Temporal Cross-attention for Action Recognition

Ryota Hashiguchi and Toru Tamaki[0000−0001−9712−7777]

Nagoya Institute of Technology, Nagoya, Japan
r.hashiguchi.651@nitech.jp, tamaki.toru@nitech.ac.jp

Abstract. Feature shifts have been shown to be useful for action recognition with CNN-based models since Temporal Shift Module (TSM) was proposed. It is based on frame-wise feature extraction with late fusion, and layer features are shifted along the time direction for the temporal interaction. TokenShift, a recent model based on Vision Transformer (ViT), also uses the temporal feature shift mechanism, which, however, does not fully exploit the structure of Multi-head Self-Attention (MSA) in ViT. In this paper, we propose *Multi-head Self/Cross-Attention* (MSCA), which fully utilizes the attention structure. TokenShift is based on a frame-wise ViT with features temporally shifted with successive frames (at time $t+1$ and $t-1$). In contrast, the proposed MSCA replaces MSA in the frame-wise ViT, and some MSA heads attend to successive frames instead of the current frame. The computation cost is the same as the frame-wise ViT and TokenShift as it simply changes the target to which the attention is taken. There is a choice about which of key, query, and value are taken from the successive frames, then we experimentally compared these variants with Kinetics400. We also investigate other variants in which the proposed MSCA is used along the patch dimension of ViT, instead of the head dimension. Experimental results show that a variant, MSCA-KV, shows the best performance and is better than TokenShift by 0.1% and then ViT by 1.2%.

1 Introduction

Recognizing the actions of people in videos is an important topic in computer vision. After the emergence of Vision Transformer (ViT) [1] which has been shown to be effective for various image recognition tasks [2, 3], research extending ViT to video has become active [4–8].

In recognition of video, unlike images, it is necessary to model the temporal information across frames of a given video clip. While most early works of action recognition were frame-wise CNN models followed by temporal aggregation [9, 10], 3D CNN models [11–13] were shown to be effective and well generalized when they were trained on large-scale datasets [14] and transferred to smaller datasets [15].

However, the computational cost of 3D convolution is usually high. To mitigate the issue of the trade-off between the ability of temporal modeling and the high computational cost, Temporal Shift Module (TSM) [16] was proposed by extending spatial feature shifting [17, 18] to the temporal dimension. TSM uses late fusion by applying 2D ResNet [19] to each frame, while it exploits the temporal interaction between adjacent frames by shifting the features in each layer of ResNet. The feature shift operation is computationally inexpensive, nevertheless it has been shown to perform as well as 3D CNN. Therefore, many related CNN models [20–22] have been proposed. TokenShift [23] is a TSM-like 2D ViT-based model that shifts a fraction of ViT features, showing that shifting only the class tokens is enough to achieve a good performance.

However, TokenShift doesn't fully exploit the advantage of the architecture of the attention mechanism of ViT, instead it simply shifts features like as in TSM. We propose a method that seamlessly utilizes the ViT structure to interact with features across adjacent frames. The original TSM or TokenShift uses a 2D model applied to each frame, while the output features of layers of the 2D model are then shifted from the current time t to one time step forward $t + 1$ and backward $t - 1$. In the encoder architecture of ViT, Multi-head Self-Attention (MSA) computes attentions between patches at a given frame t. The proposed method utilizes and modifies it for the temporal interaction; some patches at frame t attends to not only patches at the current frame t but also patches at the neighbor frames $t + 1$ and $t - 1$. We call this mechanisms *Multi-head Self/Cross-Attention* (MSCA). This allows the temporal interaction to be computed within the Transformer block without additional shift modules. Furthermore, the computational cost remains the same because some parts of self-attention computation is simply replaced with the temporal cross-attention. The contributions of this paper are as follows:

- We propose a new action recognition model with the proposed MSCA, which seamlessly combines the concepts of ViT and TSM. This avoids computing spatio-temporal attention directly in the 3D video volume.
- MSCA uses cross-attention to perform temporal interactions inside the Transformer block, unlike TokenShift that uses additional shifting modules. This allows for temporal interaction without increasing computational complexity nor changing model architecture.
- Experiments on Kinetics400 show that the proposed method improves performance compared to ViT and TokenShift.

