IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

DOI:10.1587/transfun.2024EAP1080

Publicized:2024/10/28

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

MST-Adapter : Multi-scaled Spatio-Temporal Adapter for Parameter-Efficient Image-to-Video Transfer Learning

Chenrui CHANG[†a)], **Tongwei LU**[†b)], ***and*** **Feng YAO**[†c)], ***Nonmembers***

SUMMARY Large-scale image pre-training models have recently demonstrated strong representation capabilities in spatial information contexts. Prior works apply these models to video action recognition through fully fine-tuning, which is expensive and resource-intensive. To reduce computational costs, some studies have shifted their focus to efficient parameter fine-tuning methods. However, existing efficient fine-tuning methods lack exploration of multi-scale information in videos. In this work, the Multi-scale spatio-temporal Adapter (MST-Adapter) is proposed for parameter-efficient Image-to-Video transfer learning. By freezing the pre-trained models and adding the lightweight adapters, we only need to update few parameters, which is highly efficient. In addition, extensive experiments on two video action recognition benchmarks show that our method can learn high-quality video spatio-temporal representations and achieve competitive or even better performance than prior works.

key words: *video action recognition, parameter-efficient image-to-video transfer learning, multi-scale spatio-temporal adapter, video spatio-temporal representations*

1. Introduction

As a prominent downstream task in computer vision, video action recognition has consistently been a prominent research focus. Over the past decade, with the continuous advancement of technology and the emergence of deep learning methods, substantial progress has been achieved in the domain of video action recognition. In the pre-deep learning era, most research was based on hand-crafted features [1], [2]. These methods often require extensive human intervention and are less effective in capturing complex behavioral patterns. With the development of deep learning techniques, the networks based on Convolutional Neural Networks[3]–[6] and Transformer[7]–[15] have witnessed great progress. Video understanding requires models to reason across multiple spatio-temporal resolutions, from fine-grained short-term actions to long-term actions. For longer videos, models with larger receptive fields are typically needed to capture long-range spatio-temporal dependencies, while shorter videos can be processed with smaller receptive fields. This has been shown to be effective in several prior works on multi-scale spatio-temporal modeling [13], [14]. Based on this, we believe that efficiently modeling multi-scale spatio-temporal information is a key factor in improving performance.

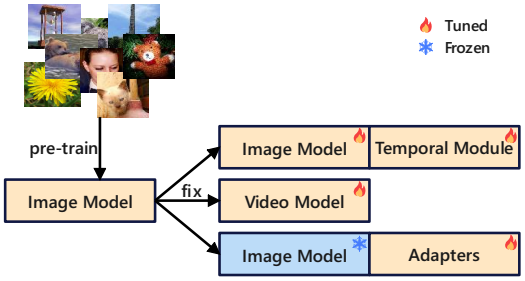


Fig. 1 The existing methods mainly include the following three approaches: directly inserting time modules into image models, constructing video models, and freezing image models while training only the adapters.

To model multi-scale spatio-temporal information, previous works have explored various approaches. In the domain of CNNs, many works model multi-scale information using pyramid structures[16], such as SlowFast[13]. SlowFast introduces a novel network that combines fast and slow pathway for video action recognition. The slow pathway, applied to low frame rates, captures spatial semantic information. The fast pathway, applied to high frame rates, provides excellent temporal resolution to capture motion information. And in the domain of transformers, several methods have been proposed to model multi-scale features, such as MViT[10] and MTV[14]. MViT employs a multi-scale hierarchical modeling technique within the transformer architecture. Specifically, the model incorporates pooling operations on the tensor after it is processed into query, key, and value (QKV) components. By reducing the sequence length, this operation effectively lowers the resolution, resulting in the construction of a multi-scale feature pyramid. MTV processes input videos through different 'views', with each 'view' being handled by a separate transformer encoder. Specifically, it processes the input video into tokens of different sizes by utilizing tubes of different scales. And then these tokens are fed into encoders of different scales to extract multi-scale spatio-temporal features. Meanwhile, lateral connections between encoders are utilized to fuse information from different 'views' to each other.

The aforementioned works and the preponderance of existing studies are predominantly based on introducing a new temporal module or modifying the architecture of image models to accommodate video action recognition task, as shown in Fig. 1 and these works are usually based on fully fine-tuning. Specifically, they use large-scale image

[†]The authors are with Hubei Key Laboratory of Intelligent Robot, the School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430205, China.
a) E-mail: 22207010113@stu.wit.edu.cn
b) E-mail: lutongwei@wit.edu.cn
c) E-mail: yaofeng@wit.edu.cn

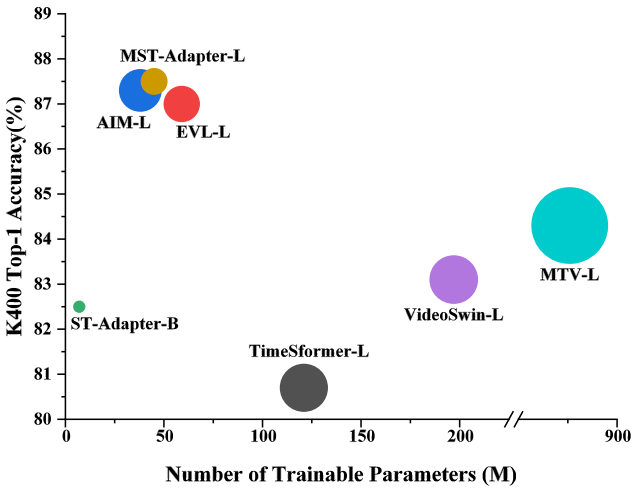


Fig. 2 We compare the performance of different methods on the K400 dataset. The size of bubbles represents the GFLOPs of inference time.

pre-trained models as the initialization of the model, and then retrain all parameters on video datasets. Although fully fine-tuning has resulted in enhanced performance, its computational and memory demands remain significantly high, particularly when handling large, high-quality video datasets such as Kinetics[17]. This poses a challenge for many researchers to thoroughly explore it.

To address the challenge of fully fine-tuning, parameter-efficient transfer learning (PETL) method is introduced in natural language processing (NLP)[18]–[21]. PETL provides an alternative to fully fine-tuning and facilitates the extension of large pre-trained models to downstream tasks. Specifically, PETL utilizes specific strategies to reduce computational complexity, such as adapter tuning[18], which freezing the pre-trained model and updating only the well-designed adapter. In this manner, during training and back-propagation, only a subset of the model's parameters is updated, resulting in a significant reduction in training costs.

