VideoMamba: Spatio-Temporal Selective State Space Model

ECCV 2024 submission

Paper ID 3565

Table of Contents

S1 Comparison of Computational Complexity

In this section, we compare the computational complexity of transformer-based model and our VideoMamba. The multi-head self-attention, which is the basic building block of transformer [9], includes computation of the following scaled dot-product attention:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V. \tag{S1}$$

Since the dot-product attention requires calculating $n \times n$ attention matrix, the complexity of self-attention is *quadratic* in input token length n. Therefore, for the input video token with the size $(n_t \cdot n_h \cdot n_w, d)$, the computation complexity of ViViT [1], which uses factorized spatio-temporal encoder, would be $\mathcal{O}((n_h \cdot n_w)^2 + n_t^2)$. On the other hand, since our VideoMamba utilizes selective SSM [5], our model achieves *linear* computational complexity of $\mathcal{O}(n_h \cdot n_w \cdot n_t)$.

S2 Additional Delta Visualizations

To better understand VideoMamba's ability to dynamically select relevant spatio-temporal contexts, we provide additional examples from HMDB51 validation set in Fig. S1, Fig. S2 and Fig. S3. These figures depict the original video sequences alongside their corresponding deltas across multiple layers.

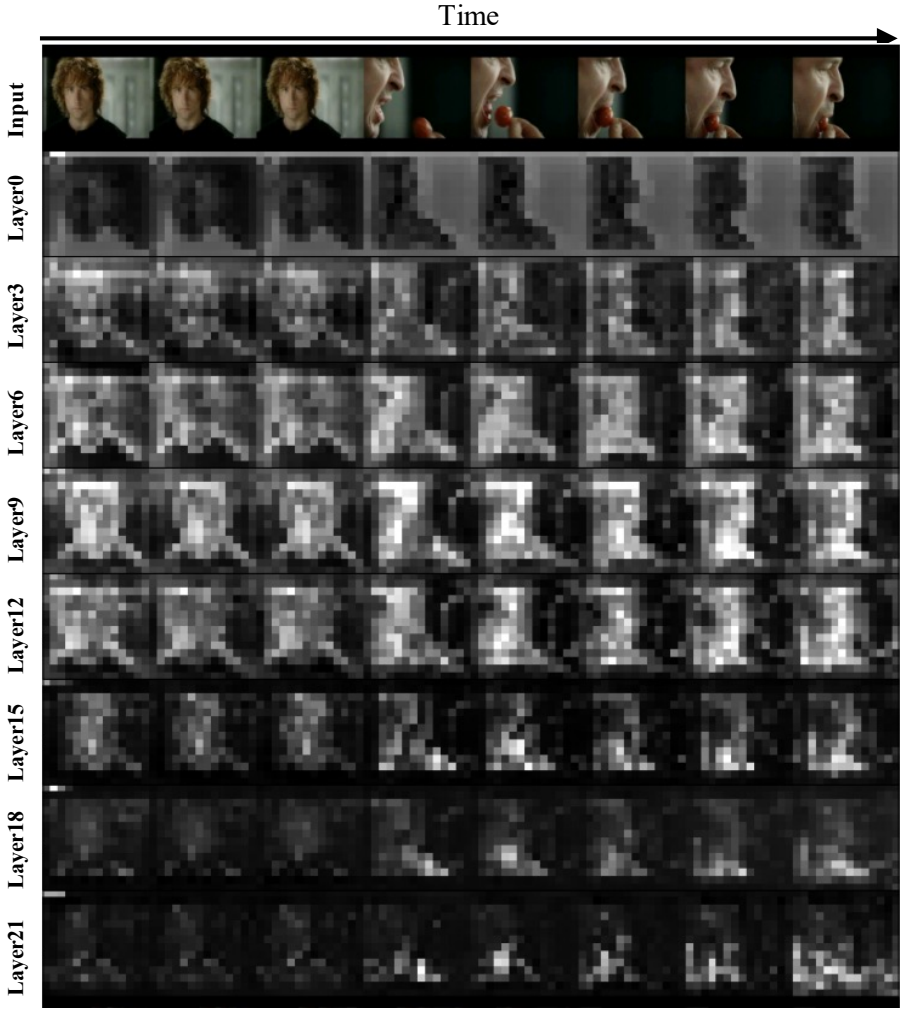

Fig. S1: **Delta visualizations** on HMDB51 validation set. Darker regions correspond to smaller delta values, indicating the model prioritizing information from previous time steps, and brighter areas represent larger delta values, representing the model placing greater emphasis on the current input. The GT label is "*Eat*". As the network goes deeper into the layers, the delta values decrease, which allows the model to effectively filter out not directly related areas(e.g., upper padded areas) or frames (e.g., initial three frames of layers 18 and 21) and focus on the elements necessary for key parts(e.g., hand and tomato). The high delta values in the initial layers demonstrate the model's process of first understanding the overall scene and then selectively focusing on important details later. Through delta value analysis, we can glean the VideoMamba's capability in performing efficient spatiotemporal reasoning.