2 Related Work

2.1 2D/3D CNN, and TSM

Early approaches of deep learning to action recognition are 2D-based methods [9, 10] that apply 2D CNN to videos frame by frame, or Two-Stream methods [24] that use RGB frames and optical flow stacks. However, it is difficult to model long-term temporal information beyond several time steps, and this is where 3D-based methods [11–13] came in. These models simultaneously consider spatial and temporal information, but their computational cost is relatively high compared to 2D-based methods. Therefore, mixed models of 2D and 3D convolution [25–27] have been proposed by separating convolution in the spatial and temporal dimensions.

TSM [16] is an attempt to model temporal information for action recognition without increasing parameters and computational complexity while maintaining the architecture of 2D CNN-based methods. TSM shifts intermediate features between neighbor frames, allowing 2D CNNs of each frame to interact with other frames in the temporal direction. Many CNN-based variants have followed, such as Gated-Shift Network [20] that learns temporal features using only 2D CNN with the gate shift module, and RubiksNet [22] which uses three learnable shifts (spatial, vertical, and temporal) for end-to-end learning.

2.2 ViT-based video models

ViT [1] performs very well in image recognition tasks, and its application to action recognition tasks has been actively studied [4–8, 28, 29]. It is known that the computational cost of self-attention in Transformer is $O(N^2)$ where N is the number of tokens, or the number of patches N for ViT. For videos, the number of frames T also contributes to the computational cost of spatio-temporal self-attention. A simple extension of ViT with full spatio-temporal attention gives a computational cost of $O(N^2T^2)$, since the number of tokens increases in proportion to the temporal extent. To alleviate this computational issue, TimeSformer [7] reduces the computational complexity to $O(T^2N+TN^2)$ by applying temporal and spatial attention separately, TokenLeaner [28] to $O(S^2T^2)$ by selecting $S(< N)$ patches that are important for the representation, and Space-time Mixing Attention [29] to $O(TN^2)$ by attending tokens in neighbor frames only.

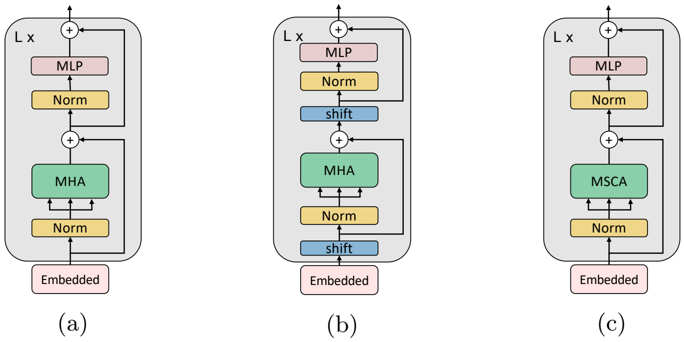
TokenShift [23] is a TSM-like model that shifts a fraction of ViT features in each Transformer block. It computes the attention only in the current frame, hence the complexity is $O(TN^2)$ with fewer FLOPs than [29]. TokenShift shifts the class token only based on experimental results, however it doesn't fully exploit the attention mechanism. In contrast, the proposed method makes use of the attention mechanism for feature shift; we replace some of the self-attention modules with the proposed cross-attention modules that shift key, query, and value to neighbor frames, instead of naively shifting features.

3 Method

Figure 1(a) shows the original ViT encoder block that has the Multi-head Self-Attention (MSA) module. Fig.1(b) shows the encoder block of TokenShift. Two modules for shifting intermediate features are added to the original ViT block. These modules shift a portion of the class tokens in the temporal direction by one time step forward and backward, similar to TSM. Fig.1(c) shows the encoder block of the proposed MSCA. The difference is that MSA is replaced with the Multi-head Self/Cross-Attention (MSCA) module, and no additional modules exist.

3.1 TokenShift and ViT

In this section, we briefly review TokenShift [23] which is based on ViT [1].



(a) (b) (c)

Fig. 1: Encoder blocks of (a) ViT, (b) TokenShift with two shift modules (in blue), and (c) the proposed method with MSCA (in green).

Input Patch Embedding Let an input video be $x \in \mathbb{R}^{T \times 3 \times H \times W}$, where T is the number of frames in the video clip, and H, W are the height and width of the frame. Each frame is divided into patches of size $P \times P$ pixels and transformed into a tensor $\hat{x} = [x_0^1, \dots, x_0^N] \in \mathbb{R}^{T \times N \times d}$, where $x_0^i \in \mathbb{R}^{T \times d}$ denotes the i-th patch, $N = \frac{HW}{P^2}$ is the number of patches of dimension $d = 3P^2$.

