MV-Adapter: Multimodal Video Transfer Learning for Video Text Retrieval

Xiaojie Jin[1*⊠], Bowen Zhang[1*], Weibo Gong[1], Kai Xu[1], Xueqing Deng[1],
Peng Wang[1], Zhao Zhang[2], Xiaohui Shen[1], Jiashi Feng[1]

[1]Bytedance Inc., [2]Hefei University of Technology

{jinxiaojie,zhangbowen.17,gongweibo,xukai.1993,xueqingdeng}@bytedance.com

{peng.wang,shenxiaohui.kevin,jshfeng}@bytedance.com cszzhang@gmail.com

Abstract

*State-of-the-art video-text retrieval (VTR) methods typically involve fully fine-tuning a pre-trained model (e.g. CLIP) on specific datasets. However, this can result in significant storage costs in practical applications as a separate model per task must be stored. To address this issue, we present our pioneering work that enables parameter-efficient VTR using a pre-trained model, with only a small number of tunable parameters during training. Towards this goal, we propose a new method dubbed **M**ultimodal **V**ideo **A**dapter (MV-Adapter) for efficiently transferring the knowledge in the pre-trained CLIP from image-text to video-text. Specifically, MV-Adapter utilizes bottleneck structures in both video and text branches, along with two novel components. The first is a* Temporal Adaptation Module *that is incorporated in the video branch to introduce global and local temporal contexts. We also train weights calibrations to adjust to dynamic variations across frames. The second is* Cross Modality Tying *that generates weights for video/text branches through sharing cross modality factors, for better aligning between modalities. Thanks to above innovations, MV-Adapter can achieve comparable or better performance than standard full fine-tuning with negligible parameters overhead. Notably, MV-Adapter consistently outperforms various competing methods in V2T/T2V tasks with large margins on five widely used VTR benchmarks (MSR-VTT, MSVD, LSMDC, DiDemo, and ActivityNet). Codes will be released.*

1. Introduction

Video text retrieval (VTR) [9, 10, 13, 18, 26, 27, 31, 32, 34, 37, 44, 47], aiming to obtain the rankings of videos/texts in a repository given text/video queries (i.e. T2V and V2T respectively) has a wide range of practical applications. Recently, with the surge of large-scale pre-trained image-text
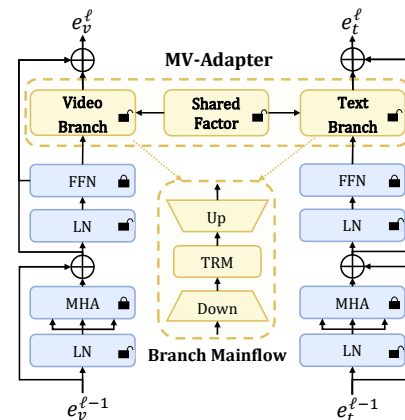

Figure 1. The overall pipeline of MV-adapter with the illustration of the basic structure of video/text branches. Only a small part of model is tunable during training, highlighted by the "unlock" symbol.

models, particularly CLIP [38], transferring the knowledge learned in CLIP to VTR tasks by fully fine-tuning has become the de-facto paradigm and adopted by state-of-the-art methods [10, 31, 32, 34, 37, 44].

However, these methods suffer from substantial storage overhead in practical applications as each task necessitates the storage of a distinct model. This issue becomes more severe when the pre-trained model size increases and multiple VTR tasks need to be solved, thus hindering the application in real-world scenarios. For instance, in mobile apps, model size is restricted to reduce downloading time and/or app package size, as a larger size may decrease active users. In cloud services, many models need to be trained to solve functionally similar but customized tasks, making standard fine-tuning unfavorable in such cases. Additionally, since most VTR tasks have relatively small training sets, performing full fine-tuning on these datasets results in instability and poor performance, as demonstrated in [6, 36].

To solve this problem, we introduce a new task to perform Parameter Efficient transfer learning of VTR (PE-VTR), i.e. only a small number of parameters are tunable during training while the majority weights are frozen. This results in high degree of parameter sharing between mod-

---

*Equal contribution. Bowen Zhang did the work during an internship.
⊠Corresponding author: Xiaojie Jin.

els, with each model requiring only a small number of additional parameters for a new task. Generally speaking, there are two challenges in tackling PE-VTR: (i) adapt image-text pre-trained models to video-text and sufficiently learn the temporal context and cross-modal correlations. (ii) ensure that the model is parameter-efficient with negligible parameter overhead while maintaining performance. Initially, we revisit current methods [31, 34] that perform full fine-tuning by adapting them to PE-VTR and freezing their CLIP backbone. However, to our surprise, they all fall far behind the full fine-tuning counterparts in terms of performance (*cf*. Sec. 5.2). In addition to inferior performance, these methods also have other limitations that make them unsuitable for PE-VTR. Some [4, 14, 22] are designed only for a single modality (image or text) and ignore the temporal modeling and/or the interactions between multimodal features. Others introduce significant parameter overhead, which contradicts the purpose of PE-VTR [35]. The above analysis demonstrates that there is still a significant research gap in addressing PE-VTR.

In this paper, we propose a novel method called multimodal video adapter for tackling PE-VTR. As illustrated in Fig. 1, MV-Adapter has two branches for video and text respectively. Each branch uses a bottleneck-style structure with three main operations: `Downsample-Transformer-Upsample`. We propose two novel components to address the challenges of PE-VTR. First, we introduce a temporal adaptation (TA) module in the video branch to enhance the temporal modeling capability. Unlike previous video adapters that apply identical weights across frames, we generate dynamic weights from both global and local features to better capture temporal variations in videos. Second, we propose a Cross Modality Tying (CMT) module that generates weights for the video and text branches from a modality shared parameter space. By implicitly "shortcutting" weights, models are apt to learn semantically aligned features between modalities, which aligns with the objective of VTR to bring multimodal representations closer together. Equipped with the above innovations, MV-Adapter is both parameter-efficient and performant on the PE-VTR task. Through extensive experiments on the five most commonly used VTR benchmarks, MV-Adapter achieves comparable or better results than the fully fine-tuned model, with negligible overhead (only 2.4% extra parameters). Compared with other VTR methods and Adapters, MV-Adapter significantly surpasses its competitors with large margins in V2T/T2V performance while using fewer parameters.

