, these features are used as input to a simple or more sophisticated trainable textual encoder $f_s(\cdot)$, e.g., [8,19], that encodes them into one single representation. Similarly to textual information processing, trained image or video networks (e.g. a ResNet-152 trained on ImageNet) are used to extract feature vectors $g_v(\cdot)$ from the video frames. Typically these features are then concatenated and used as input to a trainable video encoder $f_v(\cdot)$. Finally, the outputs of $f_s(\cdot)$ and $f_v(\cdot)$ (after a linear projection and a non-linear transformation) are embedded into a new joint space (Fig. 1a). Methods that follow this general methodology include [9,14,19].

In [20] a new approach of textual encoder assembly was proposed for exploiting diverse textual features. Instead of inputting all these features into a single textual encoder, an architecture where each textual feature is input into a different encoder (or to more than one encoders) was proposed, resulting in multiple joint latent spaces being created. However, when it comes to the video content, its treatment in [20] is much simpler: several video features derived from trained networks are combined via vector concatenation, and individual fully connected layers embed them into a number of joint feature spaces. The cross-modal similarity, which serves as the loss function, is calculated by summing the individual similarities in each latent space. Figure 1b illustrates the above architecture.

In terms of loss function, the majority of the proposed works, e.g. [9,13,19], utilize the improved marginal ranking loss introduced by [12]. This loss utilizes the hard-negative samples within a training batch to separate the positive samples from the samples that are negative but are located near to the positives. In [6] the dual softmax loss, a modification of the symmetric cross-entropy loss, was introduced. It is based on the assumption that the optimal text-video similarity is reached when the diagonal of a constructed similarity matrix contains the highest scores. So, this loss takes into consideration the cross-direction similarities within a training batch and revises the predicted text-video similarities.

In this work, inspired from [20] where multiple textual encoders are introduced, we propose a new cross-modal network architecture to explore the combination of multiple and heterogeneous textual and visual features. We expand the textual information processing strategy of [20], with adaptations, to the visual information processing as well, and we propose a multiple latent space learning approach, as illustrated in Fig. 1c. Moreover, inspired by the dual softmax loss of [6], we examine our network's performance when we introduce a dual softmax operation at the evaluation stage (contrarily to [6] that applies it to the network's training) and use it to revise the inferred text-video similarity scores. The contributions of this work are the following:

– We propose a new network architecture, named T×V, to efficiently combine textual and visual features using multiple loss learning for the text-based video retrieval task.
– We propose introducing a dual softmax operation at the retrieval stage for exploiting prior text-video similarities to revise the ones computed by the network.

2   Related Work

The general idea behind text-video retrieval is to project text and video information into comparable representations. Due to computational resources limitations, early approaches e.g. [17,24,26], dealt with relatively small datasets, and used pre-defined visual concepts as a stepping stone. I.e., videos and text were annotated with concepts, and text-video similarity was calculated by measuring the similarity between these annotations. With the explosion of deep learning, the state-of-the-art moved forward to proposing concept-free methods. The current dominant strategy is to encode both modalities into a joint latent feature space, where the text and video representations are compared.

In [10,14], dual encoding networks were proposed. Two similar sub-networks were introduced, one for the video stream and one for the text, to encode them into a joint feature space. In the dual-task network of [33], a combination of latent space encoding and concept representation, was proposed: the first task encodes text and video into a joint latent space, while the second task encodes video and text as a set of visual concepts. In [19] several textual features were used to create multiple textual encoders, instead of feeding them into a single encoder. In this way, multiple joint text-video latent feature spaces could be learned, leading to more accurate retrieval results. In [31] the problem of understanding textual or visual content with multiple meanings is addressed by combining global and local features through multi-head attention. More recently, inspired by the human reading strategy, [11] proposed a two-branches approach to encode video representations. A preview branch captures the overview information of a video, while the intense-reading branch is designed to extract more in-depth information. Moreover, the two branches interact, and the preview guides the intense-reading branch. As a general trend, the various recent works on text-video retrieval, e.g. [4,11,20], have shown that the utilization of multiple textual features to create more than one video-text joint spaces leads to improved overall performance.

Recent approaches additionally go beyond the standard evaluation protocol (i.e., training the network using the training portion of a dataset and testing it on the testing portion), benefiting from pre-training on further large-scale video-text datasets. This procedure leads to improved performance and learning transferable textual and visual representations. In [27], HowTo100M is introduced: a large-scale dataset of 100M web videos. Using this dataset to pre-train a baseline video retrieval network is shown in [27] to be beneficial. HiT [22] uses a transformer-based architecture to create a hierarchical cross-modal network for creating semantic-level and feature-level encoders. In [22] experimentation with and without pre-training also shows that the network's performance increases with the pre-training step. BridgeFormer [15] introduces a module that is trained to answer textual questions in order to be used as the pre-training step of a dual encoding network. Frozen [2], on the other hand, is based on a transformer architecture and does not use trained image DNNs as feature extractors. It did introduce, though, a large-scale video-text dataset (WebVid-2M) which was used for end-to-end pre-training of their network.

3   Proposed Approach

3.1   Overall Architecture

The text-video retrieval problem is formulated as follows: let $V = \{v_1, v_2, \ldots, v_T\}$ be a large set of T unlabeled video shots and s a free-text query. The goal of the task is, given the query s, to retrieve from V a ranked list with the most relevant video shots.