The input patch x_0^i is then transformed by the embedding matrix $E \in \mathbb{R}^{d \times D}$ and the positional encoding E_{pos} as follows:

$$z_0 = [c_0, x_0^1 E, x_0^2 E, \dots, x_0^N E] + E_{\text{pos}}, \tag{1}$$

where $c_0 \in \mathbb{R}^{T \times D}$ is the class token. This patch embedding $z_0 \in \mathbb{R}^{T \times (N+1) \times D}$ is the input to the first encoder block.

Encoder Block Let z_ℓ be the input to the ℓ-th encoder block. The output z_ℓ of the block can be expressed as follows:

$$z'_\ell = \mathrm{MSA}(\mathrm{LN}(z_{\ell-1})) + z_{\ell-1} \tag{2}$$

$$z_\ell = \mathrm{MLP}(\mathrm{LN}(z'_\ell)) + z'_\ell, \tag{3}$$

where LN is the layer normalization, MSA is the multi-head self-attention, and MLP is the multi-layer perceptron. In the following, $z_{\ell,t,n,d}$ denotes the element at (t, n, d) in z_ℓ.

Shift Modules TokenShift inserts two shift modules in the block as follows;

$$z'_{\ell-1} = \mathrm{Shift}(z_{\ell-1}) \tag{4}$$

$$z''_\ell = \mathrm{MSA}(\mathrm{LN}(z'_{\ell-1})) + z'_{\ell-1} \tag{5}$$

$$z'''_\ell = \mathrm{Shift}(z''_\ell) \tag{6}$$

$$z_\ell = \mathrm{MLP}(\mathrm{LN}(z'''_\ell)) + z'''_\ell. \tag{7}$$

The shift modules take the input $z_{\text{in}} \in R^{T\times(N+1)\times D}$ and compute the output z_{out} of the same size by shifting the part of z_{in} corresponding to the class tokens ($z_{\text{in},t,0,d}$, the first elements of the second dimension of z_{in}) while leaving the other parts untouched. This is implemented by the following assignments;

$$z_{\text{out},t,0,d} = \begin{cases} z_{\text{in},t-1,0,d}, & 1 < t \leq T, 1 \leq d < D_b \\ z_{\text{in},t+1,0,d}, & 1 \leq t < T, D_b \leq d < D_b + D_f \\ z_{\text{in},t,0,d}, & \forall t, D_b + D_f \leq d \leq D \end{cases} \tag{8}$$

$$z_{\text{out},t,n,d} = z_{\text{in},t,n,d}, \qquad \forall t, 1 \leq n < N, \forall d \tag{9}$$

The first equation shifts the class tokens; along the channel dimension D, the backward shift to $t-1$ is done for the first D_{b} channels, the forward shift to $t+1$ is done for the next D_{f}, and no shift for the rest channels. The second equation passes through features other than the class tokens.

3.2 MSCA

The proposed method replaces MSA with MSCA in the original block;

$$z'_\ell = \text{MSCA}(\text{LN}(z_{\ell-1})) + z_{\ell-1} \tag{10}$$

$$z_\ell = \text{MLP}(\text{LN}(z'_\ell)) + z'_\ell. \tag{11}$$

In the following, we first describe MSA, and then define MSCA.

MSA The original MSA computes the key $K^{(t)}$, query $Q^{(t)}$, and value $V^{(t)}$ for the portion $z^{(t)} \in \mathbb{R}^{(N+1)\times D}$ of the input feature $z \in \mathbb{R}^{T\times(N+1)\times D}$ at time t, as follows;

$$K^{(t)}, Q^{(t)}, V^{(t)} = z^{(t)}[W_k, W_q, W_v], \tag{12}$$

where $W_k, W_q, W_v \in \mathbb{R}^{D\times D}$ are embedding matrices, and $z = [z^{(1)}, \ldots, z^{(T)}]$. These are used to compute the i-th attention head;

$$\text{head}_i^{(t)} = a(Q_i^{(t)}, K_i^{(t)})V_i^{(t)} \in \mathbb{R}^{(N+1)\times D/h}, \tag{13}$$