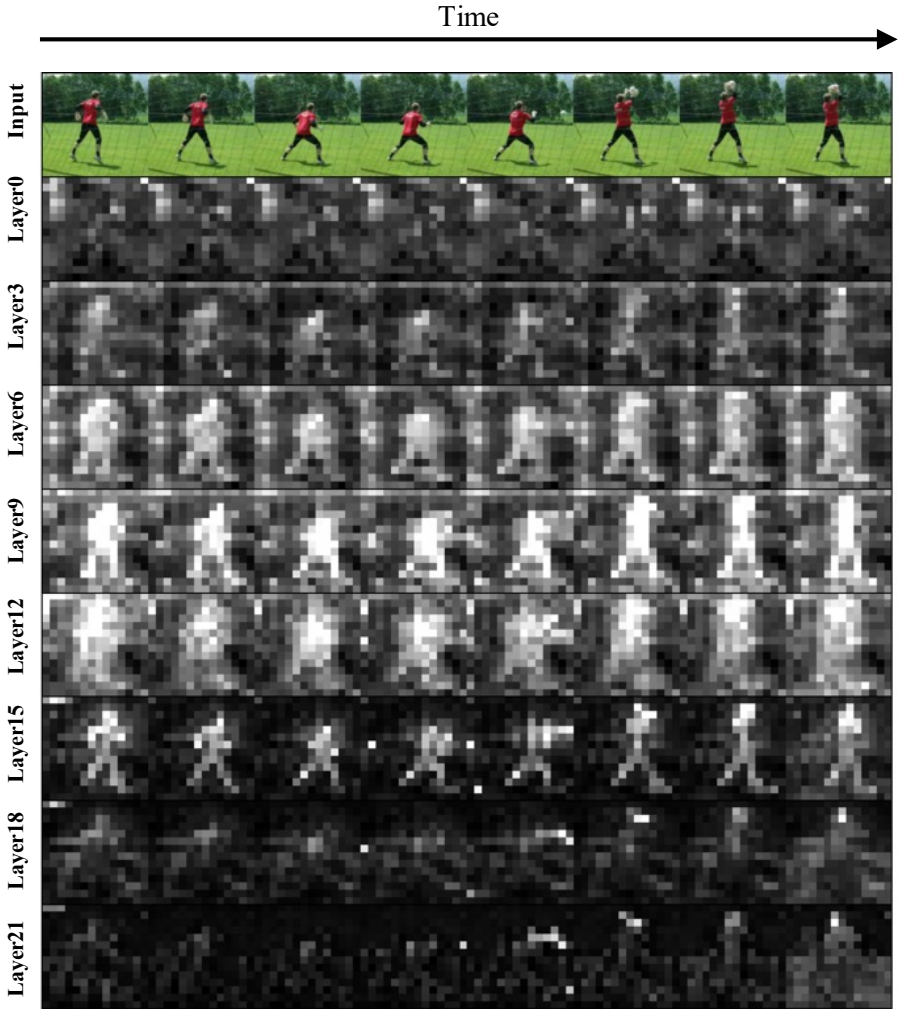

Fig. S2: **Delta visualizations** on HMDB51 validation set. The GT label is "*Catch*". Within deeper layers, the VideoMamba show a growing emphasis on extracting features relevant to the class of interest (e.g., hand and ball).

