M^{3}Video: Masked Motion Modeling for Self-Supervised Video Representation Learning

Anonymous authors
Paper under double-blind review

Abstract

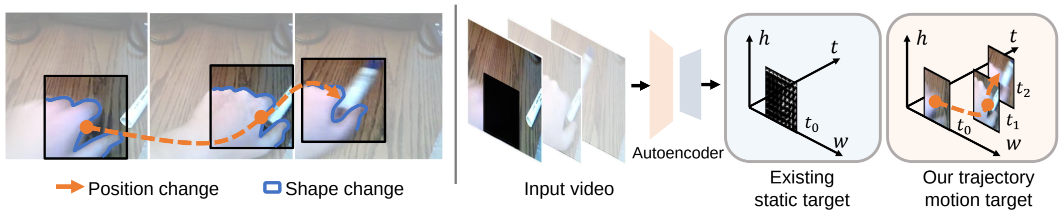
We study self-supervised video representation learning that seeks to learn video features from unlabeled videos, which is widely used for video analysis as labeling videos is labor-intensive. Current methods often mask some video regions and then train a model to reconstruct spatial information in these regions (*e.g.*, original pixels). However, the model is easy to reconstruct this information by considering content in a single frame. As a result, it may neglect to learn the interactions between frames, which are critical for video analysis. In this paper, we present a new self-supervised learning task, called **Masked Motion Modeling** (**M^{3}Video**), for learning representation by enforcing the model to predict the motion of moving objects in the masked regions. To generate motion targets for this task, we track the objects using optical flow. The motion targets consist of position transitions and shape changes of the tracked objects. To predict these trajectory motion targets, the model has to consider multiple frames comprehensively. Besides, to help the model capture fine-grained motion details, we enforce the model to predict trajectory motion targets in high temporal resolution based on a video in low temporal resolution. After pre-training using our M^{3}Video task, the model is able to anticipate fine-grained motion details even taking a sparsely sampled video as input. We conduct extensive experiments on four benchmark datasets to evaluate the effectiveness of our M^{3}Video. Remarkably, when doing pre-training with 400 epochs, we improve the accuracy from 67.6% to 69.2% and from 78.8% to 79.7% on Something-Something V2 and Kinetics-400 datasets, respectively.

1 Introduction

Video representation learning, which aims to extract powerful features to describe a video, plays a critical role in video analysis like action recognition (Yan et al., 2022), action localization (Chen et al., 2019), video retrieval (Bain et al., 2021), *etc*. Recently, vision transformers have shown their superiority in representing video, taking advantage of their spatial-temporal modeling using a flexible attention mechanism. However, training a vision transformer requires large-scale labeled data (Touvron et al., 2021), which is infeasible in most cases due to the high complexity of video annotation. How to perform self-supervised video representation learning using unlabeled video only has been a prominent research topic (Chen et al., 2021; Benaim et al., 2020; Luo et al., 2020).

Recently, the success of mask-and-predict tasks in NLP (Brown et al., 2020; Devlin et al., 2018) points out a feasible way for training a transformer model in a self-supervised manner. Prior works (Bao et al.; He et al., 2022; Zhou et al., 2022) have successfully introduced this task to pre-train an image transformer. They mask some image regions randomly and reconstruct the information within these regions. The reconstruction targets vary in different works, including raw RGB pixels (He et al., 2022), hand-crafted local patterns (Wei et al., 2022) and discrete VQ-VAE embedding (Bao et al.). All of these targets represent static appearance information in the image. Based on these successes, some researchers (Wang et al.; Wei et al., 2022; Tong et al., 2022) attempt to extend the mask-and-predict task to the video domain, where they mask 3D video regions and reconstruct appearance information as done in the image domain.

However, simply using the appearance reconstruction target for learning video representation suffers from two limitations. **First**, the model is easy to finish this mask-and-predict task by only utilizing appearance clues in every single frame. This can be proven by the fact that the mask-and-predict



(a) Two key factor to recognizing an action

(b) Static target vs. trajectory motion target

Figure 1: **Illustration of motion target for M^{3}Video.** (a) Motion information contains position change (marked by orange arrows) and shape change (marked by blue contour) over time. (b) Existing static target (*e.g.*, HOG) can not represent these changes. In contrast, we propose a trajectory motion target to better represent action.

task can be well solved in the image domain (He et al., 2022; Wei et al., 2022), where the model can reconstruct the masked region depending on the unmasked information in a single image. In this sense, the transformer may neglect to learn the interaction between frames, which is critical for representing an action. **Second**, during the process of masking regions, existing works (Tong et al., 2022; Wei et al., 2022) often sample frames sparsely with a fixed stride, and then mask some regions in these sampled frames. The reconstruction targets only contains information in these sparse sampled frames, and thus are hard to provide supervision signals for learning fine-grained motion details. Providing the model with more motion details is critical to distinguish different actions (Bertasius et al., 2021; Arnab et al., 2021).

In this paper, we aim to design a new prediction target to tackle these two issues. Figure 1(a) shows two key factors to recognizing an action, *i.e.*position change and shape change. From the position change of the hand, we realize the person would move his/her hand toward the pen, and from the shape change (pinching fingers), we are aware that he/she attend to grab something. We believe that anticipating these changes helps the model better understand an action.