In summary, the contributions of our method are:

- We are among the first to take on the task of parameter-efficient VTR (PE-VTR) to promote its real-world applications. Current VTR methods face issues with applicability due to their large parameter overhead.
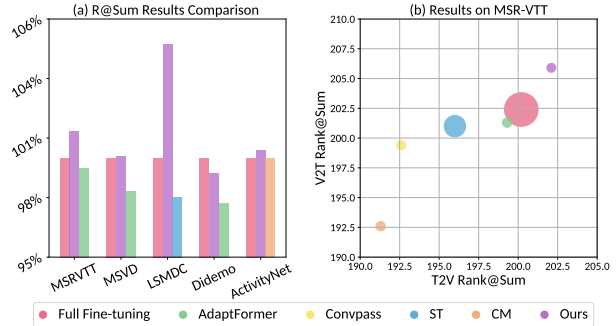

Figure 2. (a) The overall results on five widely used VTR benchmarks, We present the R@Sum (sum of the R@1, R@5, and R@10) results for the Text-to-Video and Video-to-Text tasks for full fine-tuning, ours, and the best baseline method, displayed as a ratio to the R@Sum of full fine-tuning. (b) Comparison of Text-to-Video and Video-to-Text R@Sum for different methods on MSR-VTT, where the radius of the circle is positively correlated with the trainable parameters.

- We propose a novel method called MV-Adapter to tackle PE-VTR, including two novel modules: temporal adaptation and cross modality tying. These modules effectively address the adaptation and efficiency challenges.
- We conduct extensive experiments on five widely-used VTR datasets to evaluate the performance of MV-Adapter. The results show that it achieves comparable or even better performance than standard full fine-tuning, and it also achieves the best trade-off between performance and efficiency among all competing methods.

2. Related Work

2.1. Image-text Pre-trained Model

With the increasing demand for model capabilities, a lot of works [1, 19, 24, 33, 38, 49] leverage large-scale Internet data to learn general representations. These works outperform previous methods on numerous downstream tasks, demonstrating the effectiveness of self-supervised learning on big data. Among many others, [38] is widely used by previous methods as backbone model. It has been demonstrated to provide solid prior knowledge for the downstream tasks, which is a better initialization than training from scratch. [1, 19, 38, 49] show encouraging results using the paradigm of pre-training followed by transfer learning.

2.2. Parameter-Efficient Transfer Learning

As the model grows larger, fully fine-tuning all parameters is prohibitively costly. Therefore, the demand for parameter-efficient transfer learning (PETL) increases. PETL methods can be broadly classified into two categories. The first category updates partial parameters in the model sparsely [12, 50]. The second approach update only newly added parameters/modules. For example, [17, 29] add or modify the QKV matrix in the transformer module. [11, 20] add learnable parameters to the input in the

form of prompts. Adapter [14] is one of the mainstream PETL methods in this direction. Early works [39, 40] introduce adapters to Computer Vision. [4] proposes a simple adapter AdaptFormer based on ViTs [7]. The Convpass[22] and ST-Adapter [35] utilize the spatial invariance and the temporal information of videos respectively. Adapters are also widely used in NLP [14, 16].

Adapter for multimodal tasks is relatively scarce. Previous works [4, 14, 22] focus on unimodal tasks like classification. VL-adapter [42] only adapts the text stream, while the visual projection of CLIP is fine-tuned. [21] adjusts CLIP by inserting a few parameterization layers while ignoring the temporal modeling in transfer learning. Uni-Adapter [30] and Aurora [43] respectively transfer BLIP [28] to some multimodal tasks in a parameter efficient manner through knowledge sharing and mode approximation. Different from above works, our method takes both modalities into consideration and adapts to the temporal domains.

2.3. Video Text Retrieval

[2, 3, 25, 41, 46, 51] are most widely used datasets in video-text retrieval (VTR). Early works [9, 13, 18, 26, 47] use offline features extracted by expert models for modal fusion. Since the emergence of the CLIP [38], [31, 37] transfer CLIP to VTR task. They show CLIP significantly outperformed the previous models. Afterward, using CLIP for the video-text retrieval task became a new paradigm. [10] uses text features as query vectors and applies the attention mechanism to image features. [44] designs a fine-grained token-wise interaction to calculate the similarity score. [34] designs a hierarchical aggregation mechanism of features. [32] designs a multi-grained interaction mechanism. However, all these works fine-tune the entire parameter set of CLIP, thus incurring high storage overhead. We focus on the parameter-efficient learning of VTR.

3. Methodology

3.1. Preliminary

In this sub-section, we briefly describe how VTR is performed by transferring from the pre-trained model CLIP [38]. We also introduce necessary notations used in the remainder of this paper. Like most state-of-the-art methods, CLIP is adopted due to its effectiveness and simple structure. Note our method can also be trivially extended to any CLIP-like backbone models that have the dual-encoder structure (for vision and text respectively).

In VTR, the goal is to learn a relevance function $sim(v, t)$ for calculating the similarity score between a video-text pair. In this way, given a video (text) query, we can obtain the rankings of all candidate texts (videos). Given a video-text pair (v, t), we use the vision encoder E_V and text encoder E_T in CLIP for extracting features
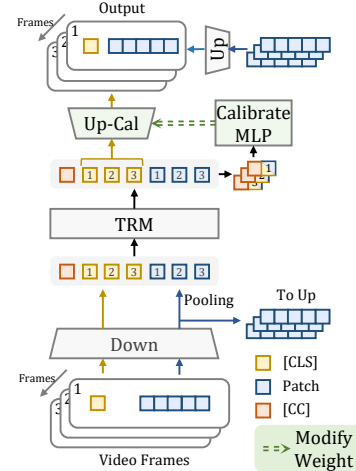

Figure 3. Illustration of temporal adaptation in visual branch, including temporal modeling using lightweight transformer block (TRM) and temporal calibration to generate dynamic upsample weights for each frame.

of v and t respectively. The depth of each encoder is denoted as L. We sample frames $(I_1, I_2, ..., I_{|v|})$ from v to represent the video as a sequence of images. Then, each frame is patchified and prepended with a special [CLS] token. We pass them through E_V and take the output of [CLS] token at the L-th layer as frame features $(E_V(I_1), E_V(I_2), ..., E_V(I_{|v|}))$. Finally, we obtain the global video features by aggregating frame features through mean-pooling: $e_v = \frac{1}{|v|} \sum_{k=0}^{|v|} E_V(I_k)$. Similarly, we pass the text t to get its encoding $e_t = E_T(t)$. We calculate the cosine similarity between video feature and text feature $sim(v, t) = \frac{e_v{}^T e_t}{\|e_v\|\|e_t\|}$.