With the introduction of PETL in the field of computer vision, it has found widespread application in downstream tasks. Several works[22]–[24] have demonstrated that PETL can replace fully fine-tuning in vision tasks, achieving superior performance with fewer training parameters. In the field of video action recognition, several works[25]–[28] also introduce PETL into model to reduce training costs and enhance accuracy. ST-Adapter[25] is a key work applying PETL to action recognition tasks. DiST[26] improves recognition performance in video tasks by separately addressing spatial and temporal dimensions. EVL[27] primarily uses a frozen pre-trained model with a carefully designed decoder to extract spatio-temporal information. AIM[28] introduces a lightweight Adapter into the model for fine-tuning. While these approaches focus on improving parameter efficiency for video action recognition tasks, these studies do not explore the modeling of multi-scale spatio-temporal information.

In this work, to reduce computational complexity while simultaneously extracting multi-scale spatio-temporal information, we introduce Multi-scaled Spatio-Temporal adapter (MST-adapter). By freezing the pre-trained image model during training and adding the MST-adapter, our results demonstrate that our approach can achieve competitive or even better results than previous state-of-the-art methods with fewer tunable parameters(Fig. 2). Specifically, we introduce two MST-Adapters to each encoder block, one after the multi-head self-attention (MHSA) and another after the Multilayer Perceptron (MLP). The MST-adapter consists of three branches designed to extract spatio-temporal information at different scales. By using this adapter, our model can learn spatio-temporal representations at each layer, enabling the network to acquire the capability of modeling the multi-scale spatio-temporal feature. In addition to adding adapter to our network, we introduce classification token shift multi-head self-attention([CLS] Token Shift MHSA) into our network to better extract features between adjacent frames with zero additional cost. With our careful design, our modifications can be easily integrated into fundamental image pre-trained models. Moreover, our model can achieve significant advantage with fewer training epochs.

Our main contributions can be summarized as follows:

1. We present a novel approach for fine-tuning image pre-trained models to video action recognition task. We introduce MST-Adapter and [CLS] token shift MHSA into our network to enhance features between adjacent frames.
2. Our method is more efficient than fully fine-tuning approaches. Compared to Timesformer[9], our method achieves a 6% increase in top-1 accuracy while utilizing only 37% of its trainable parameters.
3. We conduct extensive experiments on datasets with diverse features to demonstrate the effectiveness of our method. And compared to prior works, we achieve competitive or even better performance.

2. Related Work

Pre-trained vision models. Recently, the successful performance of Vision Transformers(ViTs) and its variants[29]–[31] on image recognition tasks has garnered significant attention. These well-performing models, trained on large datasets[32], can serve as excellent initialization for transfer learning on downstream tasks. However, collecting videos remains relatively expensive compared to images, and training a video model requires more computational power. In this work, we adopt well-trained image pre-trained model as initialization and focus on transfer image model to video model.

Video action recognition. The evolution of video action recognition methodologies has witnessed a paradigm shift from hand-crafted feature extraction, through convolutional neural networks, to transformer architectures.

In the period of hand-crafted feature extraction, most methods require human participation, such as IDT[1], [2].

IDT use optical flow[33] to obtain trajectories in video sequences, and then extract features along these trajectories. With the development of deep learning, many excellent methods of convolutional neural networks have emerged, which can be roughly divided into two ways: two-stream network[3]–[5], [34] and 3D CNN-based network[13], [35]. Two-stream networks use another independent stream to process data modality that can represent action information, such as optical flow. 3D CNN-Based methods extend the modeling capabilities of 2D convolution in the time dimension. Transformer-based methods typically incorporate attention mechanisms[9], [12], [36] or modify part of the encoder to model spatio-temporal information[10], [14], [37]. For example, TimeSformer[9] splits self-attention into spatial attention and temporal attention, not only reducing computational cost but also enhancing recognition performance. TPS[37] use patch shift in temporal dimension to model temporal information.

These methods all employ fully fine-tuning on pre-trained image models. Despite yielding impressive results, this approach still exhibits inefficiencies.

Parameter-efficient transfer learning(PETL). Since pre-trained language models are computationally expensive for fully fine-tuning in downstream tasks, some PETL methods have first been applied to NLP[18]–[21]. These methods adopt a lightweight design, aiming to achieve results comparable to fully fine-tuning by training only a small subset of parameters. The mainstream existing methods include: adapter tuning, prefix tuning, and prompt tuning.

Adapter tuning[18] is achieved by designing a lightweight adapter module and embedding it into transformer. During training, the parameters of the pre-trained model are frozen and only the adapter module is fine-tuned. Experiments show that this method trains only 3.6% of the parameters of the pre-trained model but achieves results close to fully fine-tuning.

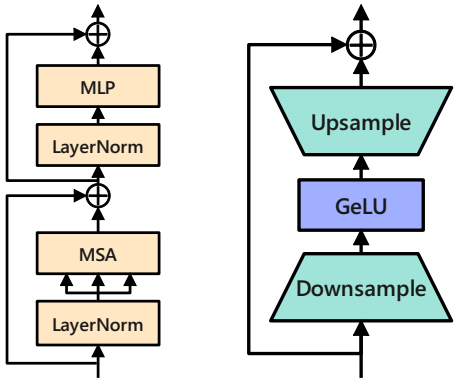
Prefix tuning[19] adds virtual tokens which related to a specific task before the model input. During training, like adapter tuning, it only update the parameters of prefix. This method implicitly learn information about downstream tasks and achieves results close to fully fine-tuning.

Prompt tuning[20] adds prompt tokens to the input layer. Through careful design, prompt tokens can contain task-related context to help the model better understand the requirements and generate the correct output. Similar to the prefix tuning method, except that prompt learns information about downstream tasks in an explicit way.

Existing works have confirmed that this idea also perform well in computer vision tasks[22]–[24], [38], [39]. In our work, we utilize adapter tuning in our model and design a multi-scale adapter module for concise and efficient spatio-temporal modeling.

3. Methodology

In this section, we initially outline the preliminaries of our work: ViT, ViT for video and the adapter (Section 3.1).