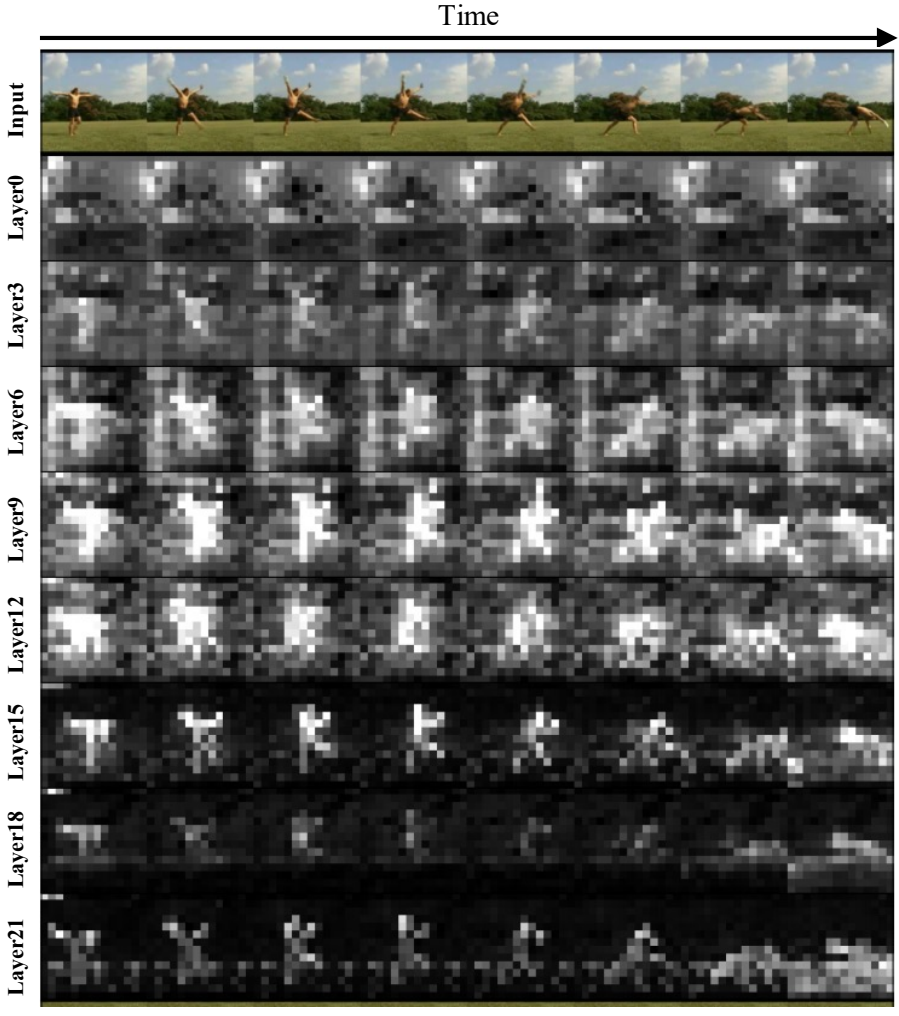

Fig. S3: **Delta visualizations** on HMDB51 validation set. The GT label is "*Cartwheel*".

S3 Importance of Pretraining

Table S1: Comparing the effectiveness of ImageNet-1K and Kinetics-400 (K400) pretraining on Something-Something V2 (SSV2) and HMDB51 (HMDB).

Pretrain	SSV2	HMDB
	44.0	18.1
ImageNet-1K	63.7	59.3
K400	63.9	68.6

As reported in previous work [2], the performance of video recognition model depends considerably on pretraining. In our main experiments, we initialized our models with ImageNet pretrained weights. In this section, using our base model, we compare the effect of different pretraining datasets on performance, including the experimental results trained from scratch.

Table S1 reports the Top-1 accuracy of differently pretrained models, on Something-Something V2 [4] and HMDB51 [6] datasets. When training from scratch, we trained the model for 100 epochs in Something-Something V2, and 200 epochs in HMDB51. We observe that training VideoMamba from scratch results in much lower accuracy, especially in small dataset such as HMDB. We also observe that pretraining on K400 leads to superior performance in HMDB, and slight improvement on SSV2.

S4 Architecture

S4.1 Architectural Details

Table S2: Architectural Details of VideoMamba.

stage	VideoMamba		Output Sizes
data	stride 2 1 1 on K400		3 32 224 224
tubelet	Conv3d 2 16 16, 384, stride 2 16 16		3136 384
encoder	[linear 384 ! 384 2 conv1d 384 2 ! 384 2 ST-SSM 384 2 ! 384 2 linear 384 2 ! 384]	24	3136 384
projector	linear 384 ! 400 (# labels)		400

Table S2 outlines the structure of the VideoMamba model, highlighting its various stages from data input to the final projection. The dimensions are emphasized in violet.

Data Stage: The initial stage involves processing video data with a stride of 2×1×1 on the K400 dataset. The output size is 3×32×224×224, indicating the transformation of video frames into a tensor with 3 channels (color depth), 32 frames per sequence, and a spatial dimension of 224×224 pixels per frame.

