Supplementary Material for Data Collection-free Masked Video Modeling

Overview of Supplementary Material

In this supplementary material, we provide more details on our framework and analyses of our experiments with respect to the following points:

- Details on video datasets (Sec. A)
- Implementation details (Sec. B)
- Pseudo-code of Pseudo Motion Generator (PMG) (Sec. C)
- Parameters of image augmentations in PMG (Sec. D)
- Examples of pseudo-motion videos generated by PMG (Sec. E)
- Quantitative results of our framework (Sec. F)
- Failure cases (Sec. G)
- Linear probing (Sec. H)

A Details on Video Datasets

In our experiments, we use seven datasets to evaluate the effectiveness of our framework; UCF101 [17], HMDB51 [12], MiniSSV2 [3], Diving48 [14], IkeaFA [21], UAV-Human (UAV-H) [13], and Kinetics400 (K400) [10]. The first six datasets are included in the SynAPT benchmark [11] We conducted our experiments following its setup. Herein, we provide an overview of the datasets used in this study.

UCF101 [17]: This dataset features approximately 13,000 videos classified into 101 categories of actions. These categories are segmented into five groups: (i) Human-object Interaction (e.g., Juggling Balls), (ii) Body-Motion Only (e.g., Push Ups), (iii) Human-Human Interaction (e.g., Head Massage), (iv) Playing Musical Instruments (e.g., Drumming), and (v) Sports (e.g., Archery).

HMDB51 [12]: Comprising roughly 6,000 video clips sourced from both movies and YouTube, this dataset is annotated across 51 action categories. These categories encompass five action types: (i) general facial actions (e.g. smile), (ii) facial actions with object manipulation (e.g. eat), (iii) general body movements (e.g. jump), (iv) body movements with object interaction (e.g. kick ball), (v) body movements for human interaction (e.g. punch).

MiniSSV2 [3]: MiniSSV2 [11] is a subset of Something-Something V2 (SSV2) [6], which encompasses over 220,000 video clips with 174 action classes. MiniSSV2 contains just half of the original action categories, with 87 randomly selected labels. The total number of videos is approximately 93,000 videos. Actions in this dataset are basic interactions with everyday objects, defined via caption templates like "Moving something up" or "Covering something with something".

Diving48 [14]: Dedicated to competitive diving, this dataset consists of about 18,000 videos categorized into 48 distinct types of diving actions. All videos

Table 1: Pre-training setting for each dataset.

configuration	Kinetics400	MiniSSV2	Other Datasets
optimizer	AdamW [15]		
learning rate	1e-3		
weight decay	0.05		
optimizer momentum	$\beta_1 = 0.9, \beta_2 = 0.95$		
mask ratio	0.75		
batch size	256		
batch size	256		
learning rate schedule	cosine decay		
warmup epochs	40		
epochs	800	2000	2000
flip augmentation	✓	-	✓

Table 2: Fine-tuning setting for each dataset.

configuration	Kinetics400	MiniSSV2	Other Datasets
optimizer	AdamW [15]		
learning rate	1e-3		
weight decay	0.05		
optimizer momentum	$\beta_1 = 0.9, \beta_2 = 0.999$		
batch size	128		
learning rate schedule	cosine decay		
warmup epochs	5		
epochs	50	100	100
repeated augmentation [7]	2		
flip augmentation	✓	-	✓
RandAug [4]	(9, 0.5)		
label smoothing [18]	0.1		
mixup [25]	0.8		
cutmix [24]	1.0		
drop path [8]	0.1	0.1	0.2
dropout	0.0	0.0	0.5
layer-wise lr decay [1]	0.75		
sampling	dense sampling [5, 23]	uniform sampling [22]	dense sampling

in Diving48 exhibit consistent background and object characteristics. Therefore, this dataset is often used to evaluate how the models capture motion information.

IkeaFA [21]: Ikea Furniture Assembly (IkeaFA) offers 111 video clips, each lasting between 2 to 4 minutes, accumulating roughly 480,000 frames. This dataset consists of videos captured by GoPro cameras showcasing furniture assembly tasks, all recorded against a uniform background by 14 individuals. IkeaFA categorizes these assembly actions into 12 classes.