(a) ViT block (b) Adapter

Fig. 3 The structure of ViT block and adapter.

Subsequently, we provide an overview of our model (Section 3.2). Following that, we present a detailed description of the MST-Adapter and [CLS] token shift MHSA(Section 3.3-3.5). Finally, we introduce variants of the MST-Adapter by inserting it into different positions (Section 3.4).

3.1 Preliminary

3.1.1 Vision transformer

Since the introduction of vision transformer(ViT)[29] in computer vision, it has demonstrated promising performance across various downstream tasks. Here, we provide a brief description on how ViT works in computer vision.

Generally, ViT consists of linear projection, encoder, and MLP head. A standard ViT encoder typically comprises 12 blocks. The structure of the block is illustrated in Fig. 3(a). For the given images $I \in \mathbb{R}^{B\times H\times W\times C}$, here, B is the batch size, (H, W) is the resolution of the image and C is the number of channels in image. In ViT, the input image is divided into N non-overlapping patches, each of size (PxP). These patches are transformed into tokens by a linear projection, and a learnable class token z_{cls} is added, forming the token set $Z = [z_{cls}, z_0, z_1, ...z_i, ..., z_{N-1}]$. To encode positional information, positional embeddings p are added to each token z_i resulting in $x_i = z_i + p$. The final token representation is $Z_N \in \mathbb{R}^{B\times(N+1)\times D}$, where N is the number of the tokens, D is the token dimension, such that:

$$Z_N = [x_{cls}, x_0, x_1, \ldots\ldots, x_{N-1}] + p \tag{1}$$

And then we feed the obtained $N+1$ tokens into transformer consisting of L encoder blocks. The details are as follows:

$$y = z^{l-1} + \mathrm{MHSA}(\mathrm{LN}(z^{l-1}))) \tag{2}$$

$$z^l = y + \mathrm{MLP}(\mathrm{LN}(y)) \tag{3}$$

where z^{l-1} and z^l denotes the input and output of the l-th transformer block, LN is LayerNorm[40], MHSA denotes multi-head self-attention and MLP stands for Multilayer Perceptron, typically consists of two linear projection layers and

an activation function. Then, we obtain the class token for visual representation of the image. Finally, with a linear classifier, z^l is mapped to one of the classes.

3.1.2 Vision transformer for video

Similar to TimeSformer[9], for a given video containing T input frames $\mathrm{F} \in \mathbb{R}^{B\times T\times H\times W\times C}$, we apply the same method to process each frame into non-overlapping patches, and then tokenize them into tokens. Then we get our video tokens $\mathrm{V} \in \mathbb{R}^{BT\times(N+1)\times D}$, which are composed of tokens for each frame(Eq.(1)).

$$\mathrm{V} = [Z^0, Z^1, ..., Z^{t-1}] \tag{4}$$

where Z^t represents the token of t-th frame. Subsequently, we feed these tokens into the transformer architecture for training, similar to Eqs.(2) and (3). After obtaining the [CLS] token of each frame, we calculate the average of these [CLS] tokens to get the feature representation of the entire video. We pass this aggregated representation through the fully connected layer (FC layer) to obtain the final classification accuracy.

For a more detailed temporal modeling within the model, the mainstream approach typically develops spatio-temporal modeling by incorporating temporal processing modules or introducing temporal attention within the network, and then conducting a fully fine-tuning of the entire network model.

3.1.3 Adapter

To transfer a pre-trained image model to the video domain and address the computational cost associated with fully fine-tuning, inspired by the latest work on PETL[18], we introduce adapters into our network. Specifically, adapter consists of a down-projection linear layer, a non-linear activation function and an up-projection linear layer, as illustrated in Fig. 3(b). For a given input token $X \in \mathbb{R}^{B\times(N+1)\times D}$, the process can be written as follows:

$$\begin{aligned}\mathrm{Adapter}(X) = X + \mathrm{upProjection}(\mathrm{Activation}(\\ \mathrm{downProjection}(X)))\end{aligned} \tag{5}$$

Existing works indicate that adapters can achieve performance comparable due to the following advantages: (1) **Training speed:** The introduction of adapters allows the model to update only a very small number of parameters during training, significantly speeding up the process; (2) **Superior performance:** The adapter exhibits superior performance, surpassing even fully fine-tuning in some tasks; (3) **Preventing forgetting problem:** With the introduction of adapters, the majority of parameters in the pre-trained model are frozen, mitigating the potential catastrophic forgetting problem arising from continuous learning, enhancing the model's robustness across various tasks.

We incorporate this idea into video action recognition, enhancing its ability for spatio-temporal reasoning through careful design.

3.2 Overview of our model

Our motivation is to train only a subset of parameters to achieve efficient performance on video action recognition task. To this end, we adopt the adapter-tuning method and select CLIP as our backbone. Within our model, we leverage CLIP's robust spatial information representation while enhancing its ability to extract multi-scale spatio-temporal information by introducing the MST-Adapter. By freezing the pre-trained model and exclusively training the adapter's parameters, we not only improve video action recognition performance but also significantly reduce memory footprint. The overall network framework is illustrated in Fig. 4.

Input: The video consists of numerous frames, and when processed into individual frames, we observe a notable similarity between adjacent ones, making it challenging to extract action information. To address this, following TimeSformer[9], we sample the video at intervals of 16, and 8, resulting in sequences of 8 and 16 frames. These sequences serve as the input for our network.

Backbone: We employ CLIP[42] as our pre-trained model, leveraging its robust spatial representation, gained from training on an extensive dataset of 400 million image-text pairs.

Adapter: When designing the adapter, we extend the standard adapter framework(Fig. 3(b)) into a multi-scale extractor to capture multi-scale spatio-temporal information.

3.3 Multi-scale Spatio-Temporal Adapter (MST-Adapter)

In this section, we propose the Multi-Scale Spatio-Temporal Adapter (MST-Adapter) to enable image pre-trained models to better model the spatio-temporal information of videos. The design details are as follows:

As illustrated in Fig.4(right), the features are first downsampled into a low-dimensional feature space, and the feature representation is reshaped from $V \in \mathbb{R}^{BT\times(N+1)\times C}$ to $V_{down} \in \mathbb{R}^{B\times T\times H\times W\times C}$ to prepare the spatial and temporal dimensions for reasoning. These features are then processed by our multi-scale extraction module to capture multi-scale spatio-temporal information. Finally, the features are upsampled back to their original size. The MST-Adapter can be expressed as:

$$\begin{aligned}\text{MST-Adapter}(V) = V + \mathrm{upProjection}(\mathrm{ME}(\\ \mathrm{downProjection}(V)))\end{aligned} \tag{6}$$

where downProjection is the down projection layer, ME denotes the multi-scale extraction and upProjection means the up projection layer.

