MuLTI: Efficient Video-and-Language Understanding with Text-Guided MultiWay-Sampler and Multiple Choice Modeling

Jiaqi Xu, Bo Liu, Yunkuo Chen, Mengli Cheng, Xing Shi

Alibaba Group, China
{zhoumo.xjq, xuanyuan.lb, chenyunkuo.cyk, mengli.cml, shubao.sx}@alibaba-inc.com

Abstract

Video-and-language understanding has a variety of applications in the industry, such as video question answering, text-video retrieval, and multi-label classification. Existing video-and-language understanding methods generally adopt heavy multi-modal encoders and feature fusion modules, which consume high computational costs. Specially, they have difficulty dealing with dense video frames or long text prevalent in industrial applications.

This paper proposes MuLTI, a highly accurate and efficient video-and-language understanding model that achieves efficient and effective feature fusion and rapid adaptation to downstream tasks. Specifically, we design a Text-Guided MultiWay-Sampler based on adapt-pooling residual mapping and self-attention modules to sample long sequences and fuse multi-modal features, which reduces the computational costs and addresses performance degradation caused by previous samplers. Therefore, MuLTI can handle longer sequences with limited computational costs. Then, to further enhance the model's performance and fill in the lack of pretraining tasks in the video question answering, we propose a new pretraining task named Multiple Choice Modeling. This task bridges the gap between pretraining and downstream tasks and improves the model's ability to align video and text features. Benefiting from the efficient feature fusion module and the new pretraining task, MuLTI achieves state-of-the-art performance on multiple datasets. Implementation and pretrained models will be released.

Introduction

Video-and-language understanding has a wide range of applications such as *video question answering (videoQA)*, *text-video retrieval* and *multi-label classification* (Diba et al. 2019). Existing methods have made significant progress in video-and-language understanding. However, they still suffer from two challenges: Balancing computational efficiency and performance when dealing with long sequences and the domain gap between pretraining and downstream tasks.

The video-text model generally consists of three modules: text encoder, video encoder, and feature fusion module. The latter two usually cause high computational costs.

Feature fusion modules face efficiency and effectiveness challenges. Previous studies (Fu et al. 2021; Huang et al. 2022) concatenate video-text encoder outputs for transformer encoders processing, with complexity growing with sequence length squared. Other studies (Lei et al. 2021b; Li et al. 2021; Yang et al. 2022; Lei et al. 2021a) reduce computation by condensing video features via mean pooling or class tokens before feature fusion, risking loss of critical details. Flamingo (Alayrac et al. 2022) employs samplers and random queries for efficient video feature condensation, though this approach is suboptimal and may compromise feature integrity. In summary, balancing computational costs and the model's accuracy in the feature fusion module is still challenging. Following (Miech et al. 2019; Sun et al. 2019; Li et al. 2020; Zhu and Yang 2020; Miech et al. 2020), we explore strategies for selectively freezing encoder components to lower visual encoder training costs.

Aligning pretraining with downstream tasks is challenging. Previous pretraining frameworks generally apply four typical pretraining tasks: Masked Frame Modeling (MVM) tasks (Lei et al. 2021a; Ma, Lou, and Ouyang 2021; Fu et al. 2021; Huang et al. 2022) for video encoder optimization, Masked Language Modeling (MLM) tasks (Devlin et al. 2018; Sun et al. 2019; Zhu and Yang 2020; Luo et al. 2020; Li et al. 2020; Lei et al. 2021b) for text encoder optimization, Video Text Matching (VTM) and Video Text Comparison (VTC) tasks (Li et al. 2020; Luo et al. 2020; Fu et al. 2021; Li et al. 2021) for joint optimization of video and text encoders. Although the above methods have proven effective in learning video and text representations, there are still significant domain gaps between pretraining and downstream tasks, especially in videoQA. Only the VTC task is consistent with text-video retrieval among the above pretraining tasks. In summary, narrowing the domain gap between the pretraining and downstream tasks is still challenging.

Addressing these challenges, we introduce MuLTI, featuring a Text-Guided MultiWay-Sampler for sequence condensation and multi-modal fusion. Existing methods typically use a learnable query vector to sample the video feature through self-attention modules (Alayrac et al. 2022). A randomly initialized query vector can discard vital original feature information, causing performance drops. We design an lightweight Adapt-Pooling method in Text-Guided MultiWay-Sampler to obtain the condensed features by calculating the importance of each sequence block. Then, we add the condensed features to the sampled features and use

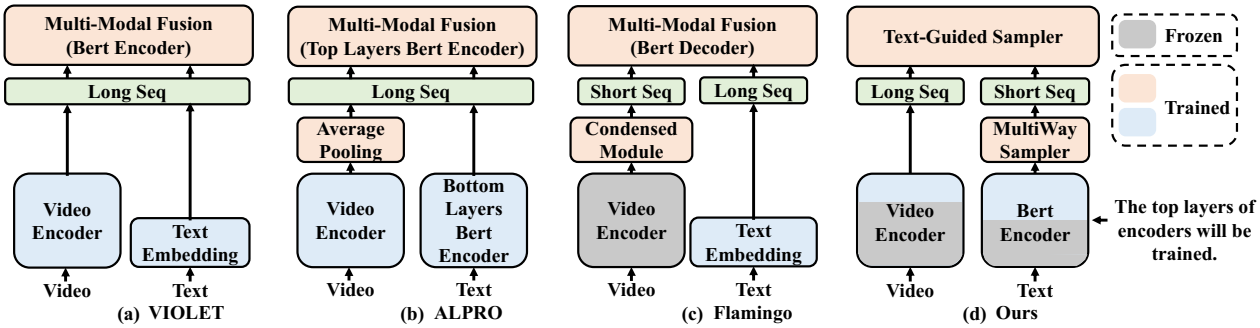

Figure 1: Comparison of different models. Previous works such as (a) and (b) cannot easily handle long sequences. Previous works such as (c) use randomly initialized query vectors for sampler and condense video features, which is sub-optimal solution.

short text features to sample and fuse long video features. We share the self-attention and reserve different feed forward networks for different modalities in the sampler.