Our T×V network consists of two key sub-networks, one for the textual and one for the visual stream. The textual sub-network inputs a free-text query and vectorizes it into M textual features $g_S : \{g_s^1(\cdot), g_s^2(\cdot), \ldots, g_s^M(\cdot)\}$. These M features are used as input in a set of carefully-selected K textual encoders $f_S : \{f_s^1(\cdot), f_s^2(\cdot), \ldots, f_s^K(\cdot)\}$ that encode the input sentence. Each of these encoders can be either a trainable network or simply an identity function that just forwards its input. Similarly to the textual one, the visual sub-network inputs a video shot consisting of a sequence of N keyframes $v = \{I_1, I_2, \ldots, I_N\}$. We use L trained DNNs in order to extract the initial frame representations $g_V : \{g_v^1(\cdot), g_v^2(\cdot), \ldots, g_v^L(\cdot)\}$. To obtain video-shot level representations we follow the mean-pooling strategy.

Subsequently, we create all the possible textual encodings-visual feature pairs $(f_s^k(s), g_v^l(v))$ and a joint embedding space is created for each pair, using to this end two fully connected layers. Thus, $K \times L$ different joint spaces are created. The objective of our network is to learn a similarity function $similarity(s, v)$ that will consider every individual similarity in each joint latent space utilizing multi-loss-based training. Figure 1c illustrates our proposed method.

3.2   Multiple Space Learning

To encode the $(f_s^k(\cdot), g_v^l(\cdot))$ pair into its joint feature space, as shown in Fig. 1c, each single part of the pair is linearly transformed by a fully connected layer (FC). A non-linearity is added in the FC output (not illustrated in Fig. 1c for brevity), for which the ReLU activation function is used, as follows:

$$\mathbf{s}_k = ReLU(FC(f_s^k(\cdot))$$
$$\mathbf{v}_l = ReLU(FC(g_v^l(\cdot))$$

This transformation encodes the $(f_s^k(\cdot), g_v^l(\cdot))$ pair into its new joint feature space. The similarity function $sim(\mathbf{s}_k, \mathbf{v}_l)$ calculates the similarity between the output of textual encoder k and video feature l in this joint feature space. The overall similarity between a video-sentence pair is calculated as follows:

$$similarity(s, v) = \sum_{k=1}^{K} \sum_{l=1}^{L} sim(\mathbf{s}_k, \mathbf{v}_l)$$

where $sim(\mathbf{s}_k, \mathbf{v}_l) = cosine\_similarity(\mathbf{s}_k, \mathbf{v}_l)$.

To train our network, similarly to [10,14], we utilize the improved marginal ranking loss introduced in [12]. This emphasizes on the hard-negative samples in order to learn to maximize the similarity between textual and video embeddings. At the training stage, given a sentence-video sample (s, v), for a specific latent feature space (k, l), the improved marginal loss is defined as follows:

$$\mathcal{L}_{(k,l)}(s, v) = max(0, \alpha + sim(\mathbf{s}_k, \mathbf{v}_l') - sim(\mathbf{s}_k, \mathbf{v}_l))$$
$$+ max(0, \alpha + sim(\mathbf{s}_k', \mathbf{v}_l) - sim(\mathbf{s}_k, \mathbf{v}_l))$$

where $\mathbf{v}_l'$ and $\mathbf{s}_k'$ are the hardest negatives of $\mathbf{s}_l$ and $\mathbf{v}_k$ respectively and α is a hyperparameter for margin regulation. The overall training loss is calculated as the sum of all $K \times L$ individual loss values:

$$\mathcal{L}(s, v) = \sum_{k=1}^{K} \sum_{l=1}^{L} \mathcal{L}_{(k,l)}(s, v)$$
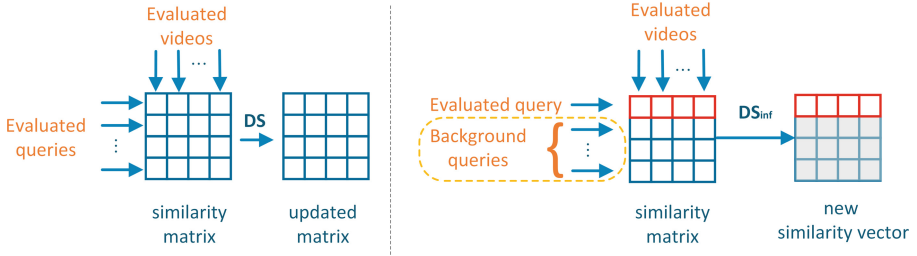
3.3   Dual Softmax Inference

In [6], a new objective function based on two softmax operations was proposed. According to this, at the training stage the predicted text-video similarities were revised by calculating a so-called *cross-direction* similarity matrix and multiplying it with the predicted one. Specifically, during a training batch, let Q be the number of examined caption-video pairs. By computing the similarities between every caption and all videos, a similarity matrix $\mathbf{X} \in \mathcal{R}^{Q \times Q}$ was generated. Next, by applying two cross-dimension softmax operations (one column-wise and one row-wise) an updated similarity matrix $\mathbf{X}'$ was calculated, and was subsequently used as discussed in [6]. Directly applying this approach at the inference stage, though, would require that all queries to be evaluated are known a priori and are evaluated simultaneously; they would need to be used for calculating matrix $\mathbf{X}$, as illustrated in the left part of Fig. 2. This is not a realistic scenario, especially in real-world retrieval applications.