In multi-scale extraction, as shown in Fig. 4(Right). we use three branches to model spatio-temporal information at different scales.Following the approach of SlowFast[13], we employ a 3D Convolutional Neural Network (CNN) to extract spatio-temporal information from video data. Unlike

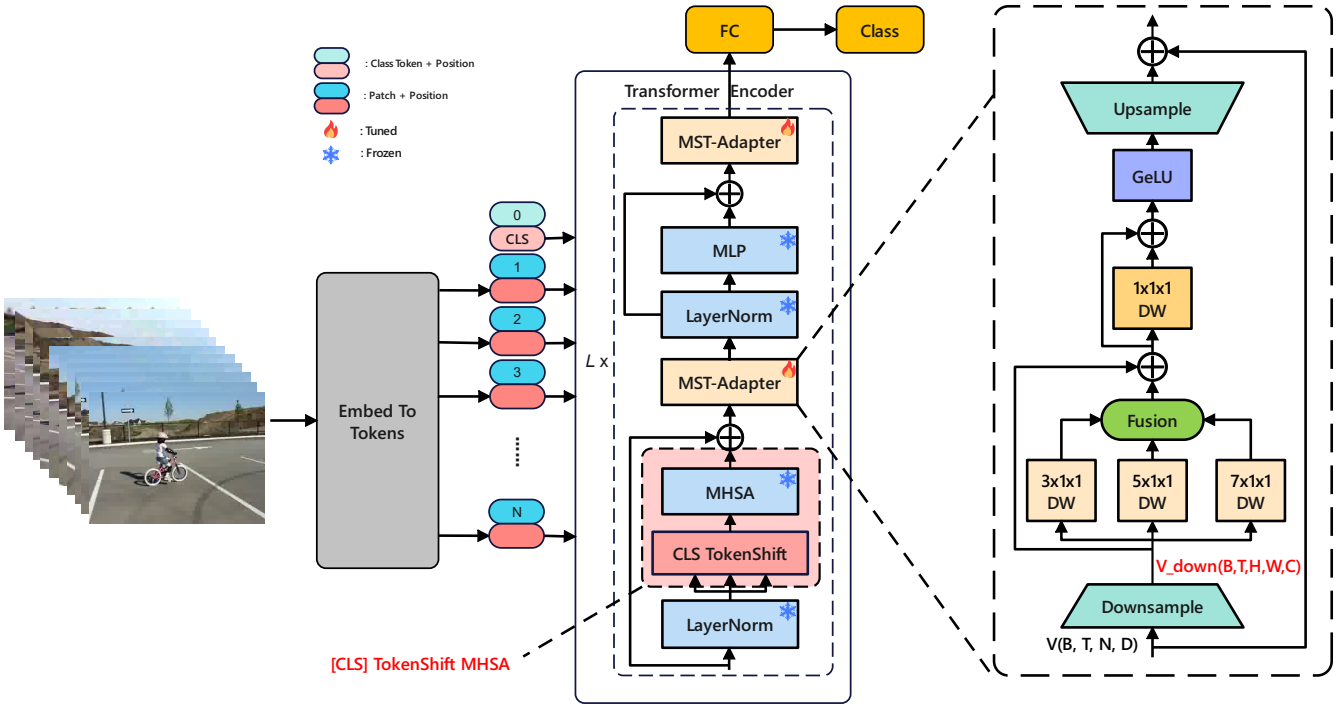


Fig. 4 The overall framework of our model. We learn multi-scale spatio-temporal contexts through MST-Adapter. During training, we only update the parameters of the MST-Adapter.

2D CNNs, which only capture spatial features within individual frames, 3D CNNs convolve across both spatial and temporal features simultaneously by convolving across both spatial dimensions (H and W) and the temporal dimension (T), allowing them to model dynamic changes and temporal relationships between frames. In each branch of the MST-Adapter, we introduce a spatio-temporal operator realized by a 3D depth-wise convolution layer[43]. Here we adopt depth-wise convolution instead of standard convolution to reduce the number of parameters and enhance the network's representation capacity with minimal overhead. Subsequently, in the fusion stage, we propose two fusion strategies: direct averaging and utilizing learnable parameters. Through extensive experiment, we discover that utilizing learnable parameters as fusion weights delivers the best top-1 accuracy. Finally, we apply a 1x1x1 3D convolution with residual connections to further enhance the model's spatio-temporal modeling ability. It can be expressed as follows:

$$B_i = DWConv3D_i(V_{down}) \tag{7}$$

$$z = X_{down} + \sum_{i=1}^{3} W_i B_i \tag{8}$$

$$y = z + DWConv3D_{1x1x1}(z) \tag{9}$$

where B_i, z, y denotes the i-th branch, the output after being processed by three branches, the output of the multi-scale extraction, X_{down} is the feature after being processed by downProjection(Eq.6). $DWConv3D_i$ is the depth-wise 3D convolution of the i-th branch, W_i is the weight of i-th branch. $DWConv3D_{1x1x1}$ is the 1x1x1 depth-wise 3D convolution.

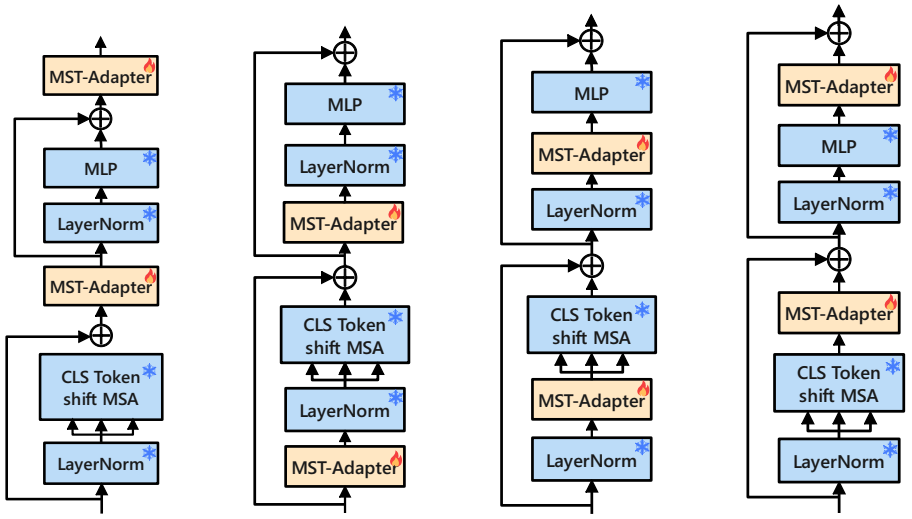
We conduct experiments to evaluate the fusion strategies for the three branches and access the performance of the 1x1x1 depth-wise 3D convolution. The detailed results and analysis are presented in Section 4.