Based on this observation, instead of predicting the appearance features of the masked regions, we design a *trajectory motion target*, which represents impending position and shape changes, for the mask-and-predict task. Specifically, for each masked patch, we track the moving object in different frames to generate a trajectory, as shown in Fig 1(b). This trajectory contains information in two aspects: the position features that describe relative movement; and the shape features that describe shape changes of the tracked object along the trajectory. To predict this trajectory motion target, the model has to learn the correlation and interaction of objects among different frames and try to estimate their accurate motions. We name the proposed mask-and-predict task **Masked Motion Modeling** (M^{3}Video). Moreover, to help the model learn fine-grained motion details, we further propose to interpolate the trajectory motion targets where the model is asked to reconstruct spatially and temporally dense trajectory motion targets from sparsely sampled video input. This is inspired by the video frame interpolation task (Xiang et al., 2020) where a deep model is able to reconstruct dense video at pixel level from sparse video input. Different from it, we aim to reconstruct the fine-grained motion details of moving objects, which has higher-level motion information and is helpful for understanding actions. Experiments on four datasets prove that the representations learned from the proposed mask-and-predict task achieve state-of-the-art performance on downstream action recognition tasks. Our main contributions are as follows:

- We propose a trajectory motion target and a masked motion modeling (M^{3}Video) task for self-supervised video representation learning. Compared with existing static targets, modeling the proposed motion target endows the model to anticipate the movement and shape change of moving objects, which is critical for understanding actions.
- In the M^{3}Video task, we propose to reconstruct the spatially and temporally dense trajectory motion target from the sparse video input. Through interpolating dense trajectories, the model learns fine-grained motion details without increasing computational costs.
- We evaluate our M^{3}Video on four benchmark datasets. Compared to static targets and short-term motion targets, reconstructing the proposed trajectory motion target achieve state-of-the-art performance, increasing action recognition accuracy from 67.6% to 69.2% and from 78.8% to 79.7% on Something-Something V2 and Kinetics-400 datasets, respectively.

2 Related Work

Self-supervised Video Representation Learning. Self-supervised video representation learning aims to learn discriminative video features for various downstream tasks in the absence of accurate video labels. Most of the existing methods try to design an advanced pretext task like predicting the temporal order of shuffled video crops (Xu et al., 2019), perceiving the video speediness (Benaim et al., 2020; Chen et al., 2021) or solving puzzles (Luo et al., 2020; Jing et al., 2018). In addition, instance discrimination is also widely used in this domain CVRL,LSTCL,SVT,BE,ASCNet,FlowCoTraining. With the help of contrastive learning and the Siamese network, constraining the consistency between different augmentation views brings significant improvement in self-supervised video learning (Qian et al., 2021; Wang et al., 2021c; Ranasinghe et al., 2022; Wang et al., 2021b), and some of these methods focus on mining hard positive samples in different perspectives (Huang et al., 2021; Han et al., 2020). Besides, Tracking video objects' movement is also used in self-supervised learning (Wang & Gupta, 2015; Wang et al., 2021a), however, they only require the model to figure out the position and size changes of a specific video patch. Different from these methods, our proposed M^{3}Video trace the fine-grained movement and shape changes of different parts of objects in the video, hence resulting in a superior video representation.

Mask Modeling for Vision Transformer. He et al. (2022) and Bao et al. show two different mask image modeling paradigms and both achieve state-of-the-art results. Because of the similarity between image and video, these two paradigms are also suitable for pre-training video transformers (Tong et al., 2022; Wang et al.). In Tong's work (Tong et al., 2022), the video volume is masked by some random tubes, and the training objective is to regress the RGB pixels located in the tubes. In contrast, Wang et al. use a pre-trained VQ-VAE tokenizer (Ramesh et al., 2021) for both video and image modalities to generate discrete visual tokens, which aims to release the model from fitting short-range dependencies and high-frequency details. Though the VQ-VAE tokenizer provides the semantic level signal for each spatial-temporal patch separately, pre-training such a tokenizer requires an unbearable number of data and computation costs (Ramesh et al., 2021), which is inefficient. Inspired by the previous success of traditional hand-craft image descriptors, Wei et al. (2022) directly predicts features of the masked video contents and found that the Histogram of Gradient (HOG) feature is particularly a strong target for the proposed objective. Existing Mask Video Modeling methods only take into account static information in each video frame, thus the model can speculate the masked area by watching the visible area in each frame independently and failed to learn multi-frame interactions. Different from the video prediction methods (Patraucean et al., 2016; Srivastava et al., 2015; Recasens et al., 2021; Gupta et al., 2022) that predict the future frames in pixel or latent space, our Masked Motion Modeling task predicts incoming fine-grained motion in masked video regions, including position changes and shape changes.

3 Proposed Method

We first revisit the current masked video modeling task for self-supervised video representation learning, followed by a toy experiment showing that the task can be easily finished by only considering the content in a single frame, and thus the existing works may neglect to learn action features (c.f. Section 3.1). Then, we introduce our masked motion modeling, where we replace the static appearance reconstruction target with a motion target to help the model anticipate impending actions (c.f. Section 3.2).

3.1 Revisiting Masked Video Modeling

Given a video clip sampled from a video, self-supervised video representation learning aims to learn a feature encoder $f_{enc}(\cdot)$ that maps the clip to its corresponding feature that best describes the video. Existing masked video modeling methods (Tong et al., 2022; Wei et al., 2022) attempt to learn such a feature encoder through a mask-and-predict task. Specifically, the input clip is first divided into multiple non-overlapped 3D patches. Some of these patches are randomly masked and the remaining patches are fed into the feature encoder, followed by a decoder $f_{dec}(\cdot)$ to reconstruct the information in the masked patches. The reconstruction targets vary in different works (*e.g.*, raw pixel in VideoMAE (Tong et al., 2022) and HOG in MaskFeat (Wei et al., 2022)). However, these

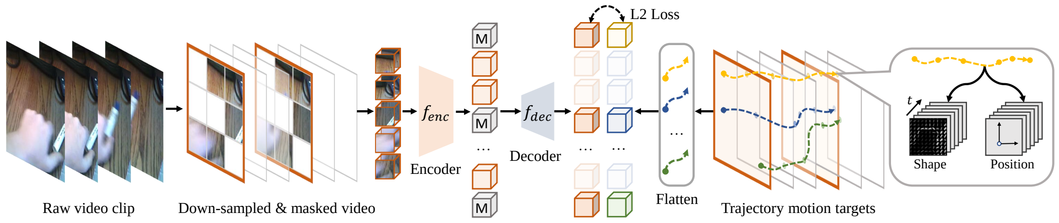


Figure 2: **Overview of M^{3}Video**. Given a sparsely sampled video, we divide it into several patches and randomly mask out some of them. An autoencoder is utilized to predict motion information, *i.e.*, a trajectory motion target containing position changes and shape changes of moving objects, in the masked regions.

targets share a common characteristic where they all represent static appearance information of the masked patches. This static information can be easily inferred from other unmasked patches in the same single frames (He et al., 2022). In this sense, the model may neglect to learn the interaction between multiple frames, which is critical for learning action features.