3.2. Overview of MV-Adapter

As illustrated in Fig. 1, MV-Adapter adds a new branch in the video/text encoder of CLIP respectively and bridges them through CMT module. Each branch is placed after the feed-forward network (FFN) in transformer block. Though there are substantial differences between their concrete forms, the basic structures of video/text branches are the same, following the bottleneck-like processing flow: `Downsample-Transformer-Upsample` (Fig. 1). Both `Downsample` and `Upsample` are fully-connected layers, and the output is added to that of FFN. Formally, by denoting the output of FFN as $x \in \mathbb{R}^{N \times d}$ (N and d are the number and dimension of tokens respectively), the output of such an abstract structure is:

$$A_{\text{basic}}(x) = s \cdot \text{TRM}(x W_{\text{down}}) W_{\text{up}}, \qquad (1)$$

where $A_{\text{basic}}()$ denotes the functions of the abstract structure we employ and $\text{TRM}()$ denotes the lightweight transformer. $W_{\text{down}} \in \mathbb{R}^{d \times d'}$ and $W_{\text{up}} \in \mathbb{R}^{d' \times d}$ are the weights of `Downsample` and `Upsample` respectively, d' is the feature dimension after downsampling and s is scalar. For the text branch, we apply the process of Eq. (1) directly

Algorithm 1 Temporal Adaption

```
# B: batchsize, F: frame count, L: sequence length
# E: feature dimension, E': middle dimension, P: patch number
# x: video features in visual encoder layer, shape is (B,F,L,E)
# cc: learnable parameter for video representation
# trm_block: lightweight transformer block
# cal_mlp: 2-layers MLP to generate calibrate weight
# down, up: down/up sample linear layer
# scale: the scaling factor when adding to the original features
def forward(x):
    # Downsample [CLS] and patch
    x_down = down(x) # B,F,L,E'
    cls_seq = x_down[:,:,0] # B,F,E'
    patch_down = x_down[:,:,1:] # B,F,P^2,E'

    # Temporal Sequence Modeling using transformer block
    patch_seq = patch_down.mean(dim=2) # B,F,E'
    temporal_seq = concat([cc, cls_seq, patch_seq], dim=1)
    cc, cls_seq, patch_seq = trm_block(temporal_seq)

    # Temporal calibration weights (Eq. (2))
    cc = cc[:, None].expand(-1, F, -1) # B,F,E'
    cal_input = concat([cc, cls_seq + patch_seq], dim=-1)
    alpha_cal_up = cal_mlp(cal_input)

    # Generate dynamic calibrated upsample weights
    # for each frame (Eq. (3))
    up_w_cal = einsum('bfi,oi->bfio', alpha_cal_up, up.w)

    # Upsample process
    cls_up = einsum('bfio,bfi->bfo', up_w_cal, cls_seq)
    patch_up = up(patch_down)

    return x + scale * concat([cls_up, patch_up])
```

Algorithm 2 Downsample using CMT

```
# E: feature dimension of current modality, E': middle dimension
# x: video features in visual encoder layer, shape is (*,E)
# down: downsample linear layer without modality interaction
# cmt_factor: cross modality factor in R^{d^m}
# cmt_proj: modality sepcific projector
def down(x):
    # project the shared factor, and generate modality-aware
    # downsample weight down_w_cal (Eq. (4))
    beta_cal = cmt_proj(cmt_factor)
    down_w_cal = einsum('i,oi->oi', beta_cal, down.w)
    return x @ down_w_cal.T
```

Consequently, each frame's [CLS] and averaged patch embeddings are enriched to be temporal-aware, and [CC] obtains global video representation, which will be used in the subsequent process to generate dynamic upsample weights for each frame.

To further enhance the temporal information on the [CLS] token, we design a temporal calibration module, which jointly use the video representation of $x_{[CC]}$, each frame's [CLS] feature $x^i_{[CLS]}$ and patch feature $\bar{x}^i_{patch}$. Specifically, for i-th frame, we fuse above three features to obtain a calibration vector $\alpha^i = \mathrm{concat}(x_{[CC]}, x^i_{[CLS]} + \bar{x}^i_{patch}) \in \mathbb{R}^{2d'}$. Then, we generate the calibration weights $\alpha^i_{cal} \in \mathbb{R}^{d'}$ by feeding α^i into a two-layer calibration MLP:

$$\alpha^i_{cal} = \mathtt{FC_2}(\mathrm{ReLU}(\mathtt{FC_1}(\alpha^i))), \tag{2}$$

where $\mathtt{FC_1}()$ and $\mathtt{FC_2}()$ are fully connected layers whose weights dimensions are of $\mathbb{R}^{2d' \times (d'/\sigma)}$ and $\mathbb{R}^{(d'/\sigma) \times d'}$ respectively. σ is a shrinkage factor. Then we calibrate the weights of `Upsample` using α^i_{cal} as follows,

$$(W^i_{up\text{-}cal})_c = \alpha^i_{cal} \odot (W_{up})_c, \tag{3}$$

where $(W^i_{up})_c \in \mathbb{R}^d$ denotes the c-th column in W_{up}. This calibration module enables the generation of individual dynamic weights for each frame, and `Upsample` the [CLS] token of i-th frame using the calibrated weight W^i_{up-cal}. For the patches in each frame x^i_{patch}, the information density is not as concentrated as that of the [CLS] token, and the temporal information modeling can be obtained in the next layer through the interaction with the [CLS] token using the attention mechanism. Therefore, we directly pass them through `Downsample` to reduce dimension, and then through `Upsample` after activation to reduce computation.

This process has two advantages. First, the weights change across different frames, allowing the model to capture the intricate variations of video dynamics. Second, since the video description x_{CC}, frame representations x^i_{CLS} and local feature $\bar{x}^i_{patch}$ contain multi-grained context, features are enhanced with rich temporal contexts during the process. This strategy is considerably more appropriate for adapting to videos than using fixed weights in the `Upsample` layer. Experimental results in Sec. 5.3 evidence the superiority of our method.

3.4. Cross Modality Tying

Essentially, VTR brings the features of video and text closer in a joint embedding space. From this point of view, we de-

where N is the number of words tokens. For the video branch, we make significant modifications to the basic form to efficiently capture temporal cues. In addition, we propose a cross modality tying module with a shared parameter space to adjust weights for the `Downsample` in both modalities. We present more details of these components in the following sections.

3.3. Temporal Adaptation

The goal of the video branch in MV-Adapter is to augment the image-text pre-trained vision encoder of CLIP with temporal modeling capability. To achieve this, as illustrated in Fig. 3, we encode temporal context into the spatial features of each frame and further enhance the model with dynamic temporal modeling. Before delving into details, we first expand the notation of the input of vision branch x for description clarity. Considering all frames, $x = \{x^i\}_{i=1}^{|v|}$ where x^i is the feature of i-th frame and $|v|$ is the number of frames. $x^i = [x^i_{cls}, x^i_{patch}] \in \mathbb{R}^{(N_P+1) \times d}$ where $x^i_{patch} = [x^i_1, \cdots, x^i_{N_P}]$ and N_P is the number of patches in each frame. Next, we detail how to enrich the image-level feature (for the [CLS] token) and patch-level features with temporal context. The pseudocode for the entire temporal adaption process is shown in Algorithm. 1.