at time t for $i = 1, \ldots, h$, where the attention a is

$$a(Q, K) = \text{softmax}(QK^T/\sqrt{D}), \tag{14}$$

and $Q_i^{(t)} \in \mathbb{R}^{(N+1)\times D/h}$ is the part of $Q^{(t)}$ corresponding to i-th head

$$Q^{(t)} = [Q_1^{(t)}, \ldots, Q_i^{(t)}, \ldots, Q_h^{(t)}], \tag{15}$$

and $K_i^{(t)}, V_i^{(t)}$ are the same. These heads are finally stacked to form

$$\text{MSA}(z^{(t)}) = [\text{head}_1^{(t)}, \ldots, \text{head}_h^{(t)}]. \tag{16}$$

In MSA, patches in t-th frame are attended from other patches of the same frame at time t, which means that there are no temporal interactions between frames.

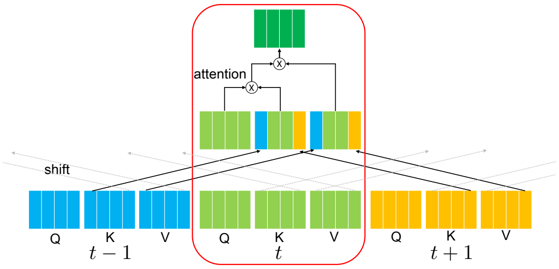
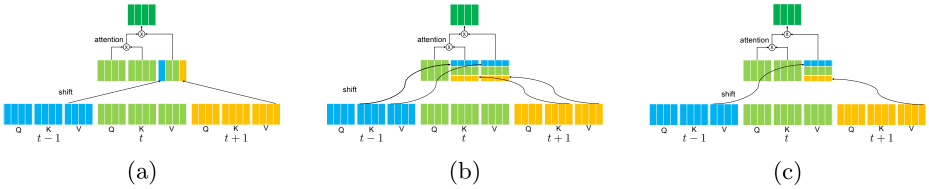


Fig. 2: Shit operation at $T = t$ in the MSCA-KV model



(a) (b) (c)

Fig. 3: Shit operations of (a) MSCA-V, (b) MSCA-pKV, and (c) MSCA-pV.

MSCA-KV The proposed MSCA computes the attention across frames, that is, patches in t-th frame are also attended from patches in frames at time $t+1$ and $t-1$. This can be done with shift operations; after generating Q, K, V at each frame, these are exchanged with the neighbor frames.

There are choices of which of Q, K, or V to be shifted. A possible choice is to shift K and V, and the query in the current frame is attended by key-value pairs in other frames. This is expressed as follows;

$$\mathrm{head}_i^{(t)} = \begin{cases} a(Q_i^{(t)}, K_i^{(t-1)})V_i^{(t-1)} & 1 \leq i < h_b \\ a(Q_i^{(t)}, K_i^{(t+1)})V_i^{(t+1)} & h_b \leq i < h_b + h_f \\ a(Q_i^{(t)}, K_i^{(t)})V_i^{(t)} & h_b + h_f \leq i \leq h. \end{cases} \tag{17}$$

Here, we have the first h_b heads with the backward shift, the next h_f heads with the forward shift, and the rest heads with no shift. We call this MSCA-KV, and Figure 2 shows the diagram of the shift operation. First, queries, keys and values are computed at each frame, and then some of them are shifted before computing the attention. The solid arrows indicate the key-value shift from time $t-1$ and $t+1$. As in the same way, the key-value shift from the current frame t to $t-1$ and $t+1$ are shown in dotted arrows. There are shifts in all other frames at the same time in the same manner.

MSCA-V Another choice of shift is shown in Fig. 3(a). Here, Q and K are not shifted, but only V is shifted as follows;

$$\mathrm{head}_i^{(t)} = \begin{cases} a(Q_i^{(t)}, K_i^{(t)})V_i^{(t-1)} & 1 \leq i < h_b \\ a(Q_i^{(t)}, K_i^{(t)})V_i^{(t+1)} & h_b \leq i < h_b + h_f \\ a(Q_i^{(t)}, K_i^{(t)})V_i^{(t)} & h_b + h_f \leq i \leq h. \end{cases} \tag{18}$$

This might not be common because the key and value are now separated and taken from different frames. However, this makes sense for modeling temporal interactions because the values (which are mixed by the attention weights) come from different frames while the attention is computed in the current frame. We call this version MSCA-V. Therefore, in addition to shifting V in this way, there are seven possible combinations; Q, K, V, QK, KV, QV, and QKV. We compared these variants in the experiments. Note that MSCA-QKV is equivalent to a simple feature shifting because attended features are computed in each frame and then shifted.