To deal with this issue and revise the inferred text-video similarities at the retrieval stage, we propose a dual softmax-based inference ($\mathrm{DS_{inf}}$) as illustrated in the right side of Fig. 2. We utilize a fixed set of C pre-defined background textual queries, which are independent of the evaluated dataset, and we calculate their similarities with all D videos of the test set. For the set of background queries, we calculate once the similarity matrix $\mathbf{X}^* \in \mathcal{R}^{C \times D}$. For each individual evaluated query s a similarity vector $\mathbf{y}(s) = [similarity(s, v_1), similarity(s, v_2),$ $\ldots, similarity(s, v_D)]^T$, is calculated. A matrix $\mathbf{Z}(s) = concat(\mathbf{y}(s); \mathbf{X}^*)$ is constructed, and a dual softmax operation revises the similarities as follows:

$$\mathbf{Z}^*(s) = Softmax(\mathbf{Z}(s), \ dim = 0) \odot Softmax(\mathbf{Z}(s), \ dim = 1)$$

where $\odot$ denotes the Hadamard product. Finally, from matrix $\mathbf{Z}^*$ we extract the revised similarity vector $\mathbf{y}^* = [Z_{0,1}^*, Z_{0,2}^*, \cdot, Z_{0,D}^*]$ (Fig. 3). This normalization

procedure is meaningful when we expect that there are multiple positive video samples in our dataset for the evaluated query; thus, by normalizing the inferred similarities we can produce a better ranking list.

Fig. 2. Different approaches to update the similarity matrix using all the evaluated queries (left subfigure) and pre-defined background queries (right subfigure)
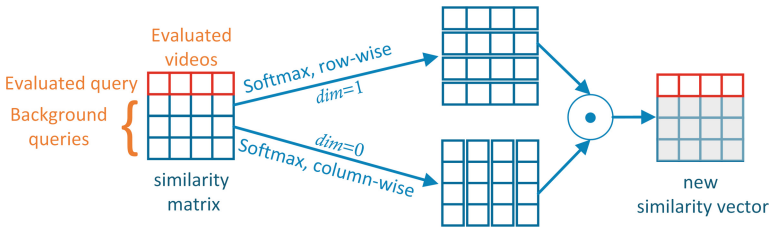
Fig. 3. Illustration of the dual softmax-based inference (DS_{inf}) approach for updating the similarities

3.4 Specifics of Textual Information Processing

In this section, we present every textual feature and every textual encoder we used in order to find the optimal textual encoder combination. Given a sentence s consisting of $\{w_1, w_2, \ldots, w_B\}$ words, we utilize $M = 4$ different textual features. These features are used as input to textual encoders.

Textual features

- Bag-of-Words (bow): We utilize Bag-of-Words to vectorize every sentence into a sparse vector representation expressing the occurrence frequency of every word from a pre-defined vocabulary.
- Word2Vec (w2v): Word2Vec model [28] is an established and well-performing word embedding model. W2v learns to embed words into a word-level representation vectors $\mathbf{W}_{w2v} : \{\mathbf{w}_1^{w2v}, \mathbf{w}_2^{w2v}, \ldots, \mathbf{w}_B^{w2v}\} \in R^{B x D_{w2v}}$. The overall sentence w2v embedding g_s^{w2v} is calculated as the mean pooling of the individual word embeddings.

– Bert: Bert [7] offers contextual embeddings by considering the sequence of all words inside a sentence, which means that the same word may have different embedding when the context of a sentence is different. We utilize the BASE variation of bert consisting of 12 encoders and 12 bidirectional self-attention heads. Similarly to [20], we calculate the bert sentence embedding g_s^{bert} by mean pooling the individual word embedding.

– Clip: The transformer-based trained model of CLIP [30] is used as a textual feature extractor. Sentence s is fed to it as a sequence of words $w_1, ..., w_B$ and token embeddings $\mathbf{W}_{clip} : \{\mathbf{w}_{startoftext}^{clip}, \mathbf{w}_1^{clip}, \ldots, \mathbf{w}_B^{clip}, \mathbf{w}_{endoftext}^{clip}\}$ are calculated. The last token embedding, $\mathbf{w}_{endoftext}^{clip} \in R^{512}$, is used as our feature vector.
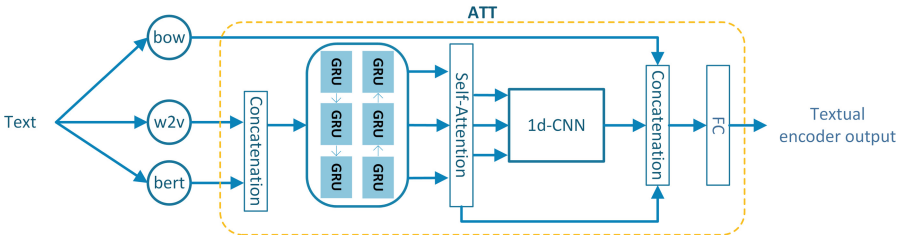
Fig. 4. Illustration of the ATT encoder, inputting several textual features and producing three levels of encodings (bow, self-attention bi-gru and CNN outputs) which are concatenated and contribute to the final output of the encoder

Textual Encoders. The textual encoders input the extracted textual features (one at a time, or a combination of them) and output a new embedding. We experimented with combinations of the followings:

– $f_s^{bow}, f_s^{w2v}, f_s^{bert}, f_s^{clip}$: these encoders feedforward the corresponding features through an identity layer.