The [CLS] token's feature $x^i_{[CLS]}$ is first passed through the `Downsample` to reduce its dimension for computational efficiency. Then we concatenate all frames' [CLS] tokens and the average of patch tokens in the order of the frames in the video. And a learnable token, denoted as [CC] ("CC" means the aggregated "Class token" of framewise "Class tokens"), is appended to this sequence. This sequence is fed into the lightweight transformer `TRM` as shown in Eq. (1). By utilizing the attention mechanism to capture the mutual dependencies among frame tokens, the transformer learns the temporal information across all frames.

sign a cross modality tying (CMT) module for facilitating the alignment between modalities. Specifically, between the corresponding layers of visual and text encoders, we share a cross modality factor $f_C \in \mathbb{R}^{1 \times d^m}$, where d^m is the dimension of the shared factor. Subsequently, within each individual modality branch, we construct a modality specific projection matrix $M_S \in \mathbb{R}^{d^m \times d}$. During the `Downsample` process, we utilize these parameters to adjust the weights of `Downsample` across modalities. We employ the modality specific projection matrix to project the shared factors across modalities onto the feature dimension corresponding to the encoder of current modality, thereby obtaining the modality calibration vector $\beta_{\text{cal}}^{\text{T/V}} = f_C \times M_S^{\text{T/V}}$, where T/V means Text or Visual branch. Then, we generated the modality-aware `Downsample` weight using a method similar to that described in Eq. 4 as follows:

$$(W_{\text{down-cal}}^{\text{T/V}})_c = \beta_{\text{cal}}^{\text{T/V}} \odot (W_{\text{down}})_c, \qquad (4)$$

and `Downsample` features with $W_{\text{down-cal}}^{\text{T/V}}$ rather than the original `Downsample` weights. The pseudocode to describe the CMT process is shown in Algorithm 2.

By sharing factor between modalities, CMT can implicitly reduce the distance in the embedding space, rather than forcibly aligning. Furthermore, it is a low-coupling method that allows the model to separately extract embeddings of videos and sentences, hence having much lower time complexity than other methods [23, 31] (i.e., $O(\#\text{videos} + \#\text{texts})$ vs. $O(\#\text{videos} \times \#\text{texts})$) that require inputting video-text pairs together to obtain embeddings.

3.5. Efficiency Analysis

Deployment Efficiency The MV-Adapter can significantly reduce the storage costs in deployment while maintaining performance, especially when serving a large amount of VTR tasks. This is due to its tunable parameter complexity of $O(2d \times d' + k(d')^2)$, where k is a constant that encompasses the parameter of both temporal calibration and TRM. MV-Adapter achieves this by storing the shared pre-trained model once and only a small number of tunable parameters for each task. Put formally, suppose the size of pre-trained model is a unit, MV-Adapter reduces the space usage from `#1 × tasks` to $1 + \#\text{tasks} \times 2.4\%$. Taken the five VTR benchmarks as examples, MV-Adapter can support all five tasks using **only 112%** times the storage space of the pre-trained model, while the full fine-tuning and hunyuan_tvr [34] takes 500% and about 600% respectively.

Training Efficiency Besides, we also observe that MV-Adapter is more efficient and less computing resource demanding. It can considerably reduce around 40% GPU memory costs compared to full fine-tuning, as the majority parts of model are frozen. We argue that this appealing property of MV-Adapter makes various training optimizations feasible, such as using a larger batch size to improve

contrastive learning [5, 15, 45].

4. Experiment Setup

4.1. Datasets and Evaluation Metric

We evaluate our module on five commonly used public video-text retrieval datasets: MSR-VTT, MSVD, LSMDC, DiDemo and ActivityNet.

MSR-VTT [46] is widely used in previous literature, consisting of 10,000 videos with 20 captions each. The training set we use has 9k video-text pairs [8], and the test set has 1k video-text pairs [48].

MSVD [3] contains 1970 videos with approximately 40 captions each. Train, validate and test sets have 1200, 100, and 670 videos respectively.

LSMDC [41] contains 118081 video-text pairs, and all the videos are extracted from 202 movies. The movie sources for the training and test videos are independent.

DiDemo [2] contains 10000 videos, and each has four sentences concatenated as the whole caption following [31].

ActivityNet [25] has 20000 Youtube videos. Similarly as in [2], we combine all sentences as the final text.

We use the standard retrieval metrics: recall at rank K (R@K, higher is better) following previous practices. To eliminate randomness, we report both the mean and std of results on three seeds (0, 42, 123) for each setup. We also report the sum of R@1/5/10 (higher is better).

4.2. Implementation Detail

We adopt CLIP (ViT-B/16) [38] as the backbone of our model by default and ablate using CLIP (ViT-L/14) to validate the scalability of MV-Adapter. The parameters of adapter are optimized by Adam, while the learning rate is 5e-6. For MSR-VTT, MSVD, and LSMDC datasets, the max lengths of frames and words in captions are 12 and 32, and we train the module for 5 epochs. For ActivityNet and DiDemo datasets, we set the lengths of video and caption to 32 and 64 respectively as they are longer and use 15 epochs. The batch size is 128 by default and 64 for Didemo and ActivityNet due to GPU memory limit. Following [31], we extract 1 frame per second from the videos and select the frames uniformly when the number of frames is larger than the max length. The middle dimension of MV-Adapter is 64, and the shrinkage factor σ and s are set as 4, 0.1.

5. Results And Analysis

5.1. Baselines

As a parameter-efficient method, we first compare with full fine-tuning. To further verify the effectiveness of our method, we build strong baselines by adapting popular vision adapters to the VTR task. The details are as follows.