Figure 1 shows that previous models (a)(Fu et al. 2021; Huang et al. 2022) and (b)(Li et al. 2021) consume substantial video memory with their lengthy concatenated feature fusion. Both (b) and (c)(Alayrac et al. 2022) compress video features, a common choice due to their greater length compared to text. However, excessive compression can impair performance because of the rich information in video features. In contrast, we design MuLTI like (d) and introduces the Text-Guided MultiWay-Sampler to efficiently condense text features for fusion. Since text is more concise, we use the streamlined text to direct video feature sampling, resulting in enhanced performance.

Pretraining on large-scale video-text datasets could improve the performance of video-text models significantly. However, there are still domain gaps between the existing pretraining tasks and downstream tasks, specifically in videoQA. The difficulty of introducing videoQA to pretraining tasks is constructing suitable question-answer pairs. To reduce the domain gap between the pretraining task and the downstream task in videoQA, we introduce a new pretraining task named Multiple Choice Modeling (MCM). The MCM can bridge the task gap between pretraining and downstream tasks by constructing multiple-choice question answering tasks on large-scale video-text datasets. It asks the model to find text descriptions that match the video most from a randomly constructed collection, which enhances the representation ability of the video and text encoders and the alignment between video and text features.

The contributions can be summarized as follows:

(1) We propose MuLTI, a highly accurate and memory-efficient video-and-language framework, which achieves efficient and effective feature fusion through the feature sampling and attention modules.

(2) We propose a Text-Guided MultiWay-Sampler to sample long sequence features and facilitate the interactions between video and text features, reducing memory cost and improving performance.

(3) We design a new pretraining task called Multiple Choice Modeling (MCM) to bridge the task gap between pretraining and downstream tasks. Experimental results on seven English tasks and one Chinese multi-label classification task demonstrate the effectiveness of MuLTI.

Although we designed MuLTI for industrial scenarios with long sequences, MuLTI still handles short sequences well and achieves state-of-the-art performance.

Related Work

Video-and-Language Structure. Clover (Huang et al. 2022) and VIOLET (Fu et al. 2021) directly concatenate video and text features, using an encoder to manage their complex interactions, with complexity tied to the concatenated sequence length squared. ALPRO (Li et al. 2021) similarly uses an encoder for fusing features but applies mean pooling on video features before concatenation, risking loss of crucial details. AllInOne (Wang et al. 2022a) reduces memory demands by merging text with image features frame-by-frame but still faces high computational loads with extensive OCR transcripts. Flamingo (Alayrac et al. 2022) attempts cost-cutting by condensing video features using samplers and random queries, which isn't ideal. To tackle the above problems, we design a Text-Guided MultiWay-Sampler based on adapt-pooling residual mapping and self-attention modules to sample long sequence features and fuse multi-modal features.

Video-and-Language Pretraining. Four typical pretraining tasks are applied in previous pretraining framework: Masked Frame Modeling (MVM) tasks (Lei et al. 2021a; Ma, Lou, and Ouyang 2021; Fu et al. 2021; Huang et al. 2022), Masked Language Modeling (MLM) tasks (Devlin et al. 2018; Sun et al. 2019; Zhu and Yang 2020; Luo et al. 2020; Li et al. 2020; Lei et al. 2021b; Fu et al. 2021), Video Text Matching (VTM) and Video Text Comparison (VTC) tasks (Li et al. 2020; Luo et al. 2020; Fu et al. 2021; Li et al. 2021). MVM is used for video encoder optimization, MLM is used for text encoder optimization, VTM and VTC are used for joint optimization of video and text encoders. In (Ge et al. 2022), Multiple Choice Questions (MCQ) is proposed to learn fine-grained video and text features. However, MCQ is trained by contrastive loss and does not correlate well with videoQA. In summary, downstream task gaps persist between pretraining and downstream tasks, particularly in videoQA. To address this, we enhance MuLTI with MCM, bridging the pretraining and downstream tasks.