To evaluate whether these static targets can be inferred from a single frame, we conduct a toy experiment. Specifically, we randomly sample a frame and replace all other frames with this one, which leads to a static video. We mask the same regions in all frames to ensure that the masked content will not appear in other frames. We then feed the static video into the model to perform the mask-and-predict task. In this sense, the model has to reconstruct the target (raw pixel in this experiment) based on the unmasked content in a single frame. In Fig 3, we show the reconstruction L2 error curve w.r.t. training epoch. The model that regresses pixels using the single frame content performs comparatively with the model using multi-frame video (two blue curves). This suggests that the masked pixel can be well reconstructed by conditioning on the single frame content. We also conduct the same experiments to reconstruct our proposed motion target (c.f. Section 3.2.2). We found that the gap between single-frame and multi-frame input (orange curves) is large. This suggests the model has to consider the content from different frames for reconstructing our proposed target. This is helpful for teaching the model to learn action features from the interaction between frames.

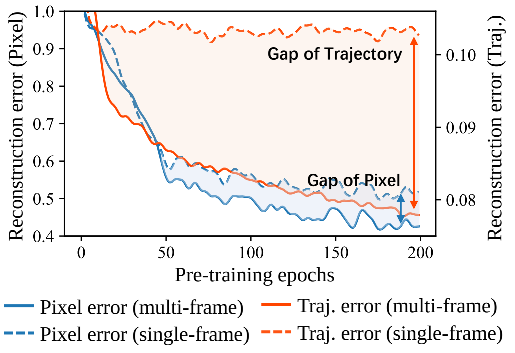


Figure 3: **Comparison of reconstruction errors.** Pixel can be reconstructed by single frame content while trajectory can not.

3.2 M^{3}VIDEO: MASKED MOTION MODELING

3.2.1 GENERAL SCHEME OF M^{3}VIDEO

Our M^{3}Video task follows the mask-and-predict paradigm of the existing masked video modeling task (Tong et al., 2022) but replaces the static appearance target with a motion target that represents the position and shape change of objects. As shown in Fig 2, a video clip is sampled uniformly from a video and is divided into a number of non-overlapped 3D patches of size $t \times h \times w$. We follow Tong et al. (2022) to use the tube masking strategy, where the masking map is the same for all frames, to mask a portion of patches. For computation efficiency, we follow He et al. (2022) to only feed the unmasked patches (and their positions) to the encoder. The output features together with learnable $[\mathrm{MASK}]$ tokens are fed to a decoder to reconstruct motion target $\mathbf{z}$ in the masked patches. The training loss for this mask-and-predict task is

$$\mathcal{L} = \sum_{i \in \mathcal{I}} |\mathbf{z}_i - \hat{\mathbf{z}}_i|^2, \tag{1}$$

where $\hat{\mathbf{z}}$ is the predicted motion target, and $\mathcal{I}$ is the index set of motion targets in all masked patches.

Motivated by the fact that we humans recognize actions by perceiving position changes and shape changes of moving objects, we extract these two types of information to build the motion target. By predicting this target, the model is endowed with the ability to anticipate actions from the interaction between frames. Another important characteristic of the proposed motion target is that it contains fine-grained motion information extracted at raw video rate. This fine-grained motion information provides the model with a supervision signal to anticipate fine-grained action from sparse video input. In the following, we will introduce the proposed motion target in detail.

3.2.2 Trajectory Motion Target for Mask Modeling

The motion of moving objects can be represented in various ways such as optical flow (Farnebäck, 2003), histograms of optical flow (HOF) (Laptev et al., 2008), and motion boundary histograms (MBH) (Laptev et al., 2008). However, these descriptors can only represent short-term motion between two adjacent frames. We hope our motion target represents long-term motion, which is critical for video representation. To this end, inspired by Wang et al. (2013), we first track the moving object in the following L frames to cover a longer range of motion, resulting in a trajectory $\mathbf{T}$, *i.e.*,

$$\mathbf{T} = (\mathbf{p}_t, \mathbf{p}_{t+1}, \cdots, \mathbf{p}_{t+L}), \tag{2}$$

where $\mathbf{p}_t = (x_t, y_t)$ represents a point located at (x_t, y_t) of frame t, and $(\cdot, \cdot)$ indicates the concatenation operation. Along this trajectory, we fetch the position features $\mathbf{z}^p$ and shape features $\mathbf{z}^s$ of this object to compose a trajectory motion target $\mathbf{z}$, *i.e.*,

$$\mathbf{z} = (\mathbf{z}^p, \mathbf{z}^s). \tag{3}$$

The position features are represented by the position transition relative to the last time step, while the shape features are the HOG descriptors of the tracked object in different time steps.

Tracking objects using spatially and temporally dense trajectories. Some previous works (Anjum & Cavallaro, 2008) try to use one trajectory to represent the motion of an individual object. In contrast, Wang et al. (2013) points out that tracking spatially dense feature points sampled on a grid space performs better since it ensures better coverage of a video. We follow Wang et al. (2013) to use spatially dense grid points as the initial position of each trajectory. Specifically, we uniformly sample K points in a masked patch of size $t \times h \times w$, where each point indicates a part of an object. For each point, we track it through temporally dense L frames according to the dense optical flow, resulting in K trajectories. In this way, the trajectory is able to capture spatially and temporally fine-grained motion information of objects as reconstruction targets for the mask-and-predict task.