UAV-Human (UAV-H) [13]: This dataset is gathered through the lens of an Unmanned Aerial Vehicle, offering a unique perspective through its collection of video footage. This dataset features a variety of recording types, including fisheye and night-vision videos. In our study, we use videos captured by standard RGB cameras. This subset includes 22,476 videos having 155 different action categories.

Kinetics400 (K400) [10]: This large-scale dataset includes around 300,000 video clips, each labeled with one of 400 actions. The Kinetics400 videos are all sourced from YouTube and last about 10 seconds each.

B Implementation Details

We conducted the experiments with 8 A100 GPUs for both pre-training and fine-tuning, mostly following the settings in VideoMAE [20]. The settings for pre-training are detailed in Tab. 1 and those for fine-tuning are described in Tab. 2. We used PyTorch [16] to implement our framework.

C Pseudo-code of Pseudo Motion Generator (PMG)

While the algorithm of our Pseudo Motion Generator (PMG) is detailed in the main paper, we offer Python pseudo-code for PMG in Fig. 1 for more clarity.

D Parameters of Image Augmentations in PMG

Since it is difficult to find the optimal parameters for each image augmentation in our framework, we implement each augmentation with a predefined range of parameters as follows:

- **Sliding Window:** Cut a 112 × 112 window from a 224 × 224 image and move it randomly.
- **Zoom-in/out:** For Zoom-out, randomly set a window from a 224 × 224 image within the size range of [0.2, 0.45], then gradually enlarge the window until it reaches a random size between [0.55, 0.95]. For Zoom-in, reverse the process for pseudo-motion videos generated by Zoom-out. We randomly choose between Zoom-in and Zoom-out with a 50% probability.
- **Fade-in/out:** For Fade-out, make an input image gradually become completely invisible. For Fade-in, reverse the process of pseudo-motion videos generated by Zoom-in. We randomly choose between Fade-in and Fade-out with a 50% probability.
- **Affine Transformation** We use the AffineTransformation class provided in PyTorch [16]. The rotation angle in degrees is randomly selected between -15 and 15. The translation is randomly selected between [-0.01, 0.01] for both horizontal and vertical directions. The scale value is randomly selected between [0.9999, 1.0001]. The shear angle value in degrees is randomly selected between -1 and 1.
- **Perspective Transformation** We use the PerspectiveTransformation class provided in PyTorch. The scale of distortion is set to 0.05.
- **Color Jitter**: We use the ColorJitter class provided in PyTorch. We set the range of brightness as [0.0, 0.2], that of contrast as [0, 0.3], that of saturation [0, 0.2], that of hue [0.0, 0.1].
- **CutMix**: As in Sliding Window, we cut a 112 × 112 window from an image and paste it to another 224 × 224 image, then move the window randomly.

We understand that these predefined parameters are not optimal and there is room for further consideration. We plan to conduct exhaustive experiments and develop a framework that does not rely on hand-crafted augmentations.

```python
import random

transform_list = [
    "Identity", "Sliding Window", "Zoom-in", "Zoom-out",
    "Fade-in", "Fade-out", "Affine Transformation",
    "Perspective Transformation", "Color Jitter", "CutMix",
]

def generate_pseudo_motion(image, T):
    """Pseudo Video Generator.

    Args:
        image: Input image.
        T: The number of frames in a video.
    """
    transform = random.choice(transform_list)
    params = transform.get_random_parameters()

    video = [image]
    previous_frame = image
    for _ in range(T - 1):
        transformed_frame = transform(previous_frame, params)
        video.append(transformed_frame)
        previous_frame = transformed_frame

    return video
```

Fig. 1: Python pseudo-code for Pseudo Motion Generator (PMG).

E Examples of Pseudo-motion Videos

Fig. 2 shows the examples of pseudo-motion videos generated from three synthetic image datasets; FractalDB [9], Shaders1k [2], and Visual Atom [19]. Although the appearance and motions in these videos differ from real videos, they exhibit a wide range of motion and appearance patterns. This variety enables VideoMAE to learn effectively. Specifically, pre-training with pseudo-motion videos generated from Shaders1k improves the model's performance compared to pre-training with those from the other sources. This improvement is attributed to the videos from Shaders1k having a clear correspondence of patches between frames, which suits for VideoMAE.