Full fine-tuning. Update all parameters for each task,

Method	Tunable Params(%)	Text-to-Video				Video-to-Text			
		R@1	R@5	R@10	Sum	R@1	R@5	R@10	Sum
Full Fine-tuning	100	45.0	73.0	82.2	200.2	45.3	73.4	83.7	202.4
Adapt_{par} [4]	2.78	45.5 ± 0.2	72.3 ± 0.4	81.5 ± 0.5	199.3	45.7 ± 0.2	72.9 ± 0.2	82.7 ± 0.3	201.3
Adapt_{seq} [4]	2.78	45.7 ± 0.1	72.4 ± 0.4	81.0 ± 0.2	199.1	45.2 ± 0.2	72.8 ± 0.9	81.7 ± 0.6	199.7
Convpass [22]	2.80	42.5 ± 0.5	69.8 ± 0.2	80.4 ± 0.1	192.6	44.7 ± 1.0	72.5 ± 0.1	82.1 ± 0.2	199.4
ST [35]	5.93	43.6 ± 0.5	70.9 ± 0.5	81.4 ± 0.7	196.0	45.5 ± 0.8	72.7 ± 0.2	82.7 ± 0.6	201.0
CM [21]*	2.76	42.1 ± 0.4	69.5 ± 0.3	79.7 ± 0.2	191.3	42.0 ± 0.6	69.9 ± 0.3	80.7 ± 0.5	192.6
CLIP4Clip [31]	8.45	42.1	68.3	78.8	189.2	40.2	68.1	79.1	187.4
Hunyuan [34]*	11.97	43.8 ± 0.3	70.9 ± 0.7	81.1 ± 0.6	195.8	41.2 ± 1.0	70.5 ± 0.6	80.6 ± 0.6	192.3
MV-Adapter	**2.39**	$\mathbf{46.2} \pm 0.5$	$\mathbf{73.2} \pm 0.3$	$\mathbf{82.7} \pm 0.3$	**202.1**	$\mathbf{47.2} \pm 0.4$	$\mathbf{74.8} \pm 0.3$	$\mathbf{83.9} \pm 0.5$	**205.9**

Table 1. Comparison results on MSR-VTT [46] using CLIP (ViT-B/16). **bold** and underline indicates the top two results. * denotes our reproduced results.

Method	Tunable Params(%)	Text-to-Video				Video-to-Text			
		R@1	R@5	R@10	Sum	R@1	R@5	R@10	Sum
Full Fine-tuning	100	**49.7**	**79.2**	**87.3**	**216.2**	71.2	92.5	95.5	259.3
Adapt_{par} [4]	2.78	48.3 ± 0.1	77.7 ± 0.0	86.8 ± 0.0	212.8	68.9 ± 0.3	90.9 ± 0.6	94.8 ± 0.2	254.6
Adapt_{seq} [4]	2.78	45.1 ± 0.2	76.2 ± 0.2	85.3 ± 0.1	206.7	62.6 ± 0.6	88.8 ± 0.2	93.4 ± 0.4	244.8
Convpass [22]	2.80	46.1 ± 0.3	76.0 ± 0.3	85.5 ± 0.1	207.6	70.0 ± 1.2	91.9 ± 0.6	95.5 ± 0.6	257.4
ST [35]	5.93	45.9 ± 0.2	75.9 ± 0.2	85.2 ± 0.2	207.0	69.9 ± 1.6	91.7 ± 1.0	95.1 ± 0.4	256.7
CM [21]*	2.76	44.7 ± 0.8	75.8 ± 0.4	85.0 ± 0.3	205.5	61.5 ± 0.9	88.2 ± 0.5	93.2 ± 0.6	242.9
CLIP4Clip [31]	8.45	45.0	74.2	83.4	202.5	62.2	88.7	94.0	244.9
Hunyuan [34]*	11.97	43.0 ± 0.4	73.7 ± 0.2	83.6 ± 0.2	200.3	53.7 ± 2.1	83.9 ± 2.2	90.2 ± 1.5	227.8
MV-Adapter	**2.39**	49.4 ± 0.2	78.3 ± 0.1	87.0 ± 0.1	214.8	$\mathbf{71.8} \pm 0.5$	$\mathbf{93.0} \pm 0.4$	$\mathbf{96.4} \pm 0.1$	**261.2**

Table 2. Comparison results on MSVD [3] using CLIP (ViT-B/16). **bold** and underline indicates the top two results. * denotes our reproduced results.

which refers to the "MeanP" Method in [31].

AdaptMLP [4]. This module is a bottleneck block containing two fully connected layers $W_{\text{down}} \in \mathbb{R}^{d \times d'}$ and $W_{\text{up}} \in \mathbb{R}^{d' \times d}$, where d is the feature dimension and $d' \ll d$. AdaptMLP has two forms according to its location in the transformer block. In *parallel* form, it is placed after FFN and takes the form of

$$A(x) = x + s \cdot \text{ReLU}(xW_{\text{down}})W_{\text{up}},$$

where x is the input of FFN as in Eq. (1). In *sequential* form, it has the form of

$$A(x) = \text{FFN}(x) + s \cdot \text{ReLU}(\text{FFN}(x)W_{\text{down}})W_{\text{up}}.$$

Due to its simplicity, AdaptMLP can be conveniently applied to transformer structures. Therefore, we use sequential AdaptMLP (for its better performance) in the text encoder to tailor the following vision adapters to VTR. We use $\text{Adapt}_{par/seq}$ to represent these forms in result tables.

Convpass [22]. It uses convolution for adaptation in vision domains. Specifically, the patch features are separated from the [CLS] features and reshaped to the 2D matrix according to their spatial locations. Then, both the [CLS] and patches features are fed into Convpass respectively

$$A(x) = \text{FFN/MHSA}(x) + s \cdot \text{Conv}_{3 \times 3}(xW_{\text{down}})W_{\text{up}}.$$

ST-Adapter [35]. This module is for adapting image model to video tasks, employed at the beginning of each transformer block. It uses a depth-wise 3D-convolution to capture Spatio-Temporal information as follows

$$A(x) = x + \text{DWConv}(xW_{\text{down}})W_{\text{up}}.$$

CM-Adapter [21]. Based on AdaptMLP [4], it simply appends a shared learnable weight between encoders. The module is inserted to each layer of CLIP.

$$A(x) = x + \sigma(xW_{\text{down}})\text{Concat}[W_{\text{up}}, W_{\text{up}}^{\text{cm}}],$$

where $\sigma(\cdot)$ is activation function.

For the above methods, we add and train AdaptMLP in their text encoders to ensure fair comparison. AdaptMLP is selected for its wide use and effectiveness across modalities. Furthermore, we adapt the state-of-the-art VTR methods using full fine-tuning: Hunyuan [34] and CLIP4Clip (+SeqTransformer) [31] to PE-VTR by freezing the update of CLIP during training. In this way, they can be regarded as adapters to the output of the last layer.

5.2. Main Result

In this section, we present in the detail the comparison of MV-Adapter with other methods on MSR-VTT, MSVD, LSMDC, Didemo, and ActivityNet. The results of experiments are shown in Tabs. 1 to 5 (note that the results of full fine-tuning and CLIP4Clip with SeqTransf are constant, as the initialization is fixed). We first compare our method with baseline method which uses standard full fine-tuning. Overall, we find our method performs on par or even better (in most cases) than the baseline. For example, on the T2V task, MV-Adapter surpasses the full fine-tuning by 1.9, 7.4 and 1.6 on the sum of R@1/5/10 on MSR-VTT, LSMDC and ActivityNet respectively. Similarly, on V2T, MV-adapter outperforms the full fine-tuning by 3.5, 1.9, and 5.6 on MSR-VTT, MSVD, and LSMDC.