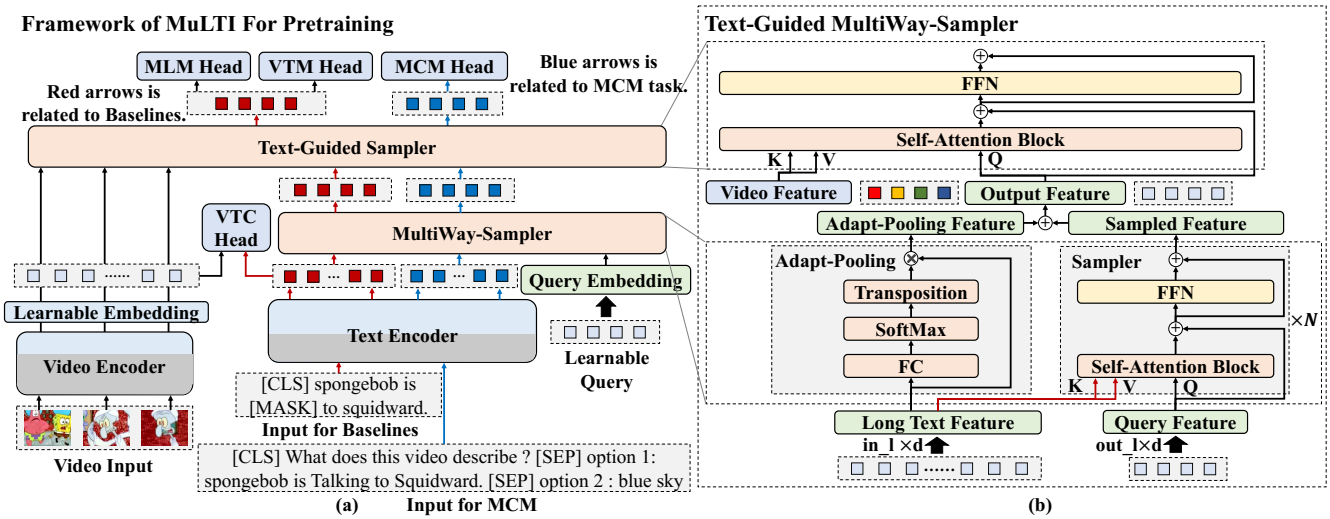

Figure 2: (a) shows the framework of MuLTI. MuLTI contains a video encoder, a text encoder, and a Text-Guided MultiWay-Sampler. Text-Guided MultiWay-Sampler is used to condense the extracted features and feature fusion. (b) shows the framework of the Text-Guided MultiWay-Sampler. The adapt-pooling feature provides origin information. We share the self-attention module and reserve different feed forward networks for different modalities in the sampler to accommodate modalities.

Methodology

MuLTI's Architecture

Figure 2 (a) gives an overview of MuLTI's architecture. Details for each component are as follows.

Video and Text Encoders: Unless specified, a 12-layer VIT-B/16_{224} (Radford et al. 2021) is used as video encoder. We sparsely sample N_v frames from the input video. The VIT-B/16_{224} model divides each frame into K non-overlapping patches. The per-video features is $\tilde{\vec{v}} \in \mathbb{R}^{N_v \times K \times d}$, where d is the feature dimension. The output of the video encoder is a video features sequence: $\{\vec{v}_1, ..., \vec{v}_{N_v}\}$, with $\vec{v}_i \in \mathbb{R}^{K \times d}$. Experiments revealed the class token is unnecessary and was removed to save computation. Unless specified, a 12-layer bert (Devlin et al. 2018) is used as the text encoder. Assuming the length of input text is N_t, the output of the text encoder is a text features sequence $\tilde{\vec{t}} \in \mathbb{R}^{N_t \times d}$: $\{\vec{t}_{\text{cls}}, \vec{t}_1, ..., \vec{t}_{N_t}\}$, with $\vec{t}_i \in \mathbb{R}^d$. The $\vec{t}_{\text{cls}}$ is the output of the text `[CLS]` token. Following (Miech et al. 2019; Sun et al. 2019; Li et al. 2020), we explore training strategies for partially freezing encoder layers.

Text-Guided MultiWay-Sampler: The multi-modal fusion core is the Text-Guided MultiWay-Sampler, adapted from the transformer decoder, shown in Figure 2 (b). The Text-Guided MultiWay-Sampler is designed to condense text features and fuse different modal features efficiently. Following (Alayrac et al. 2022), we initialize a random learnable query to condense features via sampling. The expression $Sampler(z, q)$ represents the sampling of feature z using the query vector q through the sampler.

(i) **Why we need Adapt-Pooling?** Learnable queries can compress features well, but starting with random vectors may reduce their effectiveness. Random initialization may lose key details in original features, weakening the model's ability to capture and retain the essence of the data. Therefore, we design an attention-based lightweight Adapt-Pooling method to condense long sequence features. The Adapt-Pooling structure is shown on the left side of the Figure 2 (b). The formula is shown below, $AdaPool(z)$ is the output of the Adapt-Pooling, with $W^{reduce} \in \mathbb{R}^{d \times N_s}$, d the hidden dimension of the transformer, N_s the length of condensed features, $*.T$ the transposition of the matrix.

$$AdaPool(z) = Softmax((W^{reduce} z).T) \times z \quad (1)$$

The $Softmax((W^{reduce} z).T)$ yields an importance weight matrix of shape $[N_s, N_i]$, with each element signifying the relative importance of the corresponding block within the sequence, and N_i representing the length of the input features. Adapt-Pooling selectively highlights key input segments, condensing features while preserving its critical attributes. This integration enriches the feature set with distilled information and ensures full data utilization, boosting the model's capacity and robustness.

(ii) **Why we condense text features?** The video features are often redundant, whereas text features are denser and more meaningful (He et al. 2022). Language guidance is key to distilling valuable information from videos. Both (Li et al. 2021) and (Alayrac et al. 2022) condense the video features. Excessive compression harms model performance; using condensed text to sample video features improves results. Before fusion, learnable time embeddings enhance image features for temporal modeling. The short text features are used to sample the video features to fuse multi-modal features. In our Text-Guided MultiWay-Sampler, we use shared self-attention modules but distinct FFNs for each modality to handle multi-modal features efficiently. The fuse feature is shown as follow, with q the query embedding of

Method	FLOPs	Params	FPS
MuLTI-S	99G	203M	20.74
MuLTI-B	346G	247M	10.13
VIOLET (Fu et al. 2021)	249G	198M	9.05
ALPRO (Li et al. 2021)	432G	235M	9.97
MuLTI-L	1509G	746M	3.12
FrozenBiLM (Yang et al. 2022)	1733G	1224M	2.54

Table 1: Comparison among models with 16 frames. Text length is 512. FPS is based on 1 NVIDIA V100 16GB GPU.

text features, z_{out} the fused feature:

$$z_{out} = Sampler(\tilde{\vec{v}}, Sampler(\tilde{\vec{t}}, q) + AdaPool(\tilde{\vec{t}})) \quad (2)$$

A work similar to ours is Token Learner (Ryoo et al. 2021), which uses spatial attention in model to extract 8 or 16 representative vectors from an image. The difference is that we use Adapt-Pooling and self-attention to condense features for multi-modal fusion. The sampler extracts complex information via self-attention, while Adapt-Pooling provides fast, simple features through residual mapping.