– $f_s^{w2v-bert}$: The concatenation of w2v and bert features is used to feed an identity layer.

– f_s^{bi-gru}: A self-attention bi-gru module, introduced in [14], is trained as part of the complete network architecture; it takes as input the w2v features for each word and their temporal order (i.e., not using the overall sentence w2v embedding, contrarily to f_s^{w2v} and $f_s^{w2v-bert}$).

– Attention-based dual encoding network (ATT): The textual sub-network presented in [14] (illustrated in Fig. 4) is trained (similarly to f_s^{bi-gru}, above), taking as input the bow, w2v and bert features (again, for each individual word rather than the mean-pooled sentence embeddings) and producing a vector in $\mathcal{R}^{2048}$.

Among all the above possible textual encoders, we propose to combine in our network the f_s^{clip} and ATT ones, as experimentally verified in Sect. 4.

3.5   Specifics of Visual Information Processing

Similarly to the textual sub-network, we use several deep networks that have been trained for other visual tasks as frame feature extractors. Considering a video shot v, first we uniformly sample it with a fixed rate of 2 frames per second, resulting in a set of keyframes $\{I_1, I_2, \ldots, I_N\}$. Then, frame-level representations are obtained with the help of the feature extractors listed below, followed by mean pooling of the individual frame representations to get shot-level features.

Visual features

– R_152: The first video feature extractor inputs an image frame into a ResNet-152 [18] network, trained on the ImageNet-11k dataset. The flattened output of the layer before the last fully connected layer of the network is used as a feature representation of every frame in $\mathcal{R}^{2048}$.
– Rx_101: The second feature extractor utilizes a ResNeXt-101 network, pre-trained by weakly supervised learning on web images followed and fine-tuned on ImageNet [25]. Similarly to R_152, Rx_101 inputs frames and the frame representations in $\mathcal{R}^{2048}$, are obtained as the flattened output of the layer before the last fully connected layer.
– Clip: As third video feature extractor we utilise a trained CLIP model (ViT-B/32) [30], to create frame-level representations in $\mathcal{R}^{512}$.

As illustrated in Fig. 1c, we propose using these visual features, without introducing any trainable visual encoder, directly as input to a number of individual FC layers for learning the latent feature spaces. However, in order to examine more design choices, we also tested in our ablation experiments the introduction of a visual encoder, similarly to what we do for textual information processing. To this end, we utilized the visual sub-network of the attention-based dual encoding network of [14] (ATV). Following [14], we input all three aforementioned frame-level features to a single ATV encoder, which (similarly to the ATT one) was trained end-to-end as part of our overall network.

4   Experimental Results

4.1   Datasets and Experimental Setup

We evaluate our approach and report experimental results on three datasets: the two official TRECVID AVS datasets (i.e., IACC.3 and V3C1) [1] and MSR-VTT [34]. The AVS datasets are designed explicitly for text-based video retrieval evaluation; they include the definition of tens of textual queries as well as ground-truth associations of multiple positive samples with each query. The IACC.3 dataset consists of 335.944 test videos, and the V3C1 of 1.082.629 videos for testing (most of which are not associated with any of the textual queries). As

evaluation measure we use the mean extended inferred average precision (MxinfAP), as proposed in [1] and typically done when working with these datasets. On the other hand, MSR-VTT targets primarily video captioning evaluation, but is also often used for evaluating text-video retrieval methods. It is made of 10.000 videos and each video is annotated with 20 different natural language captions (totaling 200.000 captions, which are generally considered to be unique); for retrieval evaluation experiments, given a caption the goal is to retrieve the single video that is ground-truth-annotated with it. For the MSR-VTT experiments, following the relevant literature, we use as evaluation measures the recall $R@k$, $k = 1, 5, 10$, the median rank ($Medr$) and mean average precision (mAP).

Table 1. Results and comparisons on the IACC.3 and V3C1 datasets. Bold/underline indicates the best-/second-best scores

Model		Datasets (all scores: MxinfAP)						
		IACC.3			V3C1			
	Year	AVS16	AVS17	AVS18	AVS19	AVS20	AVS21	Mean
Dual encoding [9]	2019	0.165	0.228	0.117	0.152	–	–	–
ATT-ATV [14]	2020	0.164	0.243	0.127	–	–	–	–
ATT-ATV [14] (re-training)	–	0.202	0.281	0.146	<u>0.208</u>	0.283	<u>0.289</u>	0.235
Dual-task [33]	2020	0.185	0.241	0.123	0.185	–	–	–
SEA [20]	2021	0.164	0.228	0.125	0.167	–	–	–
SEA [20] (re-training)	–	<u>0.207</u>	0.279	0.148	0.191	0.283	0.283	0.232
SEA-clip [4]	2021	0.203	**0.321**	**0.156**	0.192	**0.329**	–	–
Extended Dual Encoding [10]	2022	0.159	0.244	0.126	–	–	–	–
T×V (proposed)		**0.234**	<u>0.317</u>	<u>0.153</u>	**0.220**	<u>0.316</u>	**0.312**	**0.259**