Afterwards, we adapt previous PETL methods including AdaptFormer [4], Convpass [22] and ST-adapter [35] to VTR as introduced in Sec. 5.1. Also, we compare with state-of-the-art methods in the full fine-tuning set-

Method	Tunable Params(%)	Text-to-Video				Video-to-Text			
		R@1	R@5	R@10	Sum	R@1	R@5	R@10	Sum
Full Fine-tuning	100	20.2	41.5	51.2	112.9	21.8	40.5	50.9	113.2
Adapt_{par} [4]	2.78	19.8 ± 0.4	39.0 ± 0.3	48.8 ± 0.5	107.6	21.0 ± 0.1	38.0 ± 0.7	46.6 ± 0.9	105.5
Adapt_{seq} [4]	2.78	20.7 ± 0.5	40.2 ± 0.2	50.2 ± 0.4	111.1	20.9 ± 0.1	38.8 ± 0.5	49.1 ± 0.4	108.8
Convpass [22]	2.80	19.7 ± 0.1	37.8 ± 0.4	46.3 ± 0.5	103.8	20.7 ± 0.5	38.9 ± 0.3	47.7 ± 0.7	107.2
ST [35]	5.93	20.9 ± 0.6	39.8 ± 0.4	49.1 ± 0.9	109.8	21.9 ± 0.2	40.3 ± 0.3	49.5 ± 0.6	111.8
CM [21]*	2.76	18.7 ± 0.5	38.7 ± 0.3	48.3 ± 0.1	105.7	20.9 ± 0.3	37.8 ± 1.0	47.7 ± 0.5	106.3
CLIP4Clip [31]	8.45	20.1	37.4	46.0	103.5	18.1	34.6	43.9	96.6
Hunyuan [34]*	11.97	20.6 ± 0.3	37.6 ± 0.5	46.7 ± 0.6	104.9	18.5 ± 0.8	36.6 ± 0.5	44.5 ± 0.8	99.6
MV-Adapter	2.42	23.2 ± 0.7	43.9 ± 0.5	53.2 ± 0.6	120.3	24.0 ± 0.5	42.8 ± 0.4	52.1 ± 0.2	118.8

Table 3. Comparison results on LSMDC [41] using CLIP (ViT-B/16). **bold** and <u>underline</u> indicates the top two results. * denotes our reproduced results.

Method	Tunable Params(%)	Text-to-Video				Video-to-Text			
		R@1	R@5	R@10	Sum	R@1	R@5	R@10	Sum
Full Fine-tuning	100	44.7	73.6	81.2	199.6	45.0	73.1	80.8	198.9
Adapt_{par} [4]	2.78	42.6 ± 0.2	72.5 ± 0.4	80.8 ± 0.5	195.9	43.0 ± 0.0	70.6 ± 0.4	80.0 ± 0.7	193.5
Adapt_{seq} [4]	2.78	42.4 ± 0.8	70.6 ± 0.4	80.3 ± 0.1	193.3	42.2 ± 0.6	69.9 ± 0.3	79.7 ± 0.2	191.8
Convpass [22]	2.80	40.7 ± 1.1	69.6 ± 0.5	78.2 ± 0.7	188.6	40.9 ± 0.3	70.0 ± 0.8	79.2 ± 1.0	190.0
ST [35]	5.93	40.4 ± 0.1	69.4 ± 0.6	79.2 ± 0.3	189.0	41.2 ± 1.1	70.1 ± 0.4	80.1 ± 0.5	191.4
CM [21]*	2.77	42.9 ± 0.5	70.6 ± 0.3	80.2 ± 0.1	193.7	42.5 ± 1.0	70.8 ± 0.4	80.5 ± 0.2	193.8
CLIP4Clip [31]	8.45	36.2	61.8	72.7	170.7	34.4	62.4	72.9	169.7
Hunyuan [34]*	11.97	37.0 ± 0.6	64.2 ± 1.2	74.3 ± 1.0	175.5	34.6 ± 1.0	63.0 ± 1.1	73.6 ± 0.3	171.2
MV-Adapter	2.42	44.3 ± 0.4	72.1 ± 0.6	80.5 ± 0.2	196.8	42.7 ± 1.0	73.0 ± 0.7	81.9 ± 0.5	197.6

Table 4. Comparison results on DiDemo [2] using CLIP (ViT-B/16). **bold** and <u>underline</u> indicates the top two results. * denotes our reproduced results.

Method	Tunable Params(%)	Text-to-Video				Video-to-Text			
		R@1	R@5	R@10	Sum	R@1	R@5	R@10	Sum
Full Fine-tuning	100	42.9	73.2	85.4	201.5	43.8	75.0	86.6	205.3
Adapt_{par} [4]	2.78	41.7 ± 0.0	72.7 ± 0.2	84.5 ± 0.1	198.8	43.3 ± 0.0	73.1 ± 0.1	85.8 ± 0.1	202.2
Adapt_{seq} [4]	2.78	41.2 ± 0.1	72.2 ± 0.1	84.1 ± 0.0	197.5	42.5 ± 0.2	73.0 ± 0.2	85.2 ± 0.1	200.7
Convpass [22]	2.80	38.6 ± 0.3	69.9 ± 0.5	82.7 ± 0.3	191.2	40.5 ± 0.1	71.9 ± 0.3	84.1 ± 0.1	196.6
ST [35]	5.93	38.7 ± 0.3	69.8 ± 0.3	83.1 ± 0.3	191.6	40.7 ± 0.3	71.8 ± 0.3	84.2 ± 0.4	196.7
CM [21]*	2.25	43.1 ± 0.2	74.2 ± 0.1	85.5 ± 0.1	202.8	43.5 ± 0.5	74.4 ± 0.1	86.2 ± 0.2	204.0
CLIP4Clip [31]	8.45	36.9	68.5	8.01	186.4	35.1	68.4	82.0	185.5
Hunyuan [34]*	11.97	37.8 ± 0.3	70.4 ± 0.4	83.5 ± 0.2	191.7	35.1 ± 0.2	69.1 ± 0.9	83.0 ± 0.5	187.2
MV-Adapter	2.40	42.9 ± 0.0	74.5 ± 0.1	85.7 ± 0.1	203.1	43.6 ± 0.1	75.0 ± 0.3	86.5 ± 0.1	205.2

Table 5. Comparison results on ActivityNet [25] using CLIP (ViT-B/16). **bold** and <u>underline</u> indicates the top two results. * denotes our reproduced results.

ting by freezing the CLIP backbone: CLIP4Clip (+seqr-Transformer) [31] and Hunyuan [34]. As shown in Fig. 4, MV-Adapter consistently outperforms other methods with the smallest parameter overhead. Specifically, compared with other methods, MV-Adapter shows signficant improvements. Measured by the R@Sum of T2V/V2T, MV-Adapter outperforms the second best method by 6.1/4.9, 2.0/6.6, 10.5/7.0, 0.9/4.1 and 0.3/1.2 on MSR-VTT, MSVD, LSMDC, Didemo and ActivityNet, respectively. MV-Adapter achieves this with about 60% of the GPU memory usage as the tunable parameters are small.