MSCA-pKV All of the above seven variants perform shift operations along the head (or channel) dimension D, but similar variants of shift are also possible for the patch dimension $N + 1$.

The shapes of $K^{(t)}, Q^{(t)}, V^{(t)}$ are $\mathbb{R}^{(N+1)\times D}$, and the first dimension is for patches while the second is for heads. As in the same way the the shift along the head dimension, we have a variation of shift operations along the patch dimension as shown in Fig. 3(b).

First, K and V are expressed as stacks of keys and values of patches at different frames as follows;

$$K^{(t)} = [K_0^{(t)}, K_1^{(t)}, \ldots, K_N^{(t)}] \tag{19}$$

$$V^{(t)} = [V_0^{(t)}, V_1^{(t)}, \ldots, V_N^{(t)}], \tag{20}$$

where $K_n^{(t)}, V_n^{(t)} \in \mathbb{R}^D$ are the key and value of patch n at time t.

Then, keys of some patches in the current frame are shifted to form K';

$$K_n'^{(t)} = \begin{cases} K_n^{(t-1)} & 0 \leq n < N_b \\ K_n^{(t+1)} & N_b \leq n < N_b + N_f \\ K_n^{(t)} & N_b + N_f \leq n \leq N, \end{cases} \tag{21}$$

and also V' in the same way. Finally, the i-th head is computed as follows

$$\mathrm{head}_i^{(t)} = a(Q_i^{(t)}, K_i'^{(t)})V_i'^{(t)}. \tag{22}$$

We refer to this version as MSCA-pKV.

MSCA-pV Like as MSCA-V, a variant of the shift in the patch direction with V only can be also considered as shown in Fig. 3(c), by the following shift;

$$\mathrm{head}_i^{(t)} = a(Q_i^{(t)}, K_i^{(t)})V_i'^{(t)}. \tag{23}$$

As before, there are seven variants, and we call these MSCA-pV, and so on.

4 Experimental results

4.1 Setup

Kinetics400 [14] was used to train and evaluate the proposed method. This dataset consists of a training set of 22k videos, a validation set of 18k videos, with 400 categories of human actions. Each video was collected from Youtube, and the portion corresponding to each category was cropped to a length of 10 seconds.

We used a 2D ViT pre-trained on ImageNet21k [30] with $h = 12$ heads, 12 encoder blocks, and patches of size $P = 16$ and $D = 768 = 3 \times 16 \times 16$ (these parameters are the same for TokenShift and MSCA models). For action recognition, ViT was applied to each frame and resulting frame-wise features were aggregated by temporal averaging (this is referred to as ViT in the experiment and in [23]). We compared this ViT, TokenShift and the proposed method. Note that we report the performance of TokenShift based on our reproduction using the author's code.[1]

For training, we used the same settings as in [23]. Input clips were of 8 frames with stride of 32 frames (starting frames were randomly chosen). Frames were flipped horizontally at a probability of 50%, and the short side was randomly resized in the range of [244, 330] pixels while maintaining the aspect ratio, then a random 224×224 pixel rectangle was cropped (therefore the number of patches is $N = 196 = 14 \times 14$). In addition, brightness change, saturation change, gamma correction, and hue correction were applied to frames, each at the probability of 10%. The number of epochs was set to 12, the optimizer to SDG with momentum of 0.9 and no weight decay. The initial learning rate was set to 0.1, and decayed by a factor of 10 at 10th epoch. The batch size was set to 42, and 21 batches were trained on each of two GPUs. Gradient updates were performed once every 10 iterations, so the effective batch size was 420.

We used the multi-view test [31]. From a validation video, one clip was sampled as in training, and this was repeated 10 times to sample 10 clips. Each clip was resized to 224 pixels on its short side while maintaining the aspect ratio, and cropped to 224×224 at the right, center, and left. The results of these 30 clips (views) were averaged to compute a single prediction score.