Regarding the training/testing splits: for the evaluations on the AVS datasets, our cross-modal network (and any network of the literature, i.e. [14,20], that we re-train for comparison) is trained using a combination of four other large-scale video captioning datasets: MSR-VTTT [34], TGIF [21], ActivityNet [3] and Vatex [32]. For validation purposes, during training, we use the Video-to-Text Description dataset of TRECVID 2016. For testing, all sets of queries specified by NIST for IACC.3 (i.e., AVS16, AVS17 and AVS18) and V3C1 (i.e., AVS19, AVS20 and AVS21) are used. For the evaluations on the MSR-VTT dataset, we experimented with two versions of this dataset: MST-VTT-full [34] and MSR-VTT-1k-A [36]. MST-VTT-full consists of 6.513 videos for training, 497 for validation and 2.990 videos (thus, 2.990×20 video-caption pairs) for testing. On the other hand, MSR-VTT-1k-A contains 9.000 videos for training and 1.000 video-caption pairs for testing. For both MSR-VTT versions, we trained our network of the training portion of the dataset and report results on the testing portion, respectively.

Regarding the training conditions and parameters: To train the proposed network, (and again, also for re-training [14,20]) we adopt the setup of [13], where

six configurations of the same architecture with different training parameters were combined. Specifically, each model is trained using two optimizers, i.e., Adam and RMSprop, and three learning rates (1×10^4, 5×10^5, 1×10^5). The final results for a given architecture are obtained by combining the six returned ranking lists of the individual configurations in a late fusion scheme, i.e. by averaging the six obtained ranks for each video. For training all configurations, we follow a learning rate decay technique, and we reduce the learning rate 1% per epoch or by 50% if the validation performance does not improve for three epochs. The dropout rate is set to 0.2 to reduce overfitting. Also, following [12] the margin parameter on loss function is set to $\alpha = 0.2$. All experiments were performed on a single computer equipped with Nvidia RTX3090 GPU. Our models were implemented and trained using Pytorch 1.11.

Table 2. Results and comparisons between different encoder strategies when using dual softmax-based inference (DS_{inf}) similarity. Scores in bold/underline indicate the best-/second-best-performing strategy

Model		Datasets (all scores: MxinfAP)						
		IACC.3			V3C1			
		AVS16	AVS17	AVS18	AVS19	AVS20	AVS21	Mean
T×V		0.234	0.317	0.153	0.220	0.316	0.312	0.259
T×V + DS$_{inf}$ on the set of evaluated queries		**0.244**	**0.330**	**0.165**	**0.226**	<u>0.324</u>	**0.324**	**0.269**
T×V + DS$_{inf}$ using as background queries:	60 random captions	0.240	0.323	0.157	0.223	0.318	0.313	0.262
	200 random captions	0.239	0.322	0.158	0.224	0.315	0.315	0.262
	other years' queries of the same dataset	<u>0.243</u>	<u>0.328</u>	<u>0.162</u>	<u>0.226</u>	**0.325**	<u>0.323</u>	<u>0.268</u>

4.2 Results and Comparisons

Table 1 presents the results of the proposed T×V network, i.e. using three visual features (R_152, Rx_101, clip) followed by FC layers and clip and ATT as textual encoders, on IACC.3 and V3C1 datasets and comparisons with state-of-the-art literature approaches. We compare our method with six methods and the presented results are extracted from their original papers. Furthermore, as [4] has shown that the quality of the initial visual features is crucial for the performance of a method, we present results of re-training the ATT-ATV [14] and SEA [20] networks using the same visual features and same training datasets as we did in our experiments, using their publicly available code. Our proposed network outperforms the competitors on AVS16, AVS19 and AVS21. SEA-clip [4] achieves better results on AVS17, AVS18 and AVS20 by exploiting 3D CNN-based visual features. Comparing the mean performance on AVS16-AVS20 (since SEA-clip does not report results on AVS21), our network achieves MxinfAP equal to 0.248 while SEA-clip 0.240.

Table 3. Results and comparisons on the MSR-VTT full and 1k-A datasets. Methods marked with * use an alternative training set of 7.010 video-caption samples for the 1k-A dataset, but still report results on the same test portion of 1k-A as all other methods. Bold/underline indicates the best-/second-best scores

Model	Year	Datasets									
		MSR-VTT full					MSR-VTT 1k-A				
		R@1↑	R@5↑	R@10↑	Medr↓	mAP↑	R@1↑	R@5↑	R@10↑	Medr↓	mAP↑
W2VV [8]	2018	1.1	4.7	8.1	236	3.7	1.9	9.9	15.2	79	6.8
VSE++ [12]	2018	8.7	24.3	34.1	28	16.9	16.0	38.5	50.9	10	27.4
CE [23]	2019	7.9	23.6	34.6	23	16.5	17.2	46.2	58.5	7	30.3
W2VV++ [19]	2019	11.1	29.6	40.5	18	20.6	19.0	45.0	58.7	7	31.8
TCE [35]	2020	9.3	27.3	38.6	19	18.7	17.8	46.0	58.3	7	31.1
HGR [5]	2020	11.1	30.5	42.1	16	20.8	21.7	47.4	61.1	6	34.0
SEA [20] *	2021	12.4	32.1	43.3	15	22.3	23.8	50.3	63.8	5	36.6
HiT [22]	2021	-	-	-	-	–	27.7	59.2	72.0	3	–
HiT (Pre-train. on HT100M) [22]	2021	–	–	–	–	-	30.7	60.9	73.2	2.6	-
CLIP [29]*	2021	**21.4**	_41.1_	_50.4_	_10_	–	31.2	53.7	64.2	4	–
FROZEN [2]	2021	–	–	–	–	–	31.0	59.5	70.5	3	–
Extended Dual Encoding [10]*	2022	11.6	30.3	41.3	17	21.2	21.1	48.7	60.2	6	33.6
RIVRL [11] *	2022	13.0	33.4	44.8	14	23.2	23.3	52.2	63.8	5	36.7
RIVRL with bert [11] *	2022	13.8	35.0	46.5	13	_24.3_	27.9	59.3	71.2	4	42.0
BridgeFormer [15]	2022	–	–	–	–	–	**37.6**	_64.8_	71.2	3	–
T×V (proposed) *		–	–	–	–	–	32.3	63.7	_74.6_	3	_46.3_
T×V (proposed)		_21.2_	**46.3**	**58.2**	**7**	**33.1**	_36.5_	**66.9**	**77.7**	**2**	**50.2**