5.3. Ablations

We conducted extensive ablative experiments to validate the effectiveness of the design choices of MV-Adapter. In this section, we measure the model on MSR-VTT[46]. For simplicity, we use the R@1 and the sum of R@1/5/10 averaged on three seeds as the metric.

Temporal Adaptation. In this experiment, we investi-gate the effects of the proposed temporal adaptation module. Results are listed on Sec. 5.3. First, we construct a baseline model that simply applies the bottleneck structure. This model achieves only 42.9/43.4 T2V/V2T R@1 performance which is much worse than the performance of full fine-tuning (45.7/45.3). Then we add the temporal modeling using transformer block as introduced in Sec. 3.3, the performance significantly improves to 45.0/46.0. This verifies the importance of learning temporal contexts across frames. Furthermore, with the temporal calibration, the model is boosted to 45.9/46.6, outperforming the fully fine-tuned model. Above results strongly demonstrate the effectiveness of temporal adaptation.

Multimodal Branches and CMT. Since VTR inherently involves learning from two modalities, it is natural to assume that multimodal adaptation is more suitable for the task. To corroborate this, we conduct experiments using different combinations of video/text branches. As shown in the Tab. 6 (b), applying only text branch or video branch

Query1: sports people are fighting on field
Query2: some people are shooting outside
Query3: a man runs into the crowd trying to catch a basketball

Query4:
Baseline: a diver goes underwater
Ours: woman with swimming suit is plunging into the water and gets rid of the suit

Query5:
Baseline: a man rowing a kayak is shown in slow motion
Ours: several groups of people are kayaking on a waterway

Query6:
Baseline: a woman cooking an orange substance
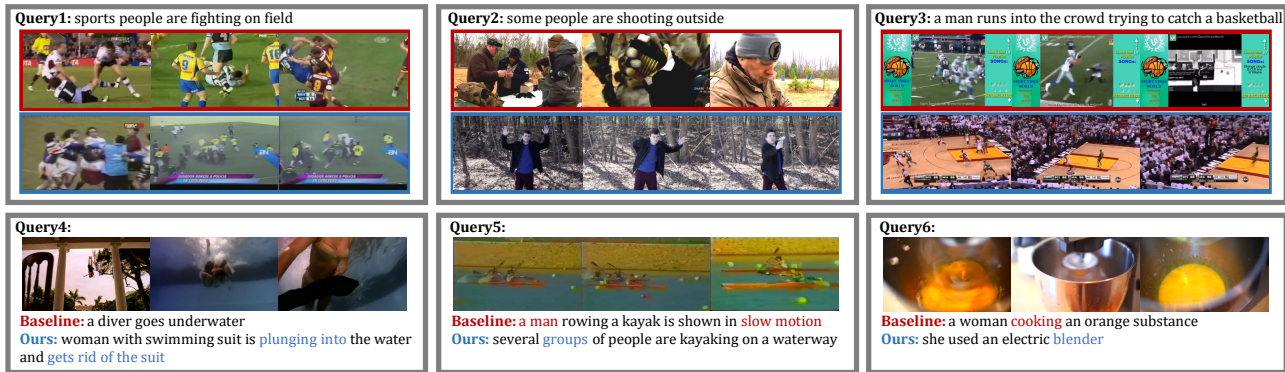**Ours:** she used an electric blender

Figure 4. Visualizations of text-to-video (top row) and video-to-text (bottom row) results from MV-Adapter and ST-Adapter [35] using the same query from MSR-VTT. In each example, the retrieval results of baseline and MV-Adapter are shown in red and blue respectively.

Settings	T2V	V2T	Visual	Text	CMT	T2V	V2T	Method	T2V	V2T
Full FT	45.0/200.2	45.3/202.4		✓		44.0/193.1	45.2/199.1	Full FT	49.2/208.0	49.2/211.7
Down&Up	42.9/192.0	43.4/195.3	✓			44.2/191.7	42.0/192.3	Ours	**49.3/209.3**	**49.6/212.4**
w/ TRM	45.0/197.5	46.0/202.9	✓	✓		45.9/200.9	46.6/204.4			
w/ calibration	**45.9/200.9**	**46.6/204.4**	✓	✓	✓	**46.2/202.1**	**47.2/205.9**			

Table 6. Ablations. We report R@1 and the R@sum on MSR-VTT, where "Full FT" means Full Fine-tuning Method. Each reported result represents the average derived from experiments conducted on three different seeds (0, 42, 123). Due to space limit, please refer to more ablations in the Supp. (a) Left. Performance improvements brought by adding temporal modeling and then adding temporal calibration. (b) Middle. Analysis of using video/text branches and CMT. (c) Right. Scalability analysis based on CLIP (ViT-L/14).

all decreases the performances (-7.8/5.3 and -9.2/12.1 on the T2V/V2T R@Sum respectively), and CMT further improves the performance of the model by 1.2/1.5 on the T2V/V2T R@Sum. These results evidently support the utilization of both branches and CMT in MV-Adapter.

Model Scalability. In order to validate the scalability of MV-Adapter, we use the CLIP V-L/14 as the backbone to perform the ablation study. Since CLIP V-L/14 has a higher feature dimension in the encoder than its base version, we adjust the middle dimension of MV-Adapter to 128 to accommodate the larger backbone. We first obtain fully fine-tuned results as the baseline, then experiment with MV-Adapter. The experimental results in Tab. 6 (c) are consistent with those of CLIP V-B/16, where MV-Adapter achieves better results with small parameter overhead.

More ablations. Due to space limit, please refer to the Supplementary Material for ablations on other datasets and ablations of the validity and locations of CMT.

5.4. Qualitative Results

In order to better understand how MV-Adapter performs compared with other methods, we illustrate some qualitative results on V2T and T2V tasks in Fig. 4. We use ST-Adapter [35] for comparison since its performance on MSR-VTT is only worse than ours. As can be observed in Fig. 4, MV-Adapter is capable of modeling rich spatiotemporal information in videos including dynamic movements and relationships among objects, thus building more accurate correspondence between video and text. In contrast, the baseline method lacks in temporal modeling capa-

bilities, thus failing to comprehend complex video scenes. For example, in Query1 (T2V), the video returned by MV-Adapter accurately understand the complete action of fighting, while the baseline method erroneously returns clips of competitive sports confrontation. Similarly, when using video Query4 (V2T) for retrieval, MV-Adapter can correctly understand the action and the entire temporal process. In comparison, the baseline method only attends to the main scene of the video, which is swimming underwater, without understanding the correlation before and after.