F Quantitative Results of Our Framework

To verify that VideoMAE successfully learns the reconstruction task, we visualized its output results on HMDB51 and UCF101. We compared the outputs of three models: (i) VideoMAE trained on real videos from each dataset, (ii) VideoMAE trained on pseudo-motion videos generated from frames on each video dataset, and (iii) VideoMAE trained on pseudo-motion videos from Shaders1k. Fig. 3 and Fig. 4 shows the results for HMDB51 and UCF101, respectively. The inputs for these models were sampled from the test set, which was not used for pre-training. Despite not being trained on real videos, VideoMAE trained on Shaders1k manages to achieve a reasonable level of accuracy in reconstructing real videos. This suggests that the method can roughly capture the complex motion and shape characteristics of the real world.

However, compared to VideoMAE trained on real videos, VideoMAE trained on pseudo-motion videos struggles with the reconstruction of finer details. This issue likely arises because our PMG applies image transformations globally, hindering its ability to learn fine-grained motions. Consequently, our framework exhibits lower performance in classifying certain fine-grained actions, compared to VideoMAE trained on real videos (See Sec. G).

G Failure Cases

We further analyzed the failure cases of our framework compared to VideoMAE when trained with real videos. For this analysis, we evaluated three models: (i) VideoMAE trained with pseudo-motion videos by Identity (no-motion videos), (ii) VideoMAE trained with pseudo-motion videos by Affine Transformation and Zoom-in/out combined with Mixup. (iii) VideoMAE trained with real videos.

Fig. 5 presents the accuracy per class on HMDB51. Between model (i) and (ii), model (ii) demonstrated improved performance of actions such as 'cartwheel', 'sit', and 'stand', which rely on motion information for recognition. However, in the comparison between model (ii) and (iii), we found that model (ii) struggled to classify actions like 'kiss', 'push', 'shake hands', and 'wave', which involve more subtle and fine-grained motion.

Fig. 6 shows the accuracy per class on UCF101. As in the patterns observed in HMDB51, model (ii) improved the performance in classes like 'BodyWeightSquats', 'CleanAndJerk', 'JumpRope' and 'YoYo', where videos lack object and background cues. Additionally, model (ii) successfully differentiated between action classes involving similar objects, for instance, 'BasketballDunk' versus 'Basketball', and 'HammerThrow' versus 'Hammering'. However, in comparison between model (ii) and (iii), we found it was difficult for model (ii) to recognize more fine-grained actions such as 'Handstand Walking', 'Nunchucks', 'PullUps', and 'WallPushups'.

Our framework struggles to capture fine-grained motion information. since our PMG applies hand-crafted image transformations globally. Consequently, the model trained by our framework has difficulty recognizing fine-grained actions, representing one of the limitations of our framework. Addressing this issue will be a priority for our future work.

H Linear Probing

Another limitation of our framework is that our framework does not learn high-level semantic features, because our framework focuses on low-level features and does not utilize labels during pre-training, unlike PPMA [26]. This limitation leads to lower performance in the linear probing settings, where the weights of the encoder are frozen while only the linear layer is trained (Tab. 3). Moreover, it is challenging to extend our framework to other tasks like video-text retrieval

Table 3: Results on SynAPT benchmark in the linear probing setting. [†] Results reported in [11].