(iii) **Why Text-Guided MultiWay-Sampler is efficient ?** Our feature fusion module outperforms flatten-based methods and transformer encoders in efficiency, as simple analysis shows: we assume VIT-B/16_{224} is used as video encoder, each frame will be flattened into a sequence of length 196. Let the number of queries be N_t^q for text, the length of the video features be $N_v \times 196$, and the length of the text features be N_t. Thus the complexity of the flatten method will be $O((N_t + N_v \times 196)^2)$. After applying the Text-Guided MultiWay-Sampler, the complexity is $O(N_t^q \times N_v \times 196 + N_t^q \times N_t)$. As N_t^q are generally much smaller than N_v and N_t. our method is much more efficient than other methods.

MuLTI for different scenes. In this section, we built scalable models for resource-varied scenes. We replace the video encoder from VIT-B/16 to VIT-L/14 and the text encoder from bert-base to bert-large. Then, we obtain MuLTI-L. In addition, we replace the video encoder from VIT-B/16 to VIT-B/32 and reduce the text encoder from 12 layers to 6 layers. Then, we obtain MuLTI-S. The floating point of operations (FLOPs), parameters (Params) and frames per second (FPS) of different models are shown in Table 1.

Pretraining for MuLTI

Multiple Choice Modeling: Despite MLM and VTM's success in learning video-text representations, a significant gap remains between pretraining and downstream tasks like videoQA. The difficulty of introducing videoQA into the pretraining task is constructing suitable question-answer pairs. Inspired by multiple choice videoQA, we find the text descriptions paired with videos are the correct natural answers. Therefore, we introduce Multiple Choice Modeling, a new pretraining task that bridges the task gap between pretraining and downstream tasks. Specifically, it is constructed as follows, which is a four-choice question.

"[CLS]<Question> ? [SEP] Option 1: <Answer 1>. [SEP] Option 2: <Answer 2>. [SEP] Option 3: <Answer 3>. [SEP] Option 4: <Answer 4>."

We randomly place the correct descriptions in <Answer 1>, <Answer 2>, <Answer 3>, <Answer 4>, and obtain answers other than the correct descriptions through the text corpus. The <Question> also has various choices, such as "What does this picture describe?", "What does this video describe?" and so on.

As shown in Figure 2 (a), typical MLM, VTM, and VTC tasks correspond to the red arrows and red squares in the image. The MCM corresponds to the image's blue arrows and blue squares, and the MCM does not conflict with the other pretraining tasks. The MCM is seamlessly integrated with other pretraining tasks and does not require additional manual annotations or data preprocessing. It utilizes video encoders to extract visual features and text encoders for generating textual representations, followed by a Text-Guided MultiWay-Sampler for feature fusion. The MCM head evaluates the given options' relevance to the video, optimizing alignment using cross-entropy loss. The MCM task, choosing the best description from options, mirrors essential videoQA cognition, enhancing the model's cross-modal reasoning and alignment. MCM directly improves the model's ability to match text with corresponding videos, enhancing performance in text-video retrieval tasks.

Pretraining Objectives: We also employ the MLM, VTM and VTC, considering their effectiveness. The MLM randomly masks input tokens with 15% probability and replaces them with [MASK], which are predicted based on video and text. The VTC treats matching video text pairs as positive pairs and other video text pairs in the batch as negative pairs. The VTM is slightly different from VTC, where the multi-modal features are fused before used for classification. The overall pretraining objective of MuLTI is:

$$L = L_{mlm} + L_{vtc} + L_{vtm} + L_{mcm} \quad (3)$$

Experiments

Implementation Details

Pretraining datasets: We pretrained the model using two large datasets. One is WebVid-2M (Bain et al. 2021), which contains 2.5M video-text pairs. Because pretraining the video-text model using image-text pairs also improves the model's performance (Lei et al. 2021b), the CC-3M(Sharma et al. 2018) is also used as a pretrained dataset containing 3M image-text pairs.

We implement MuLTI in PyTorch (Paszke et al. 2019). In detail, the video encoder is initialized with pretrained weights from CLIP (Radford et al. 2021). Text encoder is initialized with a 12-layer of the $\text{BERT}_{\text{base}}$ model (Devlin et al. 2018). Then, a 4-layer Text-Guided MultiWay-Sampler is used to condense text features and fuse multi-modal features. The length of query embedding is set to 16. MuLTI pretraining spanned 10 epochs on eight NVIDIA A100 GPUs, a 256 batch size totaling 200k iterations. Optimization used AdamW with a $1e^{-4}$ learning rate, 0.05 weight decay, and a warm-up scheduler. We uniformly sample 16 frames for each video and scale them to 224×224.