Tubelet Stage: At this stage, a tubelet operation is applied 3d conv with a kernel and stride of 2×16×16, producing an output with a dimension of 3136×384. This illustrates the extracted spatial-temporal features from the input video frames, where 384 represents the feature vector length for each of the 3136 tubelets.

Encoder Stage: This stage, which is repeated 24 times as indicated, involves a sequence of operations starting with a linear transformation from 384 to 384×2 (doubling the feature dimension), followed by a 1D convolution that maintains the feature dimension at 384×2. Spatio-Temporal SSM (ST-SSM) processes these features without altering the dimension, leading to a final linear transformation that maps the features back to a dimension of 384. The output retains the format of 3136×384, emphasizing the consistency in the model's internal representation of features.

Projector Stage: The final stage involves a linear transformation from a 384-dimensional feature vector to the number of labels required for classification. This stage is for adapting the model's learned representations to specific tasks, such as video classification or action recognition. The number of labels is dependent on the application and, thus 400 for K400 dataset in the table.

S4.2 Positional Embedding

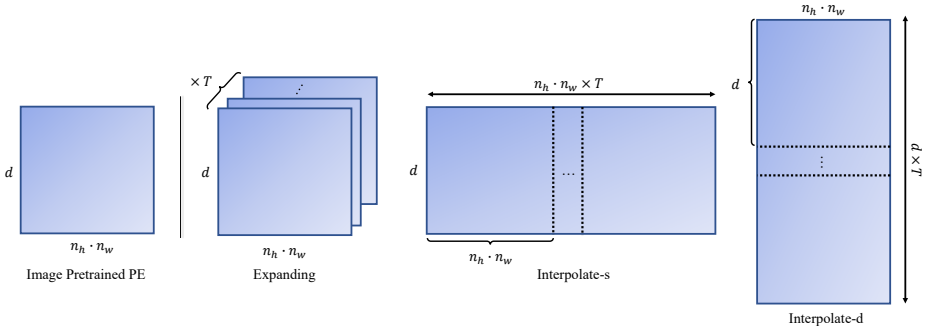

Fig. S4: Initialization strategies for learnable positional embeddings in VideoMamba. The figure illustrates three different approaches to modify pretrained image positional embeddings (P_{image}) for use in video data, which introduces an additional temporal dimension (T).

When employing learnable positional embeddings for video data, initialization plays an important role in leveraging pre-trained image model knowledge. Unlike expanding, which technically generates replicated and discontinuous PE for each frame, spatial interpolation (similar to image interpolation) can generate continuous PE across frames. We assumed this continuously initialized PE might provide additional temporal information for the model. Figure S4 illustrates the methods we propose **for adapting P_{image} to video data.** We propose several initialization techniques for the learnable positional embedding P, starting from the pretrained image positional embedding $P_{image} \in \mathbb{R}^{n_h \cdot n_w \times d}$, which represents the special case of $T = 1$. Our proposed methods include:

Temporal Expansion: We replicate P_{image} along the temporal dimension n_t times, effectively copying the spatial embeddings across the additional time frames.

Spatial Interpolation: We interpolate P_{image} in the spatial dimensions to obtain embeddings in $\mathbb{R}^{(n_h \cdot n_w \times n_t) \times d}$, matching the spatial-temporal structure of video data.

Embedding Dimension Interpolation: We interpolate P_{image} in the embedding dimension, expanding it to $\mathbb{R}^{n_h \cdot n_w \times (d \times n_t)}$ before reshaping, to integrate temporal information.

Each method is designed to adapt the effective spatial embeddings from image models to the spatio-temporal domain of video data.

S5 Implementation Details

Table S3: Training setting for VideoMamba

config	K400	SSV2	HMDB
optimizer	*AdamW*		
optimizer momentum	$_1$; $_2$ = 0:9; 0:999		
weight decay	0.05		
learning rate schedule	*CosineAnealing*		
learning rate	3e-4		
batch size	64		
warmup epochs	1	1	5
total epochs	30	35	50
drop path	0.1		
repeated augmentation	*no*		
RandAug [3]	(9,0.5)		
label smoothing [8]	0.1		
flip augmentation	*yes*	*no*	*yes*

In this section, we provide additional experimental details. Table S3 summarizes the hyperparameters employed for all experiments. We opted for the AdamW optimizer with cosine learning rate schedule. A consistent batch size of 64 was maintained across all experiments. For the large-scale Kinetics-400 dataset, we leveraged two NVIDIA A100 GPUs. Conversely, for the smaller Something-Something V2 (SSV2) and HMDB51 datasets, we employed eight NVIDIA 3090 GPUs for training. We initialized all models with pre-trained weights [10] obtained from the ImageNet dataset. We carefully re-implement VideoSwin-T with the VideoSwin-B training strategy [7] for the SSV2 dataset and the HMDB51 dataset, with the same augmentation strategy as ours.