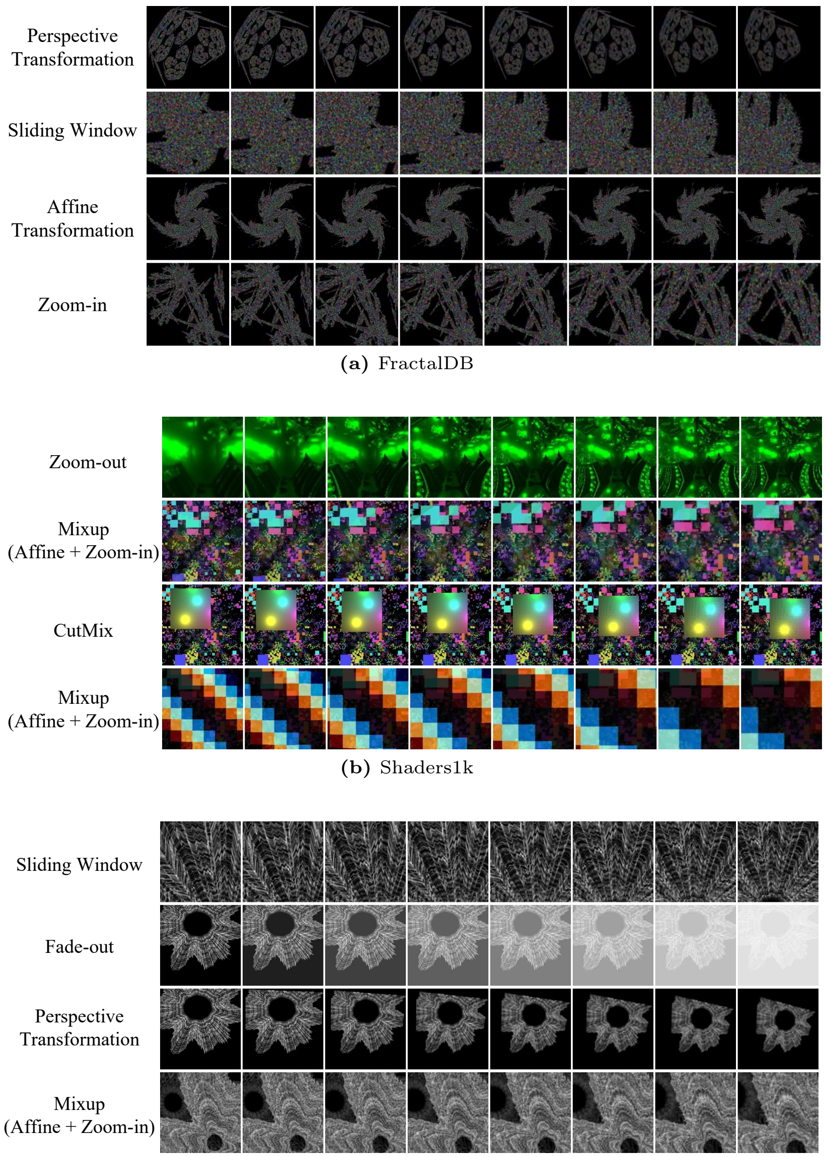
Method	Pre-training			Downstream Tasks					
	Dataset	#data	labels	UCF101	HMDB51	MiniSSV2	Diving48	IkeaFA	UAV-H
TimeSformer[†]	IN-21k +Synthetic	150k	✓	82.1	49.2	21.2	19.2	45.5	13.8
PPMA [26]	NH-Kinetics +Synthetic	300k	✓	88.4	64.9	34.9	21.9	57.7	19.3
Ours	Shaders1k	100k		42.5	28.0	10.3	6.4	33.1	1.1

and video captioning, without additional training or extra labeled data. We will also tackle this issue in future work.

References

1. Bao, H., Dong, L., Piao, S., Wei, F.: Beit: Bert pre-training of image transformers. arXiv preprint arXiv:2106.08254 (2021)
2. Baradad, M., Chen, R., Wulff, J., Wang, T., Feris, R., Torralba, A., Isola, P.: Procedural image programs for representation learning. Advances in Neural Information Processing Systems **35**, 6450–6462 (2022)
3. Chen, X., Xie, S., He, K.: An empirical study of training self-supervised vision transformers. arXiv preprint arXiv:2104.02057 **2**(5), 6 (2021)
4. 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)
5. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6202–6211 (2019)
6. 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)
7. Hoffer, E., Ben-Nun, T., Hubara, I., Giladi, N., Hoefler, T., Soudry, D.: Augment your batch: Improving generalization through instance repetition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8129–8138 (2020)
8. Huang, G., Sun, Y., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14. pp. 646–661. Springer (2016)
9. Kataoka, H., Okayasu, K., Matsumoto, A., Yamagata, E., Yamada, R., Inoue, N., Nakamura, A., Satoh, Y.: Pre-training without natural images. In: Proceedings of the Asian Conference on Computer Vision (2020)
10. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)