4.2 The amount of shift of MSCA-KV

We first investigate the effect of the number of heads to be shifted. Table 1 shows the performance of MSCA-KV. The best performance was obtained when only two heads (each for forward and backward) which corresponds to shifting $2/12 = 16.7\%$ of the channels. As the number of shifted heads increased, the performance decreased, suggesting that shifting a few heads is sufficient while most heads need not to be shifted. This observation coincides with the conclusions of TokenShift [23], which shows that the shift of the class token only is enough, and also TSM

[1] `https://github.com/VideoNetworks/TokShift-Transformer`

Table 1: The performance of MSCA-KV for the validation sets of Kinetics400. Note that the zero shift means a naive frame-wise ViT. The column "shift" means the percentage of the shifted dimensions to the total dimensions D.

model	heads	h_b, h_f	shift %	top-1	top-5
ViT	0	0	0 (ViT)	75.65	92.19
MSCA-KV	2	1	16.7	**76.47**	**92.88**
	4	2	33.3	76.07	92.61
	6	3	50.0	75.66	92.30
	8	4	66.7	74.72	91.91

Table 2: The performance of MSCA variants shifting in the head direction.

model	top-1	top-5
TokenShift	76.37	92.82
ViT	75.65	92.19
MSCA-Q	75.84	92.57
MSCA-K	75.76	92.13
MSCA-V	75.58	92.39
MSCA-QK	75.47	92.32
MSCA-KV	**76.47**	**92.88**
MSCA-QV	75.57	92.43
MSCA-QKV	75.78	92.37

Table 3: The performance with different numbers of blocks with MSCA modules of KV shift.

model	# MSCA	# MSA	top-1	top-5
ViT	0	12	75.65	92.19
	4	8	75.67	92.19
	8	4	76.40	92.77
MSCA-KV	12	0	**76.47**	**92.88**

[16], which used the shift of 1/4 = 25% of the channels (each 1/8 for forward and backward).

In the following experiments, we used shifting two heads for all MSCA variations.

4.3 Comparison of MSCA variations shifting in the head direction

Table 2 shows the performance of the MSCA model variants with shift operations in the head direction. Among them, MSCA-KV performed the best, outperforming others by at least 0.5%. Other variants performed as same as the baseline ViT, indicating that the shift operation is not working effectively in such variations.

4.4 The number of encoder blocks with MSCA

The proposed method replaces MSA modules in 12 encoder blocks in ViT with MSCA modules. However, it is not obvious that modules of all blocks need to be replaced. Table 3 shows the results when we replaced MSA modules with MSCA in some blocks near the end (or top) of the network. All 12 replacements

Table 4: The performance of MSCA-pKV. The column "shift" means the percentage of the shifted patches to the total patches $N + 1$.

patches	N_b, N_b	shift %	top-1	top-5
0	0	0 (ViT)	75.65	92.19
8	4	4.1	76.28	92.49
16	8	8.1	**76.35**	**92.91**
32	16	16.2	76.07	92.77
48	24	24.4	75.84	92.49
64	32	32.5	75.29	92.04

Table 5: The performance of MSCA models shifting in the patch direction.

model	top-1	top-5
TokenShift	**76.37**	92.82
ViT	75.65	92.19
MSCA-pQ	75.75	92.10
MSCA-pK	75.69	92.24
MSCA-pV	75.84	92.43
MSCA-pQK	75.50	91.99
MSCA-pKV	76.35	**92.91**
MSCA-pQV	75.83	92.35
MSCA-pQKV	75.73	92.21

correspond to MSCA-KV, and 0 (no MCSA) is ViT. Using four MSCA modules showed no improvement, while using 8 MSCA modules performed almost the same as MSCA-KV with all 12 MSCA modules. This is because the input video clip consists of 8 frames, and shifting more than 8 times with MSCA modules ensures that the temporal information from all frames is available for the entire network. Therefore, it would be necessary to use at least as many MSCA modules as the number of frames in the input clip.

4.5 Comparison of MSCA variations shifting in the patch direction

Table 4 shows the performance of MSCA-pKV with different amount of shift. The results indicate that, again, a small amount of shift is enough for a better performance and the best performance was obtained when 16 patches were shifted. The performance decreases for a smaller amount of shift, approaching the performance of ViT with no shift. In the experiments below, we used the shift of 16 patches.