In Table 2 we experiment with the dual softmax-based (DS_{inf}) inference on IACC.3 and V3C1. We examine the impact of different background query strategies and compare with the proposed network without DS_{inf}. "DS_{inf} on the set of evaluated queries" indicates that the operation was performed using the same year's queries. For example, to retrieve videos for an AVS16 query, the remaining AVS16 queries are used as background queries. This improves the overall performance, but as we already have discussed, is not a realistic application scenario. To overcome this problem, we experiment with using a fixed set of background queries: 60 or 200 randomly selected captions extracted from the training datasets. By using these captions, performance is improved compared to the proposed network in every dataset and every year. The difference between using 60 and 200 captions is marginal. Finally, we try using as background queries all AVS queries defined for the same video dataset but not the same test-year as the examined query. For example, to evaluate each of the AVS16 queries, we use the AVS17 and AVS18 queries as background. This strategy achieves the second-best results among all examined ones, being marginally outperformed by "DS_{inf} on the set of evaluated queries" strategy; and, contrarily to the latter, does not assume knowledge of all the evaluation queries beforehand. For a given query, the retrieval time using DS_{inf} increases (on average) from 0.4 s to 0.6 s on the IACC.3 dataset and from 1.1 s to 1.3 s on the V3C1, respectively.

In Table 3 we present results on the MSR-VTT full and 1k-A datasets and compare with literature methods. Our network outperforms most literature methods, even methods like FROZEN [2] and HiT (pre-trained on HT100M)

Table 4. Comparison of combinations of textual encoders and visual features on the IACC.3 and V3C1 datasets

Model			Datasets (all scores: MxinfAP)						
Visual features combination strategy	Textual encoders (Fig. 1c)	# of feature spaces	IACC.3			V3C1			
			AVS16	AVS17	AVS18	AVS19	AVS20	AVS21	Mean
(feat. concat. + ATV) ↓ FCs (Fig. 1a)	bow, w2v, bert, bi-gru	4	0.169	0.250	0.126	0.170	0.252	0.251	0.203
	bow, w2v, bert, bi-gru, clip	5	0.179	0.265	0.133	0.174	0.260	0.254	0.211
	w2v-bert, clip, ATT	3	0.199	0.265	0.135	0.183	0.271	0.262	0.219
	clip, ATT	2	0.211	0.288	0.143	0.188	0.288	0.271	0.232
(feat. concat.) ↓ FCs (Fig. 1b)	bow, w2v, bert, bi-gru	4	0.184	0.267	0.136	0.202	0.303	0.295	0.231
	bow, w2v, bert, bi-gru, clip	5	0.198	0.275	0.141	0.205	0.305	0.292	0.236
	w2v-bert, clip, ATT	3	0.210	0.281	0.147	0.194	0.290	0.275	0.233
	clip, ATT	2	0.227	0.291	0.149	0.190	0.296	0.292	0.241
Only FCs (Fig. 1c)	**bow, w2v, bert, bi-gru**, clip	13	**0.205**	**0.306**	**0.150**	**0.203**	**0.305**	**0.299**	**0.239**
	w2v-bert, clip, ATT	9	0.219	0.312	0.150	0.210	0.307	0.297	0.249
	clip, ATT	6	**0.234**	**0.317**	**0.153**	**0.220**	**0.316**	**0.312**	**0.259**

[22] in which their networks utilize a pre-training step on other large text-video datasets. Moreover, BridgeFormer [15], using a pre-training step on the WebVid-2M [2] dataset, marginally outperforms our network in $R@1$ terms, while our approach achieves better results on the remaining evaluation measures. Finally, we should note that experiments with DS_{inf} on MSR-VTT (not shown in Table 3) lead to only marginal differences in relation to the results of the proposed $T \times V$ network. This is expected because of the nature of MSR-VTT: given a caption the goal is to retrieve the *single* video that is ground-truth-annotated with it; thus, re-ordering the entire ranking list by introducing the DS_{inf} normalization of the caption-video similarities has limited impact.

4.3 Ablation Study

In this section we study the effectiveness of different textual encoders and visual features (or also encoders) combinations. We report results using three visual feature encoding strategies: "*feat. concat. + ATV*" indicates the early fusion of the three visual features that are then fed into the trainable ATV sub-network of [14] (as illustrated in the rightmost part of Fig. 1a), followed by the required FC layers in order to encode the ATV's output into the corresponding joint feature spaces. Similarly, "*feat. concat.*" refers to the early fusion of the three visual features (as illustrated in the rightmost part of Fig. 1b) followed by the required FC layers. Finally, "*only FCs*" refers to the proposed strategy (Fig. 1c), where visual features are individually and directly encoded to the joint spaces using only FC layers.