11. Kim, Y.w., Mishra, S., Jin, S., Panda, R., Kuehne, H., Karlinsky, L., Saligrama, V., Saenko, K., Oliva, A., Feris, R.: How transferable are video representations based on synthetic data? Advances in Neural Information Processing Systems **35**, 35710–35723 (2022)
12. 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)
13. Li, T., Liu, J., Zhang, W., Ni, Y., Wang, W., Li, Z.: Uav-human: A large benchmark for human behavior understanding with unmanned aerial vehicles. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16266–16275 (2021)
14. Li, Y., Li, Y., Vasconcelos, N.: Resound: Towards action recognition without representation bias. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 513–528 (2018)
15. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
16. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
17. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
18. 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)
19. Takashima, S., Hayamizu, R., Inoue, N., Kataoka, H., Yokota, R.: Visual atoms: Pre-training vision transformers with sinusoidal waves. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18579–18588 (2023)
20. Tong, Z., Song, Y., Wang, J., Wang, L.: Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. arXiv preprint arXiv:2203.12602 (2022)
21. Toyer, S., Cherian, A., Han, T., Gould, S.: Human pose forecasting via deep markov models. In: 2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA). pp. 1–8. IEEE (2017)
22. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks for action recognition in videos. IEEE transactions on pattern analysis and machine intelligence **41**(11), 2740–2755 (2018)
23. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7794–7803 (2018)
24. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6023–6032 (2019)
25. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)
26. Zhong, H., Mishra, S., Kim, D., Jin, S., Panda, R., Kuehne, H., Karlinsky, L., Saligrama, V., Oliva, A., Feris, R.: Learning human action recognition representations without real humans. arXiv preprint arXiv:2311.06231 (2023)



(a) FractalDB

(b) Shaders1k

(c) VisualAtom

Fig. 2: Examples of pseudo-motion videos generated from synthetic image datasets.