Method	#PT	MSRQ Acc.↑	MSVQ Acc.↑	TGIF. Act. Acc.↑	TGIF. Tran. Acc.↑	TGIF Fra. Acc.↑	MSRVTT Ret R1 / R5 / R10 ↑	MSRVTT Ret G-M ↑	DiDeMo Ret R1 / R5 / R10 ↑	DiDeMo Ret G-M ↑
CLIP4CLIP	400M	-	-	-	-	-	43.1 / 70.4 / 80.8	62.6	43.4 / 70.2 / 80.6	62.6
QB-Norm	400M	-	-	-	-	-	47.2 / 73.0 / 83.0*	65.9*	43.3 / 71.4 / 80.8*	63.0*
CAMoE	400M	-	-	-	-	-	47.3 / 74.2 / 84.5*	66.7*	43.8 / 71.4 / 79.9*	63.0*
TS2-Net	400M	-	-	-	-	-	54.0 / 79.3 / 87.4*	72.1*	47.4 / 74.1 / 82.4*	66.1*
ALPRO	5.5M	42.1	45.9	-	-	-	33.9 / 60.7 / 73.2	53.2	35.9 / 67.5 / 78.8	57.6
VIOLET	185.5M	43.9	47.9	92.5	95.7	68.9	34.5 / 63.0 / 73.4	54.2	32.6 / 62.8 / 74.7	53.5
AllInOne	102.5M	44.3	47.9	92.7	94.3	64.2	37.9 / 68.1 / 77.1	58.4	32.7 / 61.4 / 73.5	52.8
Clover	5.5M	44.1	52.4	95.0	98.2	71.6	40.5 / 69.8 / 79.4	60.7	50.1 / 76.7 / 85.6	69.0
Flamingo	2139M	47.4	52.3	-	-	-	-	-	-	-
FrozenBiLM	10M	47.0	54.4	-	-	68.6	-	-	-	-
MuLTI-S	5.5M	45.6	50.0	97.3	98.9	71.2	41.3 / 70.6 / 79.7 45.8 / 73.5 / 82.0*	61.5 65.1*	42.6 / 71.4 / 80.0 47.9 / 73.0 / 82.6*	62.5 66.1*
MuLTI-B	5.5M	46.6	53.0	97.3	**99.1**	73.5	45.1 / 72.4 / 81.8 49.4 / 75.9 / 84.0*	64.4 68.0*	45.2 / 74.6 / 82.2 48.3 / 75.4 / 83.5*	65.2 67.2*
MuLTI-L	5.5M	**47.8**	**54.7**	**97.9**	99.0	**75.6**	48.7 / 75.9 / 83.9 **54.7 / 77.7 / 86.0***	67.7 **71.5***	50.5 / 78.5 / 86.2 **56.5 / 80.2 / 87.0***	69.9 **73.3***

Table 2: Comparisons with existing methods. #PT means number of pretrain datasets. Acc. (%) denotes the performance of videoQA. R@k denotes recall (%) with k retrieval efforts. G-M denotes the geometric mean of R@1, R@5, R@10. The datasets commonly used are WebVid2M (Bain et al. 2021), WebVid10M (Bain et al. 2021), WIT (Radford et al. 2021), HowTo100M (Miech et al. 2019), YT-Temporal-180M (Zellers et al. 2021), Conceptual Captions (Sharma et al. 2018). * indicates that DSL (Cheng et al. 2021) or QB-Norm (Bogolin et al. 2021) is used for post-processing.

Method	#PT	OCR	Multi-Label
VIOLET ‡	-	✘	55.22
ALPRO ‡	-	✘	58.53
MuLTI-S	-	✘	63.97
MuLTI-S	-	✔	**66.13**
MuLTI-B	-	✘	64.60
MuLTI-B	-	✔	**67.86**

Table 3: Comparisons on multi-label classification in mAP (%). ‡ means the methods are reproduced in our framework.

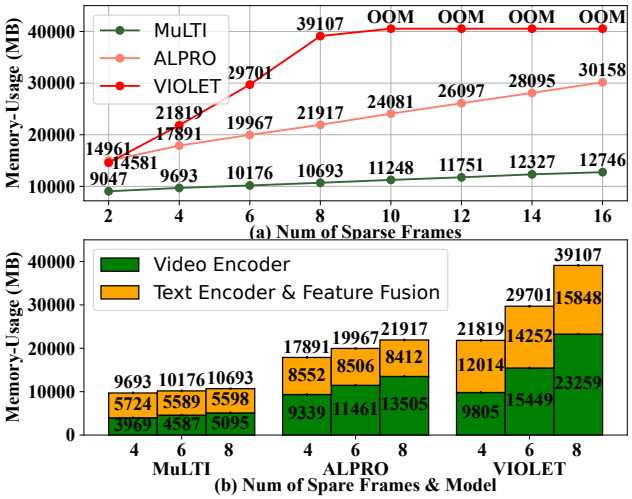

Figure 3: Comparisons with existing methods on Memory-Usage with different numbers of frames. Text length is 512.

Downstream Tasks and Datasets

Video Question Answering. We evaluate MuLTI on five widely used videoQA tasks. **(1) MSRQ (MSRVTT-QA)** (Xu et al. 2017, 2016) is a open-ended videoQA task includes 10k videos and 243k question-answer pairs. **(2) MSVQ (MSVD-QA)** (Xu et al. 2017; Chen and Dolan 2011) is a open-ended videoQA task includes 1970 videos and 50k question-answer pairs. **(3) TGIF-QA (Jang et al. 2017)** contains three datasets: TGIF-Action and TGIF-Transition for multiple-choice videoQA tasks, and TGIF-Frame for open-ended videoQA tasks.

Text-Video Retrieval. (1) MSRR (MSRVTT-Ret) contains 10K videos with 200K annotations. Following (Fu et al. 2021), we use 9k videos for training and 1k videos for testing. **(2) DiDeMo (DiDeMo-Ret)** consists of 10K videos with 40K annotations. Following (Lei et al. 2021b), we concatenate all annotations from the video into a title.

Multi-Label Classification. Video labels are crucial for the ranking models used in online advertising[1]. We create a short video dataset from our app, which includes 486k videos with captions and 21696 labels. Multiple professional editors cross check the labels. We used a high-performing text detector from ICDAR[2] for OCR transcripts. The OCR transcripts are truncated to 512. The examples for multi-label classification can be found in the appendix.