As a comparison, the reconstruction target in existing works is often extracted from temporally sparse videos sampled with a large sample stride $s > 1$. The model takes as input a sparse video and predict these sparse targets for learning video representation. Different from these works, our model also takes as input sparse video but we push the model to interpolate trajectory motion targets containing fine-grained motion information. This simple trajectory interpolation task does not increase the computational cost of the video encoder but helps models learn more fine-grained action information even given sparse video input. More details about flow calculating and tracking can be found in Appendix A.

Representing position features. Given a trajectory $\mathbf{T}$ consisting of the tracked object position at each frame, we are more interested in the related movement of objects instead of their absolute location. Consequently, we represent the position features with related movement between two adjacent points $\Delta\mathbf{p}_t = \mathbf{p}_{t+1} - \mathbf{p}_t$, *i.e.*,

$$\mathbf{z}^p = (\Delta\mathbf{p}_t, ..., \Delta\mathbf{p}_{t+L-1}), \tag{4}$$

where $\mathbf{z}^p$ is a $L \times 2$ dimensional feature. As each patch contains K position features, we concatenate and normalize them as position features part of the trajectory motion target.

Representing shape features. Besides embedding the movement, the model also needs to be aware of the shape changes of objects to recognize actions. Inspired by Dalal & Triggs (2005), we use histograms of oriented gradients (HOG) with 9 bins to describe the shape of objects.

Compared with existing works (Wei et al., 2022; Tong et al., 2022) that reconstruct HOG in every single frame, we are more interested in the dynamic shape changes of an object, which can better

Table 1: **Comparison of different targets.** Our trajectory motion target outperforms all the other targets.

Target	Acc@1	Acc@5
Pixel	61.1	86.6
HOG	61.1	86.9
HOG + Flow	60.9	86.4
HOG + HOF	61.3	86.6
HOG + MBH	61.4	86.8
HOG + Traj. w/o shape	63.5	88.2
Traj. (ours)	**64.1**	**88.4**

Table 2: **Effectiveness of trajectory interpolation.** Interpolating the trajectory motion target brings significant improvement compared to the baselines.

Target	Interpolation	Acc@1	Acc@5
HOG	✗	61.1	86.9
HOG	✓	60.9	86.8
HOG + MBH	✗	61.3	86.8
HOG + MBH	✓	61.4	86.8
Traj. (ours)	✗	62.2	87.5
Traj. (ours)	✓	**64.1**	**88.4**

represent action in a video. To this end, we follow Wang et al. (2013) to calculate trajectory-aligned HOG, consisting of HOG features around all tracked points in a trajectory, *i.e.*

$$\mathbf{z}^s = (\mathrm{HOG}(\mathbf{p}_t), ..., \mathrm{HOG}(\mathbf{p}_{t+L-1})), \tag{5}$$

where $\mathrm{HOG}(\cdot)$ is the HOG descriptor and $\mathbf{z}^s$ is a $L \times 9$ dimensional feature. Also, as one patch contains K trajectories, we concatenate K trajectory-aligned HOG features and normalize them as the shape features part of the trajectory motion target.

4 Experiments

Datasets. We evaluate our M^{3}Video on two large-scale datasets, namely Kinetics-400 (K400) (Carreira & Zisserman, 2017) and Something-Something V2 (SSV2) (Goyal et al., 2017). The K400 dataset contains 240k training videos and 20k validation videos that span 400 action categories. Compared with K400, SSV2 involves more human-object interactions. Recognizing these actions requires the model to focus more on the motion instead of background information. SSV2 contains 168k training videos and 24k validation videos in 174 action categories. We also evaluate on smaller action recognition datasets, *i.e.*, UCF101 (Soomro et al., 2012) (9.5k/3.7k videos for train/test) with 101 classes and HMDB51 (Kuehne et al., 2011) (3.5k/1.5k videos for train/test) with 51 classes.

Implementation details. Unless otherwise stated, we follow previous trails (Tong et al., 2022) and feed the model a 224×224 16-frame clip with strides 4 and 2 for K400 and SSV2 respectively. We use vanilla ViT-Base (Tong et al., 2022) as the backbone in all experiments. As for the motion target, we set the trajectory length to $L = 6$ and the number of trajectories in each patch to $K = 4$. Our models are trained on 16 NVIDIA-3090 GPUs. See Appendix for further details.

4.1 Ablation Study

To reduce training time, for all experiments in the ablation study, we randomly select 25% of data in the Something-Something V2 training set for pre-training. We pre-train models for 200 epochs and fine-tune it for 50 epochs.

4.1.1 Effectiveness of Trajectory motion target

Baselines: We compare our proposed trajectory motion target with two types of static targets and three types of short-term motion targets.

- **Static target baselines**: 1) **HOG**: predicting HOG features (Dalal & Triggs, 2005) in the middle frame of each 3D patch (Wei et al., 2022). 2) **Pixel**: predicting all the pixels of each 3D patch (He et al., 2022; Tong et al., 2022; Feichtenhofer et al., 2022).
- **Short-term motion target baselines**: 1) **Flow**: predicting the dense optical flow map of the masked patches. 2) **HOF**: predicting the histogram of optical flow orientation. 3) **MBH**: predicting the motion boundary histogram, which is the second-order local statistic of optical flow.

Table 3: **Trajectory length.**

Length	Acc@1
$L = 0$	61.1
$L = 2$	61.4
$L = 4$	63.2
$L = 6$	**64.1**
$L = 8$	63.3

Table 4: **Trajectories density.** Dense trajectories work the best.