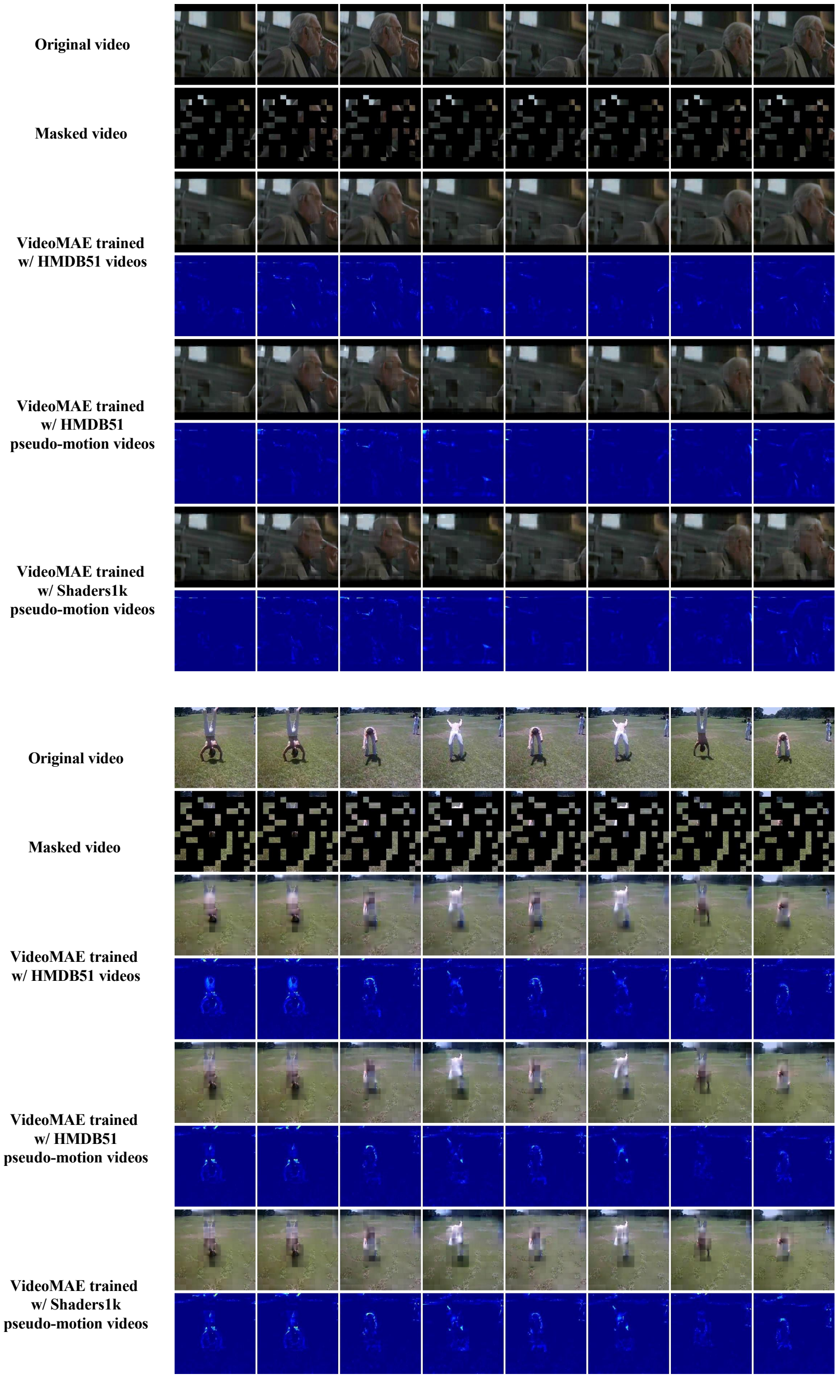

Original video
Masked video
VideoMAE trained w/ HMDB51 videos
VideoMAE trained w/ HMDB51 pseudo-motion videos
VideoMAE trained w/ Shaders1k pseudo-motion videos
Original video
Masked video
VideoMAE trained w/ HMDB51 videos
VideoMAE trained w/ HMDB51 pseudo-motion videos
VideoMAE trained w/ Shaders1k pseudo-motion videos

Fig. 3: Visualization of outputs and loss heatmaps for VideoMAE on HMDB51. The mask ratio is set as 75%. Loss heatmaps are normalized per frame.