In Table 4 we report the results on the AVS datasets when using different combinations of textual encoders together with the aforementioned three possible visual encoding strategies. Concerning the visual encoding strategies, the results indicate that the lowest performance is achieved by the "*feat. concat. + ATV*" strategy, regardless of the textual encoders choice. When the early fusion of the trained models "*feat. concat.*" is used instead of ATV, the performance consistently increases. The best results are achieved by forwarding the visual

Table 5. Comparison of combinations of textual encoders and visual features on the full and 1k-A variations of the MSR-VTT dataset

Model		Datasets									
		MSR-VTT full					MSR-VTT 1k-A				
Visual features combination strategy	Textual encoders (Fig. 1c)	R@1↑	R@5↑	R@10↑	Medr↓	mAP↑	R@1↑	R@5↑	R@10↑	Medr↓	mAP↑
(feat. concat. + ATV) ↓ FCs (Fig. 1a)	bow, w2v, bert, bi-gru	17.3	40.9	52.9	9	28.9	32.2	62.9	73.7	3	46.3
	bow, w2v, bert, bi-gru, clip	19.2	43.6	55.6	8	30.1	34.8	63.9	75.8	3	48.6
	w2v-bert, clip, ATT	21.4	46.6	58.5	7	33.4	36.5	67.4	76.4	2	50.3
	clip, ATT	**22.3**	**47.6**	**59.3**	**6**	**34.3**	**40.1**	**68.4**	**78.1**	**2**	**52.9**
(feat. concat.) ↓ FCs (Fig. 1b)	bow, w2v, bert, bi-gru	17.6	41.3	53.2	9	29.0	32.8	64.4	74.3	3	47.1
	bow, w2v, bert, bi-gru, clip	19.7	44.1	55.9	8	31.4	35.4	65.0	76.0	3	48.5
	w2v-bert, clip, ATT	21.4	46.3	57.7	7	33.3	35.7	67.4	77.4	3	49.8
	clip, ATT	22.0	46.5	57.9	7	33.7	37.1	65.4	75.9	2	50.3
Only FCs (Fig. 1c)	**bow, w2v, bert, bi-gru**, clip	17.9	40.8	53.0	8	30.6	33.0	63.2	76.3	3	48.9
	w2v-bert, clip, ATT	20.9	45.6	57.7	7	32.6	35.8	66.3	77.2	3	49.6
	clip, ATT	21.2	46.3	58.2	7	33.1	36.5	66.9	77.7	2	50.2

features independently with FC layers, as in the proposed approach. Regarding the combinations of the textual encoders, we can see that the utilization of fewer but more powerful encoders (i.e. clip and ATT) leads to better results than using a multitude of, possibly weak, encoders as in [20], regardless of the employed visual encoding strategy. These evaluations show that *how* the textual and visual features are combined has significant impact on the obtained results.

In Table 5 we presented the same ablation study for the full and 1k-A variations of the MSR-VTT dataset. In these datasets, we can observe a different behavior concerning the visual encoding: while the utilization of a few and powerful textual encoders (i.e. clip and ATT) continues to perform the best, when it comes to the visual modality, the "*feat. concat. + ATV*" strategy consistently performs the best, regardless of the textual encoder choice. This finding, combined with the results of Table 4, shows that for similar yet different problems and datasets there is no universally-optimal way of combining the visual features (and one can reasonably assume that this may also hold for the textual ones).

5　Conclusions

In this work, we presented a new network architecture for efficient text-to-video retrieval. We experimentally examined different combinations of visual and textual features, and concluded that selectively combining the textual features into fewer but more powerful textual encoders leads to improved results. Moreover, we shown how a fixed set of background queries extracted from large-scale captioning datasets can be used together with softmax operations at the inference stage for revising query-video similarities, leading to improved video retrieval. Extensive experiments and comparisons on different datasets document the value of our approach.