[1]https://algo.qq.com/index.html

[2]https://rrc.cvc.uab.es/?ch=4&com=evaluation&task=4

Method	Base	TGMS	PB	MCM	MSRQ Acc.↑	MSVQ Acc.↑	MSRR R1↑	MSRR R5↑	MSRR R10↑	MSRR G-Mean ↑
MuLTI-B	✔	✘	✘	✘	44.84	48.35	38.90	69.50	78.50	59.64
	✔	✔	✘	✘	45.54	49.86	38.80	70.30	80.10	60.22
	✔	✔	✔	✘	46.28	51.93	44.30	72.40	**81.90**	64.04
	✔	✔	✔	✔	**46.61**	**53.03**	**45.10**	**72.40**	81.80	**64.40**

Table 4: Evaluations of the proposed methods. TGMS: Text-Guided MultiWay-Sampler. PB (Pretraining Baseline): Pretraining model with MLM, VTM and VTC. MCM: Multiple Choice Modeling. Acc. (%) is used to measure the performance of videoQA. R@k denotes recall (%) with k retrieval efforts. G-Mean denotes the geometric mean of R@1, R@5, R@10.

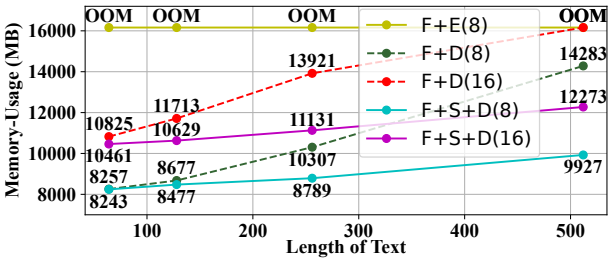

Figure 4: Comparisons of different text length and number of frames on Memory-Usage. The F means Flatten, the D means Decoder, the E means Encoder, the S means Sampler. The number in parentheses represents the number of frames.

Figure 5: A visualization of the cross-attention map from the Text-Guided MultiWay-Sampler.

Methods	MSRQ	MSVQ	Memory Usage
Class Token	44.54	47.90	7081
Mean Pooling	44.40	47.07	6941
Max Pooling	44.41	46.93	6963
Flatten + Encoder	44.84	48.35	15791
TGMS	**45.54**	**49.86**	10551

Table 5: Ablation studies on feature retention methods. The number of sparse frames is set to 6 for Flatten method. TGMS means Text-Guided MultiWay-Sample.

Method	CV	CT	SS	AP	MSRQ	MSVQ
Flatten Decoder	✘	✘	✘	✘	45.13	49.19
	✔	✘	✘	✘	44.76	48.10
	✔	✘	✔	✔	45.14	48.92
	✔	✔	✘	✘	44.57	48.50
	✘	✔	✘	✘	45.08	49.38
	✘	✔	✔	✘	45.16	49.80
	✘	✔	✘	✔	45.48	49.54
	✘	✔	✔	✔	**45.54**	**49.86**

Table 6: An ablation study on feature compression methods. CV means Condensed Video, CT means Condensed Text, SS means Shared-Sampler, AP means Adapt-Pooling.

Performance of Proposed Methods

Table 2 compares MuLTI with CLIP4CLIP (Luo et al. 2021), QB-Norm (Bogolin et al. 2021), CAMoE (Cheng et al. 2021), TS2-Net (Liu et al. 2022), ALPRO (Li et al. 2021), VIOLET (Fu et al. 2021), AllInOne (Wang et al. 2022a), Clover (Huang et al. 2022), Flamingo (Alayrac et al. 2022) and FrozenBiLM (Yang et al. 2022).

In videoQA tasks, MuLTI surpasses all baseline models on MSRQ, MSVQ, TGIF-Action, TGIF-Transition and TGIF-Frames. Since MuTLI does not use speech data as input, it is compared with FrozenBiLM(Yang et al. 2022) without using speech data. In general, MuLTI achieves state-of-the-art performance in various QA tasks.

In text-video retrieval tasks, we finetune MuLTI using the MSRVTT and DiDeMo datasets. Our results demonstrate that MuLTI is highly competitive in both benchmarks, particularly in the DiDeMo dataset. These findings highlight the effectiveness of MuLTI for text-video retrieval.

For multi-label classification, we compare MuLTI with VIOLET and ALPRO but exclude FrozenBiLM due to its impractical size for industry deployment. VIOLET and ALPRO do not use OCR transcripts as they would lead to out-of-memory on V100 GPUs. We also report MuLTI's OCR-less performance in Table 3 for a fair comparison; MuLTI significantly surpasses both VIOLET and ALPRO. As shown in Figure 3, MuLTI maintains a video memory cost less than half of ALPRO's and VIOLET's when frame count rises during training, because its efficient fusion modules minimizes memory cost increases.

Finally, we evaluate our main technical contributions in Table 4. Compared with baseline models, our main technical contributions improve performance on all datasets. The Text-Guided MultiWay-Sampler boosts MuLTI's multi-modal fusion ability, pinpointing key details in surplus video features. MCM advances the model's alignment ability and narrows the gap between pretraining and downstream tasks.

PB	MVM	MCM	MSRQ	MSVQ	MSRR
✔	✘	✘	46.28	51.93	64.04
✔	✔	✘	45.87	50.16	63.41
✔	✔	✔	46.11	51.65	63.71
✔	✘	✔	**46.61**	**53.03**	**64.40**

Table 7: Ablation studies on the Multiple Choice Modeling. PB means Pretraining Baseline.

Frozen / Total		MSRQ	MSVQ	Memory Usage
VE	TE			
12/12	12/12	44.06	46.83	6109
12/12	0/12	44.07	47.12	7439
6/12	0/12	45.10	47.57	18219
9/12	0/12	45.59	47.52	11541
9/12	3/12	45.50	49.63	11131
9/12	6/12	**45.54**	**49.86**	10551
9/12	9/12	45.04	49.14	10283

Table 8: Ablation studies on frozen layers. VE refers to video encoder and TE refers to text encoder. Frozen/Total refers to the number of frozen and total layers.