3.4 Integration into ViT Encoder

In the ViT encoder, the placement of the MST-Adapter significantly affects the model's ability to learn spatio-temporal information. Therefore, conducting research on this matter is essential. Based on previous experience, we propose four different variants, as shown in Fig. 5. This includes "post residual" (inserted after residual), "prior layer norm"(inserted before layer norm), "prior MSA/MLP" (inserted before the Multi-Head Self-Attention and Multi-Layer Perceptron) and "post MSA/MLP" (inserted after the Multi-Head Self-Attention and Multi-Layer Perceptron). We experimented extensively with these variants, and found that the "post residual" variant delivers the best performance.

3.5 [CLS] Token Shift MHSA

To better model the spatio-temporal information between frames, we introduce the [CLS] token shift multi-head self-attention into our model. While the [CLS] token captures global spatial information after transformer processing, it lacks temporal relationships with surrounding frames. To address this, we use a shift operation to enhance its temporal information. Follow previous works[44], we integrate [CLS] token shift operation into our model. This operation shifts the whole pre-trained [CLS] token channels back-and-forth



(a) MST-Adapter (b) MST-Adapter-A (c) MST-Adapter-B (d) MST-Adapter-C

Fig. 5 By inserting the MST-adapter into the ViT encoder block, we have obtained several variants: "post residual"(MST-Adapter), "prior layer norm"(MST-Adapter-A), "prior MSA/MLP"(MST-Adapter-B) and "post MSA/MLP"(MST-Adapter-C).

across adjacent frames. In this case, MST-Adapter learn spatio-temporal information in [CLS] tokens with zero parameters, zero FLOPs, and no extra computational overhead.

4. Experiments

4.1 Experiments Setup

Datasets: For the experiments, we use two popular video action recognition datasets:

Kinetics400(K400)[17] is a large-scale video action recognition dataset with about 240K training videos and 20K validation videos for 400 action classes. Each video is about 10s and is derived from YouTube, covering various daily action scenes.

Something-SomethingV2(SSV2)[46] consists of 174 classes and contains about 169k training videos and 24k validation videos. This dataset contains a rich set of gestures and motions, which requires more powerful temporal modeling capabilities than the K400 dataset.

Pre-Trained model: We adopt two variants of CLIP as our pre-trained model, VIT-B/16 and VIT-L/14. ViT-B/16 is a medium-sized Vision Transformer model, comprising a total of 12 blocks. It processes input images into tokens with a dimensionality of 768 using a patch size of 16x16. ViT-L/14 is a larger-scale Vision Transformer model with a total of 24 blocks. It processes input images into tokens with a dimensionality of 1024 using a patch size of 14x14.

Implementation details: We conduct all experiments on 4 Tesla T4 and RTX8000 GPUs. We implement MST-Adapter using PyTorch, building upon the existing codebase of MMAction2[47].

We employ dense sampling to extract a certain number of frames from videos to construct input clips. Subsequently, we apply random scaling to the video frames within the range of (256, 320), followed by random cropping to resize each frame to 224×224 pixels and perform random flipping with a probability of 0.5. We utilize the AdamW optimizer[48] with a momentum of (0.9, 0.999) and a weight decay of 0.05. By default, we adopt cosine annealing learning rate scheduling[49] and perform a 5-epoch warm-up period before training for a total of 30 epochs. The default base learning rate is 1.5e-04. The batch size per GPU is set to 4.

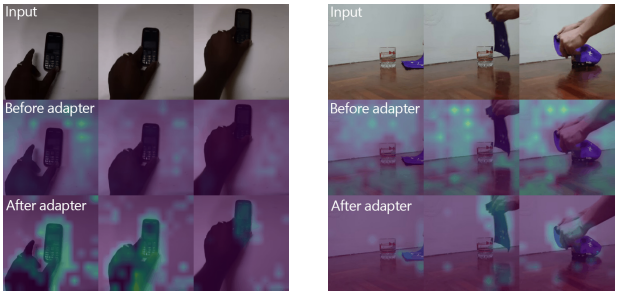
4.2 Comparisons To The-state-of-the-art

We conduct comprehensive experiments on two benchmark datasets and compare them with state-of-the-art methods. For each method, we report its total number of parameters, tunable parameters, GFLOPs, Top-1, and Top-5 accuracy. In the experimental setup, we conducted experiments using only 8 and 16 frames to demonstrate the effectiveness of our method. Increasing the number of frames could enhance the model's ability to capture more detailed spatio-temporal information, potentially leading to improved accuracy in video action recognition tasks. However, it is important to note that using more frames would also increase computational load and memory requirements, which could be a limiting factor in practical applications. Due to our computational resource constraints, we focused on experiments with 8 and 16 frames.

Results on K400 We present performance comparisons on K400[24] in Table 1. Firstly, we compare our results with direct fine-tuning of the pure CLIP model on K400, demonstrating that integrating our MST-Adapter into CLIP effectively enhances the model's spatio-temporal modeling capability and achieves higher accuracy with fewer training parameters. When applying the input of 8-frames and ViT-B/16 model, compared to TimeSformer-L, we achieve better performance (82.7% vs 80.7%) with fewer trainable parameters (18M vs 121M) and less computations cost in GFLOPs(348 vs 7140 GFLOPs). And compared with other fully fine-

Table 1 Performance comparisons of video action recognition on the Kinetics-400. Note that Views = #frames × #clips × #spatial.