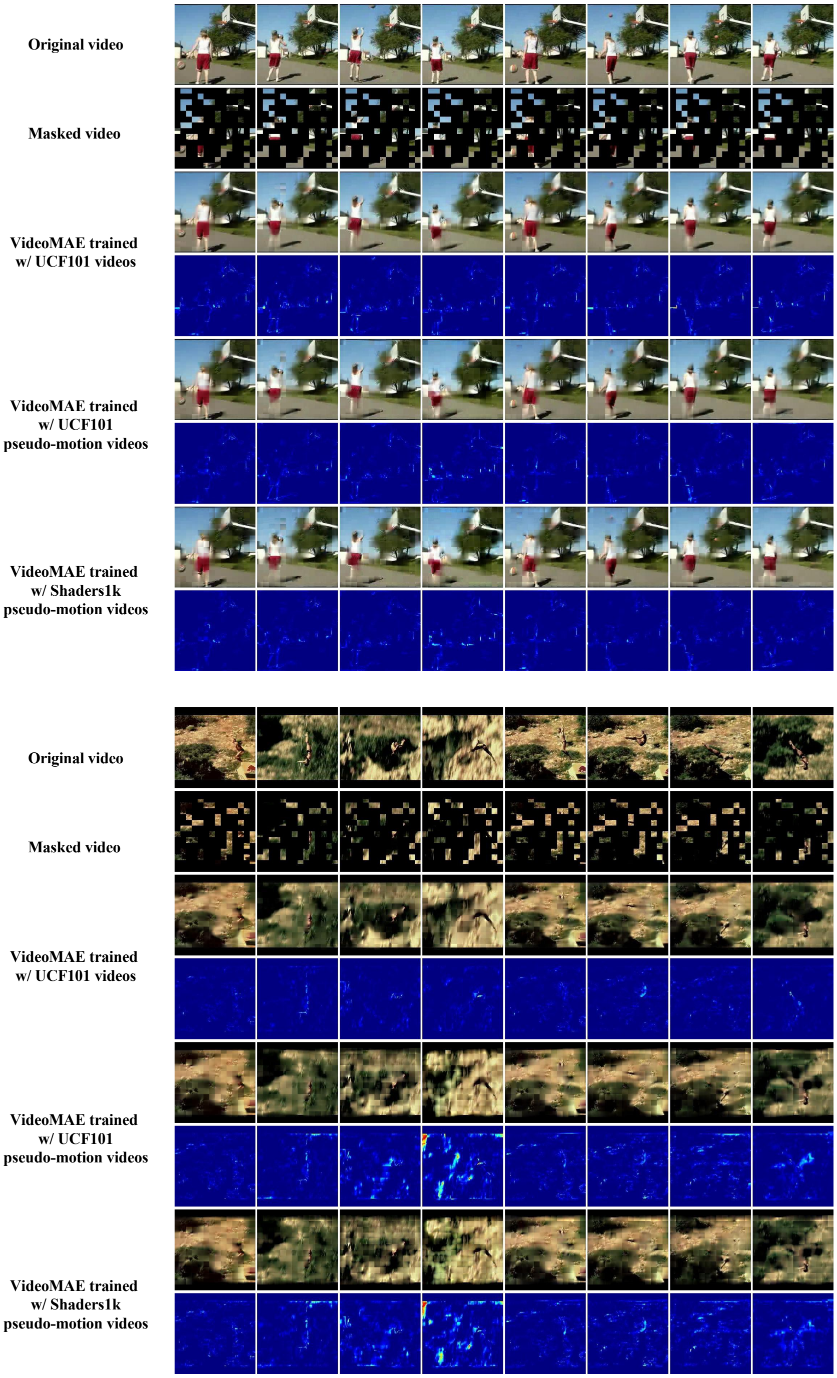

Original video
Masked video
VideoMAE trained w/ UCF101 videos
VideoMAE trained w/ UCF101 pseudo-motion videos
VideoMAE trained w/ Shaders1k pseudo-motion videos
Original video
Masked video
VideoMAE trained w/ UCF101 videos
VideoMAE trained w/ UCF101 pseudo-motion videos
VideoMAE trained w/ Shaders1k pseudo-motion videos

Fig. 4: Visualization of outputs and loss heatmaps for VideoMAE on UCF101. The mask ratio is set as 75%. Loss heatmaps are normalized per frame.