References

1. Awad, G., et al.: Evaluating multiple video understanding and retrieval tasks at trecvid 2021. In: Proceedings of TRECVID 2021. NIST, USA (2021)
2. Bain, M., Nagrani, A., Varol, G., Zisserman, A.: Frozen in time: a joint video and image encoder for end-to-end retrieval. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1728–1738 (2021)
3. Caba Heilbron, F., Escorcia, V., B., G., Niebles, J.C.: Activitynet: a large-scale video benchmark for human activity understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 961–970 (2015)
4. Chen, A., Hu, F., Wang, Z., Zhou, F., Li, X.: What matters for ad-hoc video search? a large-scale evaluation on trecvid. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 2317–2322 (2021)
5. Chen, S., Zhao, Y., Jin, Q., Wu, Q.: Fine-grained video-text retrieval with hierarchical graph reasoning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10638–10647 (2020)
6. Cheng, X., Lin, H., Wu, X., Yang, F., Shen, D.: Improving video-text retrieval by multi-stream corpus alignment and dual softmax loss. arXiv preprint arXiv:2109.04290 (2021)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: arXiv preprint arXiv:1810.04805 (2018)
8. Dong, J., Li, X., Snoek, C.G.M.: Predicting visual features from text for image and video caption retrieval. IEEE Trans. Multimedia **20**(12), 3377–3388 (2018)
9. Dong, J., et al.: Dual encoding for zero-example video retrieval. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9346–9355 (2019)
10. Dong, J., Li, X., Xu, C., Yang, X., Yang, G., Wang, X., Wang, M.: Dual encoding for video retrieval by text. IEEE Trans. Pattern Anal. Mach. Intell. **44**(8), 4065–4080 (2022)
11. Dong, J., Wang, Y., Chen, X., Qu, X., Li, X., He, Y., Wang, X.: Reading-strategy inspired visual representation learning for text-to-video retrieval. IEEE Trans. Circuits Syst. Video Technol. (2022)
12. Faghri, F., Fleet, D.J., Kiros, J.R., Fidler, S.: Vse++: improving visual-semantic embeddings with hard negatives. In: Proceedings of the British Machine Vision Conference (BMVC) (2018)
13. Galanopoulos, D., Mezaris, V.: Hard-negatives or Non-negatives? a hard-negative selection strategy for cross-modal retrieval using the improved marginal ranking loss. In: 2021 IEEE/CVF ICCVW (2021)
14. Galanopoulos, D., Mezaris, V.: Attention mechanisms, signal encodings and fusion strategies for improved ad-hoc video search with dual encoding networks. In: Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR 2020). ACM (2020)
15. Ge, Y., Ge, Y., Liu, X., Li, D., Shan, Y., Qie, X., Luo, P.: Bridging video-text retrieval with multiple choice questions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16167–16176 (2022)
16. Habibian, A., Mensink, T., Snoek, C.G.M.: Video2vec embeddings recognize events when examples are scarce. IEEE Trans. Pattern Anal. Mach. Intell. **39**(10), 2089–2103 (2017)

17. Habibian, A., Mensink, T., Snoek, C.G.: Videostory: a new multimedia embedding for few-example recognition and translation of events. In: Proceedings of the 22nd ACM International Conference on Multimedia, MM 2014, pp. 17–26. ACM, New York (2014)

18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

19. Li, X., Xu, C., Yang, G., Chen, Z., Dong, J.: W2VV++ fully deep learning for ad-hoc video search. In: Proceedings of the 27th ACM International Conference on Multimedia, pp. 1786–1794 (2019)

20. Li, X., Zhou, F., Xu, C., Ji, J., Yang, G.: SEA: sentence encoder assembly for video retrieval by textual queries. IEEE Trans. Multimed. **23**, 4351–4362 (2021)

21. Li, Y., Song, Y., et al.: TGIF: a new dataset and benchmark on animated gif description. In: Proceedings of IEEE CVPR, pp. 4641–4650 (2016)

22. Liu, S., Fan, H., Qian, S., Chen, Y., Ding, W., Wang, Z.: Hit: Hierarchical transformer with momentum contrast for video-text retrieval. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11915–11925 (2021)

23. Liu, Y., Albanie, S., Nagrani, A., Zisserman, A.: Use what you have: Video retrieval using representations from collaborative experts. arXiv preprint arXiv:1907.13487 (2019)

24. Lu, Y.J., Zhang, H., de Boer, M., Ngo, C.W.: Event detection with zero example: Select the right and suppress the wrong concepts. In: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval. p. 127–134. ICMR '16, ACM, New York, NY, USA (2016)

25. Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., Van Der Maaten, L.: Exploring the limits of weakly supervised pretraining. In: Proceedings of the European conference on computer vision (ECCV). pp. 181–196 (2018)

26. Markatopoulou, F., Galanopoulos, D., Mezaris, V., Patras, I.: Query and keyframe representations for ad-hoc video search. In: Proceedings of the 2017 ACM International Conference on Multimedia Retrieval. pp. 407–411. ICMR '17, ACM (2017)

27. 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: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2630–2640 (2019)

28. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: 1st International Conference on Learning Representations, Workshop Track Proceedings. ICLR '13 (2013)

29. Portillo-Quintero, J.A., Ortiz-Bayliss, J.C., Terashima-Marín, H.: A straightforward framework for video retrieval using clip. In: Mexican Conference on Pattern Recognition. pp. 3–12. Springer (2021)

30. 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: Proc. of the 38th Int. Conf. on Machine Learning (ICML) (2021)

31. Song, Y., Soleymani, M.: Polysemous visual-semantic embedding for cross-modal retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1979–1988 (2019)

32. Wang, X., Wu, J., et al.: Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In: Proc. of the IEEE Int. Conf. on Computer Vision. pp. 4581–4591 (2019)

33. Wu, J., Ngo, C.W.: Interpretable embedding for ad-hoc video search. In: Proceedings of the 28th ACM International Conference on Multimedia. p. 3357–3366. ACM, New York, NY, USA (2020)
34. Xu, J., Mei, T., Yao, T., Rui, Y.: MSR-VTT: A large video description dataset for bridging video and language. In: Proc. of IEEE CVPR. pp. 5288–5296 (2016)
35. Yang, X., Dong, J., Cao, Y., Wang, X., Wang, M., Chua, T.S.: Tree-augmented cross-modal encoding for complex-query video retrieval. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp. 1339–1348 (2020)
36. Yu, Y., Kim, J., Kim, G.: A joint sequence fusion model for video question answering and retrieval. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 471–487 (2018)