Trajectory density	Acc@1
Dense	**64.1**
Sparse (Mean)	60.2
Sparse (Max)	61.8

Table 5: **Trajectory normalization.** Patch normalization performs better.

Setting	Acc@1
w/o patch norm.	62.3
w/ patch norm.	**64.1**

Table 6: **Sampling stride.** Sampling stride $s = 2$ on SSV2 performs the best.

Stride	Acc@1
$s = 4$	62.9
$s = 3$	63.4
$s = 2$	**64.1**
$s = 1$	62.7

Table 7: **Spatial masking.** Tube masking with a 70% ratio performs the best.

Ratio	Type	Acc@1
90%	Tube	62.7
70%	Tube	**64.1**
40%	Tube	61.3
40%	Cube	61.2

Table 8: **Decoder setting.** Sep. means adopt two parallel decoder. A slight decoder performs the best

Type	Depth	FLOPs	Acc@1
Joint	1	51G	**64.1**
	2	55G	63.4
Sep.	1	56G	63.4
	2	65G	63.3

Results: In Tab 1, two types of static targets (*i.e.*, Pixel and HOG) perform similarly. Based on the HOG baseline, adding short-term motion targets brings slight improvement (at least by 0.2%) except for adding a Flow target (decreases by 0.2%). We suspect that predicting the dense flow map for each pixel is too hard for a model and thus harms the performance. Instead, the histogram version of flow (*i.e.*, HOF and MBH) help to reduce the prediction ambiguity (Wei et al., 2022), resulting in better results. Furthermore, after adding our slight variant of trajectory motion target (*i.e.*, with position features but without shape features shown in Eq (3)), the performance improves significantly compared to both static target baselines (by 2.4%) and short-term motion target baselines (by 2.1%). On top of this variant, adding trajectory-aligned shape features further brings additional gain by 0.6%. We believe predicting both shape and position changes of moving objects helps the model to learn more long-term motion clues for recognizing actions.

4.1.2 Effectiveness of Trajectory Interpolation

In M^{3}Video, we feed temporally sparse video to the model and push it to interpolate temporally dense trajectories. We are interested in whether interpolating static target (*i.e.*, HOG) or short-term motion target (*i.e.*, MBH) benefits learning fine-grained motion details. To this end, we conduct experiments to predict dense HOG and MBH, *i.e.*, predicting these features of all frames within a masked patch even though these frames are not sampled as input. In Tab 2, interpolating HOG or MBH bring little improvement. We suspect this is because these dense targets still can not represent the continuous movement of objects because they have not explicitly tracked the moving objects. In contrast, our temporally dense trajectory motion target tracks the object densely, providing temporal fine-grained information for learning. The trajectory interpolation improves our M^{3}Video by 1.9%.

4.1.3 Ablation Study on M^{3}Video Design

In this section, we ablate to find the best settings for M^{3}Video task.

Trajectory length. Trajectory length L indicates how many frames we track an object for extracting the motion target. If $L = 0$, our method degrades to Wei et al. (2022), which only predicts HOG features in the masked patch. In Tab 3, as the length grows, the performance of M^{3}Video grows and the best performance is achieved when $L = 6$. This is because a longer trajectory provides more motion information for learning. But if the trajectory is too long, the drift of flow introduces accumulated noise when tracking the objects, which may harm the performance.

Trajectory density. In our experiments, we spatially dense track $K = 4$ trajectories as motion targets. We conduct experiments to evaluate whether it is necessary to densely track multiple trajectories in a patch. Specifically, we assume only one object exists in a patch, and thus the 4 trajectories are similar. We merge these 4 trajectories as one by averaging or selecting the most salient one. We then consider the merged trajectory as the motion target. In Tab 4, using dense trajectories signif-

Table 9: **Comparison with state-of-the-arts on Something-Something V2.**

Method	Backbone	Pre-train	Label	Epoch	Frames	GFLOPs	Param	Acc@1
TimeSformer (Bertasius et al., 2021)	ViT-B	IN-21K	✓	15	8	$196\times1\times3$	121M	59.5
ViViT-L (Arnab et al., 2021)	ViT-L	IN-21K+K400	✓	35	16	$3992\times3\times4$	311M	65.4
Motionformer (Patrick et al., 2021)	ViT-B	IN-21K+K400	✓	35	16	$370\times1\times3$	109M	66.5
SlowFast (Feichtenhofer et al., 2019)	ResNet-101	K400	✓	196	8+32	$106\times1\times3$	53M	63.1
MViT (Fan et al., 2021)	MViT-B	K400	✓	100	64	$455\times1\times3$	37M	67.7
SVT (Ranasinghe et al., 2022)	ViT-B	IN-21K+K400	✗	20	8+64	$196\times1\times3$	121M	59.6
LSTCL (Wang et al., 2021c)	Swin-B	K400	✗	200	16	$360\times5\times1$	88M	67.0
BEVT (Wang et al.)	Swin-B	K400+DALL-E	✗	150	32	$282\times1\times3$	88M	67.1
VIMPAC (Tan et al., 2021)	ViT-L	HowTo100M+DALLE	✗	100	10	N/A$\times10\times3$	307M	68.1
VideoMAE (Tong et al., 2022)	ViT-B	SSV2	✗	400	16	$180\times2\times3$	87M	67.6
M^{3}Video (Ours)	ViT-B	SSV2	✗	400	16	$180\times2\times3$	87M	69.2
M^{3}Video (Ours)	ViT-B	K400	✗	400	16	$180\times2\times3$	87M	**69.7**

icantly outperform the merged baselines. We suspect the spatial dense trajectories provide more motion information for learning.

Trajectory normalization. Due to the irregular distribution of motion and shape information in the video, the trajectory values in different patches vary a lot. To help the model better model the trajectory, we adopt the patch norm method to normalize the trajectory values in each patch into a standard normal distribution, as has also been done in He et al. (2022). The mean and variance are calculated by all $K = 4$ trajectories in a patch. Tab 5 shows the effectiveness of adopting patch norm.