S6 Inference Speed

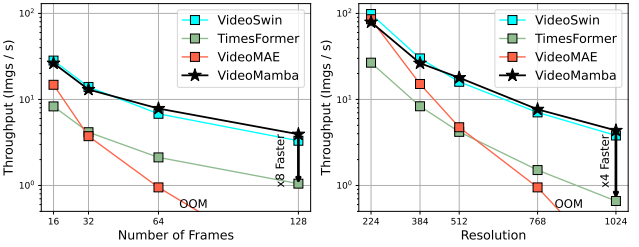

Fig. S5: Comparison of Inference Speed (Throughput).

In Fig. S5, our model (black curves) demonstrates comparable or even faster (×8) inference speed compared to transformer-based models, especially with longer and higher-resolution videos.

S7 Long-Term Video Modeling

To further validate VideoMamba's long-term modeling capabilities, we conducted additional experiments on the Breakfast dataset, which contains longer untrimmed videos. Our VideoMamba$_{f64}$ model, using 64 input frames, achieves state-of-the-art performance with 91.5% accuracy on Breakfast, surpassing all previous methods.

Table S4: Long-Term Video Modeling Results on Breakfast.

Method	Backbone	Pretrain	TOP-1
Timeception	3D-ResNet	IN-1K+K400	71.3
GHRM	I3D	IN-1K+K400	75.5
Distant S.	TimeSformer	IN-21K+HTM	89.9
Turbo$_{f32}$	VideoMAE-B	K400	86.8
ViS4mer$_{f32}$	Swin-B+SSM	IN-21K+K600	88.2
LSMCL$_{f64}$	Swin-B+SSM	K600	90.1
Ours$_{f32}$	Mamba	IN-1K+K400	90.4
Ours$_{f64}$	Mamba	IN-1K+K400	**91.5**

S8 Applicability to Other Video Tasks

In addition to action recognition, our additional experiments present strong performance in action detection and temporal segmentation. This demonstrates VideoMamba's potential as a versatile and efficient backbone for various video understanding tasks.

Action Detection. Compared to models with similar size, Tab. S5 shows VideoMamba outperforms the transformer-based VideoSwin-T backbone on the AVA action detection dataset, while requiring less computation (34 vs 44 GFLOPs).

Temporal Action Segmentation. Table S6 shows that integrating VideoMamba into the ASFormer model leads to improved performance on the GTEA dataset, especially in terms of the F1 score and edit score.

Table S5: Action Detection on Results AVA 2.2.

Method	Backbone	GFLOPs (#)	mAP
CVRL$_{f32}$	SlowOnly-R50	42	16.3
VideoMAE$_{f16}$	ViT-S	57	22.5
VideoSwin$_{f16}$	Swin-T	44	18.0
Ours$_{f16}$	Mamba	**34**	**22.1**

Table S6: Action Segmentation Results on GTEA.

Method	F1@f10,25,50g			MoF	Edit
BCN	88.5	87.1	77.3	79.8	84.4
MS-TCN++	88.8	85.7	76.0	**80.1**	83.5
ASFormer	90.1	88.8	79.2	79.7	84.6
ASFormer w/ Ours	**90.6**	**89.7**	**79.9**	79.6	**86.6**

References

1. 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. pp. 6836–6846 (2021) 1
2. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: ICML. vol. 2, p. 4 (2021) 5
3. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 702–703 (2020) 8
4. Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The" something something" video database for learning and evaluating visual common sense. In: Proceedings of the IEEE international conference on computer vision. pp. 5842–5850 (2017) 5
5. Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752 (2023) 1
6. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: Hmdb: a large video database for human motion recognition. In: 2011 International conference on computer vision. pp. 2556–2563. IEEE (2011) 5
7. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3202–3211 (2022) 8
8. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016) 8
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 1
10. Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., Wang, X.: Vision mamba: Efficient visual representation learning with bidirectional state space model. arXiv preprint arXiv:2401.09417 (2024) 8