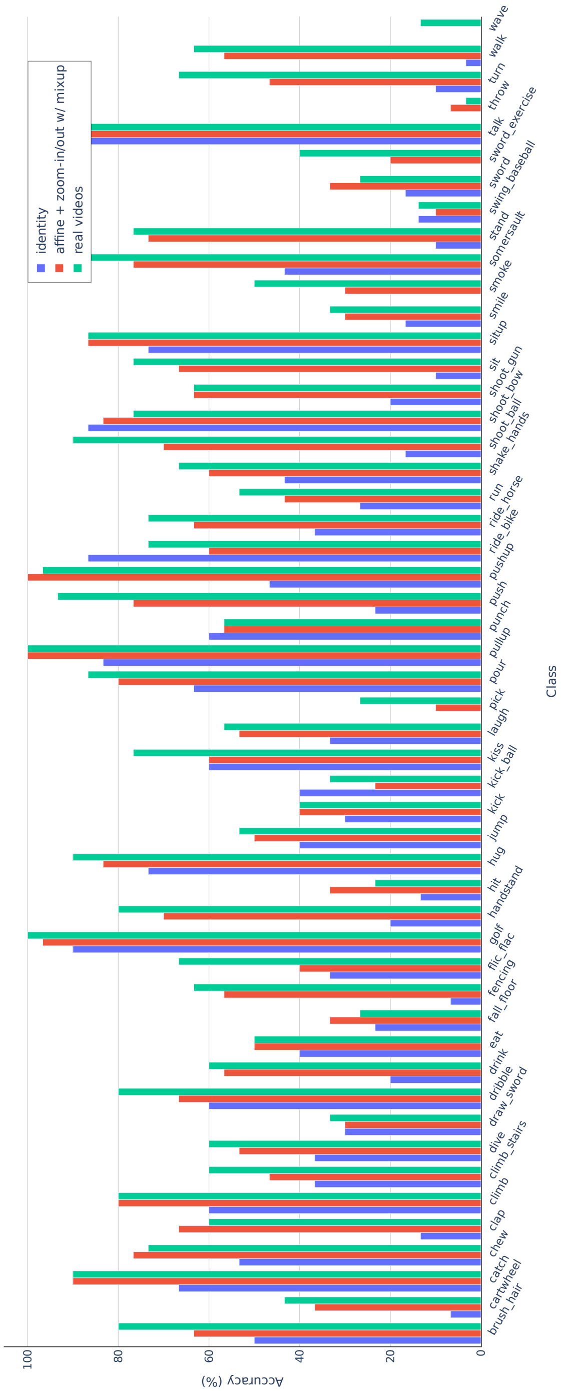


Fig. 5: Comparison of accuracy per class for each model on HMDB51.

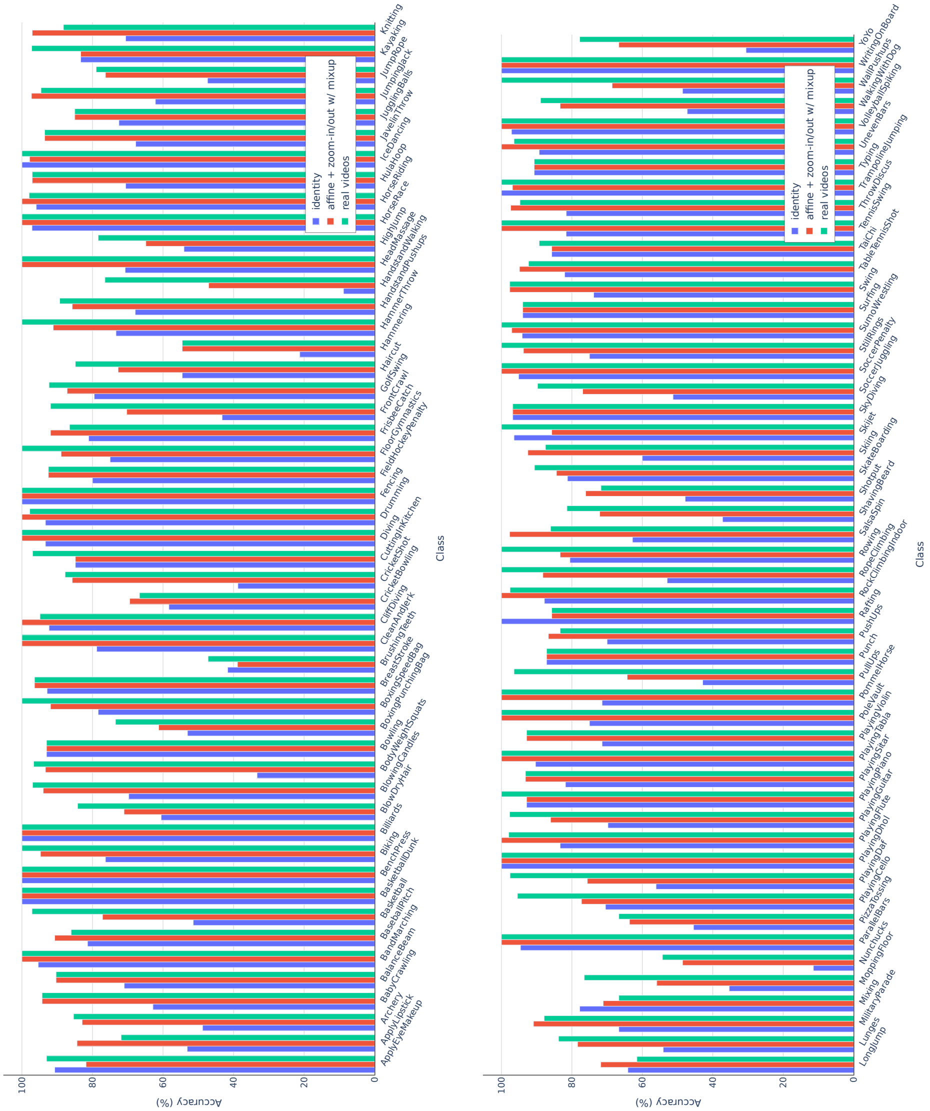


Fig. 6: Comparison of accuracy per class for each model on UCF101.