Sampling stride. In order to controls the difficulty of interpolating the trajectory motion target, we ablate different temporal sampling stride leading to different level of sparse input video. With a medium sampling stride ($s = 2$), our M^{3}Video achieves the best performance. We also found that the performance decreases when the input video shares the same stride as the dense trajectory motion target ($s = 1$). One possible reason is that the model can not learn to anticipate fine-grained motion details in this case.

Spatial masking strategy. We ablated study different spatial masking strategies and ratios in Tab 7. We start from the default setting of Wei et al. (2022) that use a cube masking strategy with 40% masking ratio, then we increase the masking ratio and adopt the tube masking strategy proposed by Tong et al. (2022). Using tube masking with a medium masking ratio (70%) performs the best.

Decoder setting. We use several transformer blocks as decoder to reconstruct motion targets from extracted video features. We ablate different architecture of the decoder in Tab 8. Using a slight decoder with 1 transformer block performs the best, with the lowest computational cost. Increasing the depth of the decoder or using two parallel decoders to separately predict position and shape features of motion targets introduces more computational cost but brings little improvement.

4.2 Comparisons with State-of-the-arts

Fine-tuning. We compare our M^{3}Video with the previous state-of-the-art self-supervised and supervised methods on Something-Something V2 and Kinetics400 datasets. The results are reported in Tab 9 and Tab 10. On both dataset our method achieved state-of-the-art compared to previous self-supervised video representation learning methods, drawing the improvement by 1.6% on Something-Something V2 and 0.9% on Kinetics-400. The improvement on Kinetics-400 is relatively small, due to the dataset bias that rely more on appearance information (Wang et al., 2021b; Wang et al.; Tong et al., 2022). Besides, we found that our M^{3}Video significantly outperform the Motionformer (Patrick et al., 2021), which is equipped with a carefully designed trajectory attention mechanism to capture motion information. Last but not least, pre-training on the Kinetics-400 and transferring to the Something-Something V2 brings larger improvement, compared to directly pre-trained on Something-Something V2. This indicates that our M^{3}Video can leverage motion information in the extra videos to learn better video representation.

Our M^{3}Video also achieves state-of-the-art action recognition result on the UCF101 and HMDB51 dataset in Tab 11. It is worth noting that we only pre-trained 400 epochs on Kinetics-400, while previous state-of-the-art method (Tong et al., 2022) has pre-trained for 3200 epochs.

Table 10: **Comparison with state-of-the-arts on Kinetics-400**.

Method	Backbone	Pre-train	Label	Epoch	Frames	GFLOPs	Param	Acc@1
TimeSformer (Bertasius et al., 2021)	ViT-B	IN-21K	✓	15	8	196×1×3	121M	78.3
Motionformer	ViT-B	IN-21K	✓	35	16	370×10×3	121M	79.7
LSTCL (Wang et al., 2021c)	Swin-B	K400	✗	200	16	360×5×1	88M	81.5
BEVT (Wang et al.)	Swin-B	K400+DALL-E	✗	150	32	282×1×3	88M	76.2
VIMPAC (Tan et al., 2021)	ViT-L	HowTo100M+DALLE	✗	100	10	N/A×10 × 3	307M	77.4
SVT (Ranasinghe et al., 2022)	ViT-B	IN-21K+K400	✗	20	8+64	196×1×3	121M	78.1
VideoMAE (Tong et al., 2022)	ViT-B	K400	✗	400	16	180×5×3	87M	78.8
M^3Video (Ours)	ViT-B	K400	✗	400	16	180×5×3	87M	**79.7**

Table 11: **Comparison with state-of-the-arts on UCF101 and HMDB51**.

Methods	Backbone	Pre-train	UCF101	HMDB51
RSPNet (Chen et al., 2021)	S3D-G	K400	93.7	64.7
CVRL (Qian et al., 2021)	R152	K600	94.4	70.6
VideoMAE	ViT-B	K400	96.1	73.3
M^3Video (Ours)	ViT-B	K400	**96.1**	**75.4**

Table 12: **Linear Probing on Something-Something V2**.

Methods	Backbone	Pre-train	Acc@1
TimeSformer	ViT-B	IN-21K	14.0
SVT	ViT-B	IN-21K+K400	18.3
VideoMAE	ViT-B	SSV2	24.3
M^3Video (Ours)	ViT-B	SSV2	**29.2**

Linear probing. We further evaluate our M^3Video under liner probing setting on the Something-Something V2 as this dataset focuses more on the action. We follow Ranasinghe et al. (2022) to fix the transformer backbone and train a linear layer for 20 epochs. In Tab 12, our method significantly outperform previous contrastive-based method (Ranasinghe et al., 2022) by 10.9% and masked video modeling methods (Tong et al., 2022) by 4.9%. These results indicate that the representation learned from our M^3Video task contains more motion clues for action recognition.

4.3 Visualization of Predicted Motion

We visualize the predicted trajectory motion target in Fig 4. The model is able to reconstruct the spatially and temporally dense trajectories of different object parts from a temporally sparse input video. Even without seeing the object in the current frame (the hand being masked out), the model is able to locate it in contextual frames and accurately estimate its movement (the dense trajectories). Our prediction of HOG features has similar qualitative results to the previous work (Wei et al., 2022). Thus we do not show it in this paper. See Appendix for more visualization results.

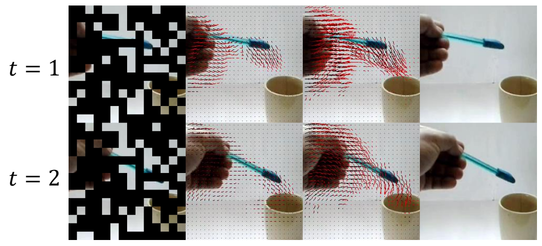


Figure 4: **Visualization of trajectory motion targets**. Each column represents: (a) Masked frames; (b) ground-truth; (c) prediction; (d) original frame. Trajectories are colored dark to light in chronological order.