Methods	Pretrain	Params (M)	Tunable Params(M)	GFLOPs	Top-1	Top-5	Views
Fully Fine-tuning							
SlowFast[13]	-	60	60	7020	79.8	93.9	16x3x10
TimeSfomrer-L[9]	IN-21K	121	121	7140	80.7	94.7	96x3x1
MViT-B[10]	-	37	37	4095	81.2	95.1	64x3x3
MTV-B[14]	JFT	310	310	4790	81.8	95.2	32×3×4
ViViT-L[50]	IN-1K	311	311	3980	80.6	92.7	32x1x1
UniFormer-B[51]	IN-1K	50	50	3108	83.0	95.4	32×1×4
VideoSwin-L[12]	IN-21K	197	197	7248	83.1	95.9	32x3x1
Frozen backbone							
ST-Adapter [25]ViT-B/16	CLIP	93	7	455	82.5	96.0	16x3x1
EVL [27]ViT-L/14	CLIP	368	59	4044	87.0	-	16x3x1
AIM [28]ViT-L/14	CLIP	341	38	5604	87.3	97.6	16x3x1
CLIP* ViT-B/16	CLIP	86	86	419	77.3	90.3	8x3x1
MST-Adapter ViT-B/16	CLIP	104	18	348	82.7	96.2	8x3x1
MST-Adapter ViT-B/16	CLIP	104	18	696	83.4	96.7	16x3x1
MST-Adapter ViT-L/14	CLIP	349	45	1116	86.7	97.3	8x3x1
MST-Adapter ViT-L/14	CLIP	349	45	2232	**87.5**	**97.6**	16x3x1



(a) Moving something down (b) Cover something with something

Fig. 6 We use GradCam to visualize our network. With the help of our MST-Adapter, our network is able to better focus the model's attention on the action.

tuning methods, which are pretrained on large-scale image datasets and then fully finetuned on K400, our approach achieves better performance than most prior works. When we apply ViT-L/14, we achieve 87.5% top-1 accuracy with 45M parameters and 2232 GFLOPs. Compared to methods also using the parameter efficient strategy, our method achieves competitive performance with less computational cost under similar conditions.

Result on SSV2 We summarize our results on SSV2 in Table 2. Similarly, compared to the pure CLIP model, our approach achieves an improvement (44.5%→65.3%) in Top-1 accuracy by utilizing the ViT-B/16 pretrained model with the input of 8-frames. Compared to TimeSformer-L, our method exhibits better performance(65.3% vs 62.4%) with fewer trainable parameters(18M vs 121 M). Furthermore, when compared to EVL which also freezes the backbone network, our approach outperforms it under similar conditions. Additionally, when employing the ViT-L/14 model with the input of 16-frames, we achieve a Top-1 accuracy of 69.7% with only 45M trainable parameters, surpassing most prior works. And we visualize our method on some of the actions in the SSV2, as shown in Fig. 6. We observe that through our MST-Adapter, the network can focus more on the temporal information in videos.

However, our approach slightly falls behind some fully fine-tuning methods, mainly due to two reasons. Firstly, SSV2 is a dataset with complex spatio-temporal relations, containing numerous subtle actions and temporal nuances. This requires the model to effectively understand the temporal relationships between frames in the video. On the other hand, most fully fine-tuning methods are initially trained on large-scale video datasets such as Kinetics-400, then fine-tuned on the SSV2. In contrast, our method is pretrained on image datasets and subsequently fine-tuned on the SSV2, achieving competitive performance with lower computational costs.

4.3 Ablation study

Component of the MST-Adapter: We conduct ablation experiments on the components of MST-Adapter on the Something-SomethingV2 and Kinetics400 datasets, primarily to verify the effects of inserting 1x1x1 depth-wise convolution and two different fusion methods. The results are shown in Table 3. Firstly, we observed that adding 1x1x1 depth-wise convolution enhance the model's representation ability for spatio-temporal information, resulting in a 2% improvement in top-1 accuracy with an increase of 4M parameters. Next, we compare the effects of these two different fusion methods. Here, we initialize the learnable parameters to 0.33. According to the results, we find that employing learnable parameters to learn the weights of the three branches can achieve better results than direct averaging on three branches. Furthermore, using learnable parameters results in an increase of only 72 parameters, which is practically negligible.

Fusion Weights: To further investigate the role of each branch at different stages, we visualized the weights of three branches, as shown in the Fig. 7. We find that the weights of the large size branch are higher than those of the small size branch in layers 1 to 8. However, in layers 9 to 12, the situation is reversed.

Table 2 Performance comparisons of video action recognition on the SSV2. Note that Views = #frames × #clips × #spatial. † means the model is pre-trained on both IN-21K and K400/K600.

Methods	Pretrain	Params (M)	Tunable Params(M)	GFLOPs	Top-1	Top-5	Views
SlowFast[13]	-	60	60	7020	63.1	87.6	16x1x3
TimeSfomrer-L[9]	IN-21K	121	121	7140	62.4	-	64x1x3
MViT-B[10]	K400	37	37	510	67.1	90.8	32x1x3
MTV-B[14]	K400	310	310	4790	67.6	90.4	32x4x3
ViViT-L[50]	K400†	311	311	11892	65.4	89.8	16x4x3
MViTv2-L[11]	K400†	51	51	675	**73.3**	**94.1**	40x1x3
Uniform-B[51]	K600†	50	50	777	71.2	92.8	32x1x3
VideoSwin-B[12]	K400†	89	89	963	69.6	92.7	32x1x1
Frozen backbone							
ST-Adapter [25]ViT-B/16	CLIP	93	7	977	69.3	92.3	16x1x3
EVL [27]ViT-L/14	CLIP	484	174	4044	66.7	-	32x1x3
AIM [28]ViT-L/14	CLIP	354	50	5754	69.4	92.3	16x1x3
CLIP* ViT-B/16	CLIP	86	86	419	44.5	76.3	8x1x3
MST-Adapter ViT-B/16	CLIP	104	18	348	65.3	90.2	8x1x3
MST-Adapter ViT-B/16	CLIP	104	18	696	66.1	91.3	16x1x3
MST-Adapter ViT-L/14	CLIP	349	45	1116	68.3	92.3	8x1x3
MST-Adapter ViT-L/14	CLIP	349	45	2232	69.7	92.5	16x1x3

Table 3 Performance comparisons of components of the MST-Adapter.