Adapter	ATT	MSRQ	MSVQ	MSRR	DiDeMo
✘	✘	45.54	49.86	60.22	51.68
✔	✘	45.61	50.48	60.54	52.08
✔	✔	**45.71**	**50.63**	**61.16**	**52.42**

Table 9: Ablation studies on the Attention-Adapter. ATT means Attention.

The Importance of Text-Guided MultiWay-Sampler

Why we condense text features? We compare performance of different aggregation methods (*i.e.* Class Token, Mean Pooling, Max Pooling and Flatten) in Table 5. Results show that Flatten outperforms other aggregation methods but requires substantial video memory. Above section reveals the decoder uses less memory than the encoder for long sequences, prompting its use in feature fusion. The decoder handles datasets like MSRQ well. However, the cost is still high when processing long text and video like our multi-label datasets. The specific memory cost is shown in Figure 4. Following (Alayrac et al. 2022), we use a decoder-based sampler for feature condensation Table 6 compares different condensation methods, showing text compression's superiority. As shown in Figure 5, the visual part most relevant to the problem is given more weight.

The importance of Shared-Sampler. The sampler and feature fusion module, using the same decoder structure, can share weights without compromising performance, simplifying model optimization (Wang et al. 2022b). We share the sampler and decoder's self-attention but keep separate FFNs for each modality, cutting parameters while maintaining performance. Compared with the Flatten Method, the Shared-Sampler improves accuracies on MSRQ and MSVQ by 0.32% and 1.45%, respectively.

The importance of Adapt-Pooling. As shown in Table 6, the sampler leads to worse performance when condensing text and video features. The sampler's random query vector carries the risk of losing original key features; we design a lightweight aggregation module, Adapt Pooling, to preserve the original features. As shown in Table 6, the Adapt-Pooling improves accuracy on MSRQ and MSVQ. Additionally, we explored various combination methods (*i.e.* add, concatenate, and multiply), and noted slight performance differences. We achieved an accuracy of 45.51% using concatenate and 45.45% using multiply on MSRQ.

To verify these techniques' robustness, we applied them to condense video features, which also improved performance.

The Importance of Multiple Choice Modeling

MCM aims to bridge the gap between pretraining and downstream tasks by integrating videoQA into pretraining, enhancing the model's focus on video and sentence subjects for better multimodal feature extraction.

We use the classical MLM, VTM, and VTC tasks to pretrain the model as a baseline. Due to video content corruption caused by MVM, the MVM task conflicts with other tasks (Lei et al. 2021a). In our initial attempts to include MVM for pretraining, we observed a degradation in performance as shown in Table 7. Thus, we have decided not to use MVM for pretraining. To confirm MCM's robustness, we also added MCM for pretraining based on the usage of MVM. The results show MCM still substantially enhances model's performance. Compared to the model pretrained with baseline, MCM explicitly improves the model's performance on the videoQA task by narrowing the task gap between pretraining and downstream tasks. MCM's promotion of multi-modal feature alignment enhances the model's retrieval task performance. As shown in Table 7, the models pretrained with MCM outperformed the baseline in both videoQA and retrieval tasks, demonstrating its effectiveness.

Ablation Experiment on Training Strategies

Analysis of Frozen Layers. In this section, we systematically evaluate the effect of the number of frozen layers. The results on videoQA are demonstrated in Table 8. It indicates that unfreezing the top layers of video and text encoders can improve performance on both datasets.

Analysis of Attention-Adapter. Analyzing frozen layers reveals that unfreezing excess layers reduces accuracy due to overfitting from excessive parameter adjustments. Following (Yang et al. 2022), we add adapters to the encoders in the shallow layers. Table 9 shows that while adapters perform effectively, their capability is constrained by the basic FFN module. By integrating a lightweight attention module (Hu et al. 2017), the model focuses better on informative tokens.

Conclusion

We present MuLTI, a high-performing video-language framework with a novel Text-Guided MultiWay-Sampler for improved sampling efficiency and a pretraining task to better align with downstream tasks. MuLTI achieves state-of-the-art performance on seven video-language benchmarks.

References

Alayrac, J.-B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; Ring, R.; Rutherford, E.; Cabi, S.; Han, T.; Gong, Z.; Samangooei, S.; Monteiro, M.; Menick, J.; Borgeaud, S.; Brock, A.; Nematzadeh, A.; Sharifzadeh, S.; Binkowski, M.; Barreira, R.; Vinyals, O.; Zisserman, A.; and Simonyan, K. 2022. Flamingo: a Visual Language Model for Few-Shot Learning. *ArXiv*, abs/2204.14198.

Bain, M.; Nagrani, A.; Varol, G.; and Zisserman, A. 2021. Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1708–1718.

Bogolin, S.-V.; Croitoru, I.; Jin, H.; Liu, Y.; and Albanie, S. 2021. Cross Modal Retrieval with Querybank Normalisation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5184–5195.

Chen, D.; and Dolan, W. B. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, 190–200.

Cheng, X.; Lin, H.; Wu, X.; Yang, F.; and Shen, D. 2021. Improving Video-Text Retrieval by Multi-Stream Corpus Alignment and Dual Softmax Loss. *ArXiv*, abs/2109.04290.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805.

Diba, A.; Fayyaz, M.; Sharma, V.; Paluri, M.; Gall, J.; Stiefelhagen, R.; and Gool, L. V. 2019. Large Scale Holistic Video Understanding. In *European Conference on Computer Vision*.