5 Conclusion

We present Masked Motion Modeling (M^3Video), a simple mask-and-predict task that learns video representation by reconstructing model details of masked regions. In particular, we introduce *trajectory motion targets* as the reconstruction target, which is a hand-crafted motion descriptor that tracks moving objects densely to record their position and shape changes. By reconstructing this target, the model learns long-term fine-grained motion clues from sparse input videos. Empirical results demonstrate that our M^3Video outperforms the state-of-the-art mask-and-predict methods on Kinetics-400, Something-Something V2, UCF-101, and HMDB-51 datasets for action recognition.

REPRODUCIBILITY STATEMENT

In this work, we implement our method with ViT-Base models on four action recognition datasets, including Something-Something V2, Kinetics-400, UCF101, and HMDB51. Reproducing all the results in our paper depends on the following three aspects:

DATASET. All the adopted datasets are widely used and large-scale benchmark for video representation learning. In the first paragraph of Section 4 and Appendix A.4, we provide the details of the dataset and the download `url`.

MODEL. All the adopted datasets are widely used and large-scale benchmarks for video representation learning. In the first paragraph of Section 4 and Appendix A.4, we provide the details of the dataset and the download `url`.

PROTOCOLS OF EACH METHOD. Appendix A provides the implementation details of all compared methods and our M^{3}Video. ***The source code of M^{3}Video will be released as soon as this paper is accepted***.

REFERENCES

Nadeem Anjum and Andrea Cavallaro. Multifeature object trajectory clustering for video analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, 18:1555–1564, 2008. 5

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, pp. 6836–6846, 2021. 2, 8

Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, pp. 1728–1738, 2021. 1

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: BERT pre-training of image transformers. In *ICLR*. 1, 3, 15

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, pp. 404–417, 2006. 14

Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. Speednet: Learning the speediness in videos. In *CVPR*, pp. 9922–9931, 2020. 1, 3

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding. In *ICML*, pp. 813–824, 2021. 2, 8, 9

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020. 1

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pp. 6299–6308, 2017. 6

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, pp. 1691–1703, 2020. 15

Peihao Chen, Chuang Gan, Guangyao Shen, Wenbing Huang, Runhao Zeng, and Mingkui Tan. Relation attention for temporal action localization. *IEEE Transactions on Multimedia*, 22:2723–2733, 2019. 1

Peihao Chen, Deng Huang, Dongliang He, Xiang Long, Runhao Zeng, Shilei Wen, Mingkui Tan, and Chuang Gan. Rspnet: Relative speed perception for unsupervised video representation learning. In *AAAI*, pp. 5, 2021. 1, 3, 9

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR*, pp. 702–703, 2020. 15

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pp. 886–893, 2005. 5, 6, 14

Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, pp. 428–441, 2006. 15

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pp. 4171–4186, 2018. 1

Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *ICCV*, pp. 6824–6835, 2021. 8

Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *SCIA*, pp. 363–370, 2003. 5

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, pp. 6202–6211, 2019. 8

Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. *arXiv preprint arXiv:2205.09113*, 2022. 6

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24: 381–395, 1981. 14

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, pp. 5842–5850, 2017. 6

Agrim Gupta, Stephen Tian, Yunzhi Zhang, Jiajun Wu, Roberto Martín-Martín, and Li Fei-Fei. Maskvit: Masked visual pre-training for video prediction. *arXiv preprint arXiv:2206.11894*, 2022. 3

Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. *NeurIPS*, 33:5679–5690, 2020. 3

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pp. 16000–16009, 2022. 1, 2, 3, 4, 6, 8

Deng Huang, Wenhao Wu, Weiwen Hu, Xu Liu, Dongliang He, Zhihua Wu, Xiangmiao Wu, Mingkui Tan, and Errui Ding. Ascnet: Self-supervised video representation learning with appearance-speed consistency. In *ICCV*, pp. 8096–8105, 2021. 3

Longlong Jing, Xiaodong Yang, Jingen Liu, and Yingli Tian. Self-supervised spatiotemporal feature learning via video rotation prediction. *arXiv preprint arXiv:1811.11387*, 2018. 3

Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, pp. 2556–2563, 2011. 6

Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *CVPR*, pp. 1–8, 2008. 5, 15

Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 15

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 15

Dezhao Luo, Chang Liu, Yu Zhou, Dongbao Yang, Can Ma, Qixiang Ye, and Weiping Wang. Video cloze procedure for self-supervised spatio-temporal learning. In *AAAI*, pp. 11701–11708, 2020. 1, 3

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, volume 32, pp. 8024–8035, 2019. 15

Viorica Patraucean, Ankur Handa, and Roberto Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR Workshop*, 2016. 3

Mandela Patrick, Dylan Campbell, Yuki M. Asano, Ishan Misra Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and Joo F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. In *NeurIPS*, pp. 12493–12506, 2021. 8

Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal contrastive video representation learning. In *CVPR*, pp. 6964–6974, 2021. 3, 9

Zhiwu Qing, Shiwei Zhang, Ziyuan Huang, Xiang Wang, Yuehuan Wang, Yiliang Lv, Changxin Gao, and Nong Sang. Mar: Masked autoencoders for efficient action recognition. *arXiv preprint arXiv:2207.11660*, 2022. 16

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, pp. 8821–8831, 2021. 3

Kanchana Ranasinghe, Muzammal Naseer, Salman Khan, Fahad Shahbaz Khan, and Michael Ryoo. Self-supervised video transformer. In *CVPR*, pp. 2874–2884, 2022. 3, 8, 9