1x1x1	fusion	Tunable Param	SSV2 Top-1	K400 Top-1
	avg	14M	63.0	80.4
✓	avg	18M	65.1	82.6
	lp	14M	63.4	81.0
✓	lp	18M	**65.3**	**82.7**

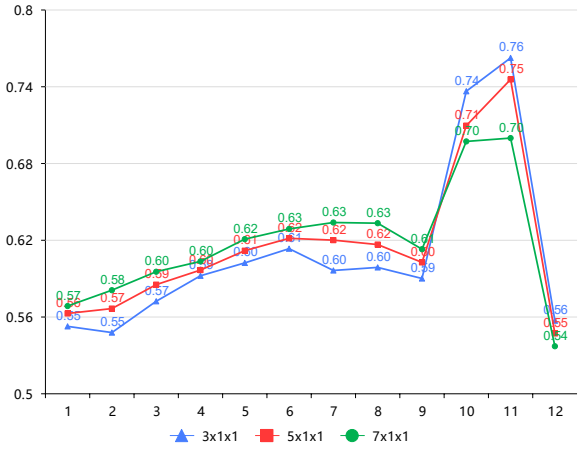


Fig. 7 Visualization of Fusion Weights.

Through visual analysis, we believe that in the early layers, the network primarily relies on the large-size branch to extract coarse spatio-temporal information, thereby obtaining general action classification insights. And in the deeper layers, the network shifts its focus to the small-size branch, extracting more nuanced spatio-temporal features for finer-grained action classification. This is consistent with the conclusions of previous studies, such as the results in MTV[14]. This finding helps explain how the model utilizes branches at different stages, providing strong support for a deeper understanding.

[CLS] Token Shift MHSA: In the Transformer model, the [CLS] token contains spatial features for each frame. We perform a shift operation on the [CLS] token to enable the model to learn features from adjacent frames. The results are shown in Table 4. We found that applying [CLS] MHSA to the SSV2 dataset and the K400 dataset resulted in accuracy improvements of 5% and 7%, respectively. This demonstrates the effectiveness of [CLS] token shift MHSA in enhancing the model's temporal reasoning ability without incurring any additional computational costs.

Table 4 Effect of whether use [CLS] Token Shift MHSA. Here, 'w/o' denotes 'without,' and 'w/' denotes 'with' in the table.

Model variants	Tunable Param	SSV2 Top-1	K400 Top-1
MST-Adapter w/o shift	18M	65.1	82.0
MST-Adapter w/ shift	18M	**65.3**	**82.7**

Table 5 Effect of position of MST-Adapter in encoder.

Model	Tunable Param	SSV2 Top-1	K400 Top-1
CLIP(fully fine-tuned)	86M	44.5	77.3
MST-Adapter-A	18M	64.7	81.5
MST-Adapter-B	18M	63.8	80.1
MST-Adapter-C	18M	64.6	81.9
MST-Adapter	18M	**65.3**	**82.7**

Which position to insert MST-Adapter into Encoder: We introduce four different models by inserting the MST-Adapter into the encoder, as shown in Fig. 5, and conduct detailed experiments. We use CLIP as our baseline, which is the case when the MST-Adapter module is not inserted. The experimental results are shown in Table 5.

We observe that: (1) No matter where it is inserted, our MST-Adapter module consistently improves the performance of the baseline, indicates that our MST-Adapter module can effectively model the spatio-temporal information of the video. (2) Placing the module after the residual connection achieves the best top-1 accuracy.

Which layer to adopt MST-Adapter: By default, we insert MST-Adapter into every layer of the network. Here, we explore the effect of reducing the number of MST-Adapters on performance. We use ViT-B/16 as our backbone and divide its 12 layers into three stages: early (layers 1-4), mid

Table 6 Effect of position of MST-Adapter in layers.

1-4	5-8	9-12	Tunable Param	SSV2 Top-1	K400 Top-1
✓			6M	55.9	75.5
	✓		6M	58.9	76.9
		✓	6M	60.5	78.0
✓	✓		12M	62.7	80.3
	✓	✓	12M	63.3	82.1
✓	✓	✓	18M	**65.3**	**82.7**

(layers 5-8), and late (layers 9-12), to explore the model's performance at different stages. As shown in Table 6, we find that as more MST-Adapters are applied to the network, the model's performance gradually improves. Moreover, the effectiveness of applying MST-Adapter in deeper layers is significantly better than in early layers.

5. Conclusion

In this paper, we propose a novel approach to effectively transfer pretrained image models to video action recognition. We introduce a multi-scale spatio-temporal adapter and [CLS] token shift operation into our model. With only a few parameter updates required, our model incurs lower computational costs compared to fully fine-tuning models. Extensive experiments on datasets with diverse features demonstrate the effectiveness of our approach in enhancing the model's representation of spatio-temporal information. Moreover, our method achieves competitive or even better performance compared to prior works on a broad range of popular video benchmarks.

Acknowledgment

The authors would like to thank all the people who have contributed to this paper for their selfless work. This work is supported by the National Natural Science Foundation of China (62171327, 62171328, 62072350), the Hubei Technology Innovation Project (2019AAA045).

References

[1] H. Wang and C. Schmid, "Action recognition with improved trajectories," Proceedings of the IEEE international conference on computer vision, pp.3551–3558, 2013.

[2] H. Wang, A. Kläser, C. Schmid, and C.L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," International journal of computer vision, vol.103, pp.60–79, 2013.

[3] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," Proceedings of the IEEE conference on computer vision and pattern recognition, pp.1933–1941, 2016.

[4] C. Feichtenhofer, A. Pinz, and R.P. Wildes, "Spatiotemporal multiplier networks for video action recognition," Proceedings of the IEEE conference on computer vision and pattern recognition, pp.4768–4777, 2017.

[5] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," European conference on computer vision, pp.20–36, Springer, 2016.

[6] Y. Wang, M. Long, J. Wang, and P.S. Yu, "Spatiotemporal pyramid network for video action recognition," Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp.1529–1538, 2017.

[7] D. Neimark, O. Bar, M. Zohar, and D. Asselmann, "Video transformer network," Proceedings of the IEEE/CVF international conference on computer vision, pp.3163–3172, 2021.

[8] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp.244–253, 2019.

[9] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?," ICML, p.4, 2021.