We summarize two failure modes from case studies. (1) Since we use a fixed number of frames per video in training, it may fail to capture fine movements in long videos as the differences between frames aggregate. Employing a sliding window of videos may be helpful. (2) Results may be incorrect when the caption is related to the audio aspect. Performing ASR or augmenting the input with audio features may solve this issue. Examples will be shown in Supp.

6. Conclusion

In this paper, we propose the task of parameter-efficient VTR (PE-VTR) to save storage costs of models. Previous methods fail to address PE-VTR, leading to inferior performance and/or large parameter overheads. To tackle PE-VTR, we introduce MV-Adapter with two novel components: temporal adaptation module and cross modality tying. MV-Adapter achieves comparable results with full fine-tuning and outperforms competing methods. In the future, we plan to extend MV-Adapter to more multimodal tasks such as video question answering and captioning.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022. 2

[2] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE international conference on computer vision*, pages 5803–5812, 2017. 3, 5, 7

[3] David Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 190–200, 2011. 3, 5, 6

[4] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022. 2, 3, 6, 7

[5] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 5

[6] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020. 1

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[8] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision*, pages 214–229. Springer, 2020. 5

[9] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision*, pages 214–229. Springer, 2020. 1, 3

[10] Satya Krishna Gorti, Noël Vouitsis, Junwei Ma, Keyvan Golestan, Maksims Volkovs, Animesh Garg, and Guangwei Yu. X-pool: Cross-modal language-video attention for text-video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5006–5015, 2022. 1, 3

[11] Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*, 2021. 2

[12] Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020. 2

[13] Feng He, Qi Wang, Zhifan Feng, Wenbin Jiang, Yajuan Lü, Yong Zhu, and Xiao Tan. Improving video retrieval by adaptive margin. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1359–1368, 2021. 1, 3

[14] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021. 2, 3

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 5

[16] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. 3

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 2

[18] Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2758–2766, 2017. 1, 3

[19] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021. 2

[20] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022. 2

[21] Haojun Jiang, Jianke Zhang, Rui Huang, Chunjiang Ge, Zanlin Ni, Jiwen Lu, Jie Zhou, Shiji Song, and Gao Huang. Cross-modal adapter for text-video retrieval. *arXiv preprint arXiv:2211.09623*, 2022. 3, 6, 7

[22] Shibo Jie and Zhi-Hong Deng. Convolutional bypasses are better vision transformer adapters. *arXiv preprint arXiv:2207.07039*, 2022. 2, 3, 6, 7

[23] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR, 2021. 5

[24] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Large scale learning of general visual representations for transfer. *arXiv preprint arXiv:1912.11370*, 2(8), 2019. 2

[25] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017. 3, 5, 7

[26] Thao Minh Le, Vuong Le, Svetha Venkatesh, and Truyen Tran. Hierarchical conditional relation networks for video question answering. In *Proceedings of the IEEE/CVF con-*

ference on computer vision and pattern recognition, pages 9972–9981, 2020. 1, 3

[27] Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L Berg, Mohit Bansal, and Jingjing Liu. 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*, pages 7331–7341, 2021. 1

[28] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022. 3

[29] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 2

[30] Haoyu Lu, Mingyu Ding, Yuqi Huo, Guoxing Yang, Zhiwu Lu, Masayoshi Tomizuka, and Wei Zhan. Uniadapter: Unified parameter-efficient transfer learning for cross-modal modeling. *arXiv preprint arXiv:2302.06605*, 2023. 3

[31] Huaishao Luo, Lei Ji, Ming Zhong, Yang Chen, Wen Lei, Nan Duan, and Tianrui Li. Clip4clip: An empirical study of clip for end to end video clip retrieval and captioning. *Neurocomputing*, 508:293–304, 2022. 1, 2, 3, 5, 6, 7

[32] Yiwei Ma, Guohai Xu, Xiaoshuai Sun, Ming Yan, Ji Zhang, and Rongrong Ji. X-clip: End-to-end multi-grained contrastive learning for video-text retrieval. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 638–647, 2022. 1, 3

[33] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196, 2018. 2

[34] Shaobo Min, Weijie Kong, Rong-Cheng Tu, Dihong Gong, Chengfei Cai, Wenzhe Zhao, Chenyang Liu, Sixiao Zheng, Hongfa Wang, Zhifeng Li, et al. Hunyuan_tvr for text-video retrivial. *arXiv preprint arXiv:2204.03382*, 2022. 1, 2, 3, 5, 6, 7

[35] Junting Pan, Ziyi Lin, Xiatian Zhu, Jing Shao, and Hongsheng Li. St-adapter: Parameter-efficient image-to-video transfer learning for action recognition. *arXiv preprint arXiv:2206.13559*, 2022. 2, 3, 6, 7, 8

[36] Matthew E Peters, Sebastian Ruder, and Noah A Smith. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*, 2019. 1

[37] Jesús Andrés Portillo-Quintero, José Carlos Ortiz-Bayliss, and Hugo Terashima-Marín. A straightforward framework for video retrieval using clip. In *Mexican Conference on Pattern Recognition*, pages 3–12. Springer, 2021. 1, 3

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1, 2, 3, 5

[39] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30, 2017. 3

[40] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8119–8127, 2018. 3

[41] Anna Rohrbach, Marcus Rohrbach, and Bernt Schiele. The long-short story of movie description. In *German conference on pattern recognition*, pages 209–221. Springer, 2015. 3, 5, 7

[42] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5237, 2022. 3

[43] Haixin Wang, Xinlong Yang, Jianlong Chang, Dian Jin, Jinan Sun, Shikun Zhang, Xiao Luo, and Qi Tian. Parameter-efficient tuning of large-scale multimodal foundation model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[44] Qiang Wang, Yanhao Zhang, Yun Zheng, Pan Pan, and Xian-Sheng Hua. Disentangled representation learning for text-video retrieval. *arXiv preprint arXiv:2203.07111*, 2022. 1, 3

[45] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018. 5

[46] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016. 3, 5, 6, 7

[47] Youngjae Yu, Hyungjin Ko, Jongwook Choi, and Gunhee Kim. End-to-end concept word detection for video captioning, retrieval, and question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3165–3173, 2017. 1, 3

[48] Youngjae Yu, Jongseok Kim, and Gunhee Kim. A joint sequence fusion model for video question answering and retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 471–487, 2018. 5

[49] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021. 2

[50] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021. 2

[51] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 3