Table 5 shows the performance of MSCA variants with shifting in the patch direction. Just like as MSCA-KV was the best for shifting in the head direction, MSCA-pKV has the best performance here, while it is just comparable to TokenShift. An obvious drawback of this approach is the first layer; the patches to be shifted were fixed to the first N_b and N_f patches in the order of the patch index, which is irrelevant to the content of the frame. This might be mitigated by not using MSCA modules in the first several layers.

5 Conclusions

In this paper, we proposed MSCA, a ViT-based action recognition model that replaces MSA modules in the encoder blocks. The MSCA modules compute the attention by shifting of the key, query, and value for temporal interaction between frames. Experimental results using Kinetics400 showed that the proposed method is effective for modeling spatio-temporal features and performs better than a naive ViT and TokenShift. Future work includes evaluations on other datasets and comparisons with similar methods such as Space-time Mixing Attention [29].

Acknowledgements. This work was supported in part by JSPS KAKENHI Grant Number JP22K12090.

References

1. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations. (2021)
2. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. CoRR **abs/2103.00020** (2021)
3. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. In Meila, M., Zhang, T., eds.: Proceedings of the 38th International Conference on Machine Learning. Volume 139 of Proceedings of Machine Learning Research., PMLR (2021) 8821–8831
4. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2021) 6836–6846
5. Li, X., Zhang, Y., Liu, C., Shuai, B., Zhu, Y., Brattoli, B., Chen, H., Marsic, I., Tighe, J.: Vidtr: Video transformer without convolutions. CoRR **abs/2104.11746** (2021)
6. Girdhar, R., Carreira, J., Doersch, C., Zisserman, A.: Video action transformer network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
7. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: Proceedings of the International Conference on Machine Learning (ICML). (2021)
8. Sharir, G., Noy, A., Zelnik-Manor, L.: An image is worth 16x16 words, what is a video worth? CoRR **abs/2103.13915** (2021)
9. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2014)
10. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2015)
11. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (2015)
12. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)

13. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018) 6546–6555
14. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., Zisserman, A.: The kinetics human action video dataset. CoRR **abs/1705.06950** (2017)
15. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. CoRR **abs/1212.0402** (2012)
16. Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019)
17. Chen, W., Xie, D., Zhang, Y., Pu, S.: All you need is a few shifts: Designing efficient convolutional neural networks for image classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019)
18. Wu, B., Wan, A., Yue, X., Jin, P., Zhao, S., Golmant, N., Gholaminejad, A., Gonzalez, J., Keutzer, K.: Shift: A zero flop, zero parameter alternative to spatial convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016)
20. Sudhakaran, S., Escalera, S., Lanz, O.: Gate-Shift Networks for Video Action Recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2020)
21. Chang, Y.L., Liu, Z.Y., Lee, K.Y., Hsu, W.: Learnable gated temporal shift module for deep video inpainting". BMVC (2019)
22. Fan, L., Buch, S., Wang, G., Cao, R., Zhu, Y., Niebles, J.C., Fei-Fei, L.: Rubiksnet: Learnable 3d-shift for efficient video action recognition. In Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., eds.: Computer Vision – ECCV 2020, Cham, Springer International Publishing (2020)
23. Zhang, H., Hao, Y., Ngo, C.W. In: Token Shift Transformer for Video Classification. Association for Computing Machinery, New York, NY, USA (2021) 917–925
24. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K.Q., eds.: Advances in Neural Information Processing Systems. Volume 27., Curran Associates, Inc. (2014)
25. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (2017)
26. Zhang, D., Dai, X., Wang, X., Wang, Y.F.: S3d: Single shot multi-span detector via fully 3d convolutional network. In: Proceedings of the British Machine Vision Conference (BMVC). (2018)
27. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)
28. Ryoo, M.S., Piergiovanni, A., Arnab, A., Dehghani, M., Angelova, A.: Tokenlearner: Adaptive space-time tokenization for videos. In: Advances in Neural Information Processing Systems (NeurIPS). (2021)

29. Bulat, A., Perez-Rua, J.M., Sudhakaran, S., Martinez, B., Tzimiropoulos, G.: Space-time mixing attention for video transformer. In Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W., eds.: Advances in Neural Information Processing Systems. (2021)
30. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. (2009) 248–255
31. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)