[10] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," Proceedings of the IEEE/CVF international conference on computer vision, pp.6824–6835, 2021.

[11] Y. Li, C.Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, and C. Feichtenhofer, "Mvitv2: Improved multiscale vision transformers for classification and detection," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp.4804–4814, 2022.

[12] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu, "Video swin transformer," Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp.3202–3211, 2022.

[13] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," Proceedings of the IEEE/CVF international conference on computer vision, pp.6202–6211, 2019.

[14] S. Yan, X. Xiong, A. Arnab, Z. Lu, M. Zhang, C. Sun, and C. Schmid, "Multiview transformers for video recognition," Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp.3333–3343, 2022.

[15] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp.591–600, 2020.

[16] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, vol.25, 2012.

[17] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, *et al.*, "The kinetics human action video dataset," arXiv preprint arXiv:1705.06950, 2017.

[18] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," International Conference on Machine Learning, pp.2790–2799, PMLR, 2019.

[19] X.L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," arXiv preprint arXiv:2101.00190, 2021.

[20] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," arXiv preprint arXiv:2104.08691, 2021.

[21] E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," arXiv preprint arXiv:2106.09685, 2021.

[22] S. Chen, C. Ge, Z. Tong, J. Wang, Y. Song, J. Wang, and P. Luo, "Adaptformer: Adapting vision transformers for scalable visual recognition," Advances in Neural Information Processing Systems, vol.35, pp.16664–16678, 2022.

[23] X. He, C. Li, P. Zhang, J. Yang, and X.E. Wang, "Parameter-efficient model adaptation for vision transformers," arXiv preprint arXiv:2203.16329, 2022.

[24] M. Jia, L. Tang, B.C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.N. Lim, "Visual prompt tuning," European Conference on Computer Vision, pp.709–727, Springer, 2022.

[25] J. Pan, Z. Lin, X. Zhu, J. Shao, and H. Li, "St-adapter: Parameter-efficient image-to-video transfer learning," Advances in Neural Information Processing Systems, vol.35, pp.26462–26477, 2022.

[26] Z. Qing, S. Zhang, Z. Huang, Y. Zhang, C. Gao, D. Zhao, and

N. Sang, "Disentangling spatial and temporal learning for efficient image-to-video transfer learning," Proceedings of the IEEE/CVF International Conference on Computer Vision, pp.13934–13944, 2023.

[27] Z. Lin, S. Geng, R. Zhang, P. Gao, G. de Melo, X. Wang, J. Dai, Y. Qiao, and H. Li, "Frozen clip models are efficient video learners," European Conference on Computer Vision, pp.388–404, Springer, 2022.

[28] T. Yang, Y. Zhu, Y. Xie, A. Zhang, C. Chen, and M. Li, "Aim: Adapting image models for efficient video action recognition," arXiv preprint arXiv:2302.03024, 2023.

[29] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," arXiv preprint arXiv:2010.11929, 2020.

[30] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," Proceedings of the IEEE/CVF international conference on computer vision, pp.10012–10022, 2021.

[31] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, *et al.*, "Swin transformer v2: Scaling up capacity and resolution," Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp.12009–12019, 2022.

[32] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," 2009 IEEE conference on computer vision and pattern recognition, pp.248–255, Ieee, 2009.

[33] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l 1 optical flow," Pattern Recognition: 29th DAGM Symposium, Heidelberg, Germany, September 12-14, 2007. Proceedings 29, pp.214–223, Springer, 2007.

[34] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," Advances in neural information processing systems, vol.27, 2014.

[35] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," Proceedings of the IEEE international conference on computer vision, pp.4489–4497, 2015.

[36] Y. Liang, P. Zhou, R. Zimmermann, and S. Yan, "Dualformer: Local-global stratified transformer for efficient video recognition," European Conference on Computer Vision, pp.577–595, Springer, 2022.

[37] W. Xiang, C. Li, B. Wang, X. Wei, X.S. Hua, and L. Zhang, "Spatiotemporal self-attention modeling with temporal patch shift for action recognition," European Conference on Computer Vision, pp.627–644, Springer, 2022.

[38] C. Wei, H. Fan, S. Xie, C.Y. Wu, A. Yuille, and C. Feichtenhofer, "Masked feature prediction for self-supervised visual pre-training," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp.14668–14678, 2022.

[39] Y.L. Sung, J. Cho, and M. Bansal, "Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp.5227–5237, 2022.

[40] J.L. Ba, J.R. Kiros, and G.E. Hinton, "Layer normalization," arXiv preprint arXiv:1607.06450, 2016.

[41] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," arXiv preprint arXiv:1606.08415, 2016.

[42] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," International conference on machine learning, pp.8748–8763, PMLR, 2021.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," Proceedings of the IEEE conference on computer vision and pattern recognition, pp.1–9, 2015.

[44] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," Proceedings of the IEEE/CVF international conference on computer vision, pp.7083–7093, 2019.

[45] H. Zhang, Y. Hao, and C.W. Ngo, "Token shift transformer for video classification," Proceedings of the 29th ACM International Conference on Multimedia, pp.917–925, 2021.

[46] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, *et al.*, "The" something something" video database for learning and evaluating visual common sense," Proceedings of the IEEE international conference on computer vision, pp.5842–5850, 2017.

[47] M. Contributors, "Openmmlab's next generation video understanding toolbox and benchmark." `https://github.com/open-mmlab/mmaction2`, 2020.

[48] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," arXiv preprint arXiv:1711.05101, 2017.

[49] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," arXiv preprint arXiv:1608.03983, 2016.

[50] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," Proceedings of the IEEE/CVF international conference on computer vision, pp.6836–6846, 2021.

[51] K. Li, Y. Wang, P. Gao, G. Song, Y. Liu, H. Li, and Y. Qiao, "Uniformer: Unified transformer for efficient spatiotemporal representation learning," arXiv preprint arXiv:2201.04676, 2022.

Chenrui Chang is a graduate student in college of Computer Science and Engineering, Wuhan Institute of Technology, China. His research interests includes machine vision.

Tongwei Lu is an professor at Wuhan Institute of Technology, China. He received his Ph.D. degree from Huazhong University of Science and Technology, China. His research interests includes digital image processing, machine vision, image understanding and recognition.

Feng Yao is an associate professor at Wuhan Institute of Technology, China. His research interests includes technology for computer applications and intelligent computing.