Fu, T.-J.; Li, L.; Gan, Z.; Lin, K.; Wang, W. Y.; Wang, L.; and Liu, Z. 2021. VIOLET : End-to-End Video-Language Transformers with Masked Visual-token Modeling. *ArXiv*, abs/2111.12681.

Ge, Y.; Ge, Y.; Liu, X.; Li, D.; Shan, Y.; Qie, X.; and Luo, P. 2022. BridgeFormer: Bridging Video-text Retrieval with Multiple Choice Questions. *ArXiv*, abs/2201.04850.

He, K.; Chen, X.; Xie, S.; Li, Y.; Doll'ar, P.; and Girshick, R. B. 2022. Masked Autoencoders Are Scalable Vision Learners. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 15979–15988.

Hu, J.; Shen, L.; Albanie, S.; Sun, G.; and Wu, E. 2017. Squeeze-and-Excitation Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42: 2011–2023.

Huang, J.; Li, Y.; Feng, J.; Sun, X.; and Ji, R. 2022. Clover: Towards A Unified Video-Language Alignment and Fusion Model. *ArXiv*, abs/2207.07885.

Jang, Y.; Song, Y.; Yu, Y.; Kim, Y.; and Kim, G. 2017. TGIF-QA: Toward Spatio-Temporal Reasoning in Visual Question Answering. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1359–1367.

Lei, C.; Luo, S.; Liu, Y.; He, W.; Wang, J.; Wang, G.; Tang, H.; Miao, C.; and Li, H. 2021a. Understanding Chinese Video and Language via Contrastive Multimodal Pre-Training. *Proceedings of the 29th ACM International Conference on Multimedia*.

Lei, J.; Li, L.; Zhou, L.; Gan, Z.; Berg, T. L.; Bansal, M.; and Liu, J. 2021b. Less is more: Clipbert for video-and-language learning via sparse sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7331–7341.

Li, D.; Li, J.; Li, H.; Niebles, J. C.; and Hoi, S. C. H. 2021. Align and Prompt: Video-and-Language Pre-training with Entity Prompts. *ArXiv*, abs/2112.09583.

Li, L.; Chen, Y.-C.; Cheng, Y.; Gan, Z.; Yu, L.; and Liu, J. 2020. Hero: Hierarchical Encoder for Video+Language Omni-representation Pre-training. *ArXiv*, abs/2005.00200.

Liu, Y.; Xiong, P.; Xu, L.; Cao, S.; and Jin, Q. 2022. TS2-Net: Token Shift and Selection Transformer for Text-Video Retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Luo, H.; Ji, L.; Shi, B.; Huang, H.; Duan, N.; Li, T.; Li, J.; Bharti, T.; and Zhou, M. 2020. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*.

Luo, H.; Ji, L.; Zhong, M.; Chen, Y.; Lei, W.; Duan, N.; and Li, T. 2021. CLIP4Clip: An Empirical Study of CLIP for End to End Video Clip Retrieval. *Neurocomputing*, 508: 293–304.

Ma, Z.; Lou, M.; and Ouyang, X. 2021. Top1 Solution of QQ Browser 2021 Ai Algorithm Competition Track 1 : Multimodal Video Similarity. *ArXiv*, abs/2111.01677.

Miech, A.; Alayrac, J.-B.; Smaira, L.; Laptev, I.; Sivic, J.; and Zisserman, A. 2020. End-to-End Learning of Visual Representations From Uncurated Instructional Videos. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9876–9886.

Miech, A.; Zhukov, D.; Alayrac, J.-B.; Tapaswi, M.; Laptev, I.; and Sivic, J. 2019. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2630–2640.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32: 8026–8037.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; Krueger, G.; and Sutskever, I. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*.

Ryoo, M. S.; Piergiovanni, A. J.; Arnab, A.; Dehghani, M.; and Angelova, A. 2021. TokenLearner: What Can 8 Learned Tokens Do for Images and Videos? *ArXiv*, abs/2106.11297.

Sharma, P.; Ding, N.; Goodman, S.; and Soricut, R. 2018. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning. In *ACL*.

Sun, C.; Myers, A.; Vondrick, C.; Murphy, K. P.; and Schmid, C. 2019. VideoBERT: A Joint Model for Video and Language Representation Learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 7463–7472.

Wang, A.; Ge, Y.; Yan, R.; Ge, Y.; Lin, X.; Cai, G.; Wu, J.; Shan, Y.; Qie, X.; and Shou, M. Z. 2022a. All in One: Exploring Unified Video-Language Pre-training. *ArXiv*, abs/2203.07303.

Wang, W.; Bao, H.; Dong, L.; Bjorck, J.; Peng, Z.; qiang liu; Aggarwal, K.; Mohammed, O. K.; Singhal, S.; Som, S.; and Wei, F. 2022b. Image as a Foreign Language: BEiT Pretraining for All Vision and Vision-Language Tasks. *ArXiv*, abs/2208.10442.

Xu, D.; Zhao, Z.; Xiao, J.; Wu, F.; Zhang, H.; He, X.; and Zhuang, Y. 2017. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the ACM international conference on Multimedia*, 1645–1653.

Xu, J.; Mei, T.; Yao, T.; and Rui, Y. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5288–5296.

Yang, A.; Miech, A.; Sivic, J.; Laptev, I.; and Schmid, C. 2022. Zero-Shot Video Question Answering via Frozen Bidirectional Language Models. *ArXiv*, abs/2206.08155.

Zellers, R.; Lu, X.; Hessel, J.; Yu, Y.; Park, J. S.; Cao, J.; Farhadi, A.; and Choi, Y. 2021. MERLOT: Multimodal Neural Script Knowledge Models. In *Neural Information Processing Systems*.

Zhu, L.; and Yang, Y. 2020. ActBERT: Learning Global-Local Video-Text Representations. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8743–8752.