Adria Recasens, Pauline Luc, Jean-Baptiste Alayrac, Luyu Wang, Florian Strub, Corentin Tallec, Mateusz Malinowski, Viorica Pătrăucean, Florent Altché, Michal Valko, et al. Broaden your views for self-supervised video learning. In *ICCV*, pp. 1255–1265, 2021. 3

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 6

Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, pp. 843–852, 2015. 3

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pp. 2818–2826, 2016. 15

Hao Tan, Jie Lei, Thomas Wolf, and Mohit Bansal. Vimpac: Video pre-training via masked token prediction and contrastive learning. *arXiv preprint arXiv:2106.11250*, 2021. 8, 9

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *arXiv preprint arXiv:2203.12602*, 2022. 1, 2, 3, 4, 5, 6, 8, 9, 16

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pp. 10347–10357, 2021. 1

Guangting Wang, Yizhou Zhou, Chong Luo, Wenxuan Xie, Wenjun Zeng, and Zhiwei Xiong. Unsupervised visual representation learning by tracking patches in video. In *CVPR*, pp. 2563–2572, 2021a. 3

Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *ICCV*, pp. 3551–3558, 2013. 14, 16

Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, 103:60–79, 2013. 5, 6, 16

Jinpeng Wang, Yuting Gao, Ke Li, Yiqi Lin, Andy J Ma, Hao Cheng, Pai Peng, Feiyue Huang, Rongrong Ji, and Xing Sun. Removing the background by adding the background: Towards background robust self-supervised video representation learning. In *CVPR*, pp. 11804–11813, 2021b. 3, 8

Jue Wang, Gedas Bertasius, Du Tran, and Lorenzo Torresani. Long-short temporal contrastive learning of video transformers. *arXiv preprint arXiv:2106.09212*, 2021c. 3, 8, 9

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41:2740–2755, 2018. 14

Rui Wang, Dongdong Chen, Zuxuan Wu, Yinpeng Chen, Xiyang Dai, Mengchen Liu, Yu-Gang Jiang, Luowei Zhou, and Lu Yuan. Bevt: Bert pretraining of video transformers. In *CVPR*, pp. 14733–14743. 1, 3, 8, 9

Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, pp. 2794–2802, 2015. 3

Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *CVPR*, pp. 14668–14678, 2022. 1, 2, 3, 5, 6, 7, 8, 9, 14, 15

Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P Allebach, and Chenliang Xu. Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution. In *CVPR*, pp. 3370–3379, 2020. 2

Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *CVPR*, pp. 10334–10343, 2019. 3

Shen Yan, Xuehan Xiong, Anurag Arnab, Zhichao Lu, Mi Zhang, Chen Sun, and Cordelia Schmid. Multiview transformers for video recognition. In *CVPR*, pp. 3333–3343, 2022. 1

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pp. 6023–6032, 2019. 15

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 15

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan L. Yuille, and Tao Kong. Image BERT pre-training with online tokenizer. In *ICLR*. OpenReview.net, 2022. 1

APPENDIX

A IMPLEMENTATION DETAILS

A.1 EXTRACTING OPTICAL FLOW

Removing camera motion. To alleviate the influence of camera motion, we follow previous work (Wang & Schmid, 2013) to warp the optical flow, which is also be applied in two stream video action recognition method (Wang et al., 2018). Specifically, by matching SURF (Bay et al., 2006) interest point in two frames to estimate the camera motion using RANSC (Fischler & Bolles, 1981) algorithm. Then the camera motion is removed by rectifying the frame, and the optical flow is re-calculated using the warped frame. This also relieves the model from being disturbed by the background and makes it pay more attention to the moving objects in the foreground.

Pre-extracting optical flow. We use the *denseflow*[1] tool box to pre-extract warp optical flow before training following Wang & Schmid (2013); Wang et al. (2018). We set the upper bound of flow value to 20. The stride of flow is set to $s_{flow} = 1$ if perform motion target interpolation, otherwise is equal to the sampling stride of the input rgb frames $s_{flow} = s_{rgb}$. The whole video dataset is split into chunks and processed on 4 nodes, each node is equipped with 2 Intel Xeon CPUs (32 cores per CPU) for video codec and 8 TITAN X GPUs for optical flow calculation speed-up. The extraction process spends about 1 day on the Kinetics400 dataset.

A.2 GENERATING TRAJECTORIES

Tracking the trajectories. With the calculated warp optical flow ω, we are able to track the position transition of each densely sampled grid point to produce a continuous trajectory, *i.e.*

$$\mathbf{p}_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * \omega_t)_{(x_t, y_t)}, \tag{6}$$

where ω_t is the optical flow at step t. M is the median filter kernel shapes 3×3 and $*$ denotes the convolution operator.

Pre-extracting trajectories. To reduce the io overhead of loading and decoding flow image during pre-training, we pre-process the dataset to generate dense trajectories of each video and pack them in the compressed pickle files. In pre-training, we use a special dataloader to sample K trajectories with length L starting from the first frame of each input 3D patch. During pre-training, we then crop and resize the trajectories in the spatial dimension to maintain the corresponding area with the input RGB frames.

Masking inaccurate loss. We notice that in most of the video samples, the objects move out of the range of the camera FOV. This circumstance often occurs on the objects that are at the edge of the video. The tracked trajectories in this case could be inaccurate, as visualized in Fig 5. We use an loss mask to remove these trajectories from the loss calculation:

$$\mathcal{L}^M = \frac{1}{N_p} \sum_{i=1}^{N_p} m_i \cdot \mathcal{L}_i \tag{7}$$

$$m_i = \begin{cases} 1, \mathbf{p}_t \in \mathbf{P} \\ 0, otherwise \end{cases} \tag{8}$$

where N_p is the total number of unmasked patches, $\mathbf{P} = \{(x_t, y_t) \mid 0{<}x_t{<}W, 0{<}y_t{<}H\}$ represents the points in the video clip.

A.3 EXTRACTING DIFFERENT PREDICTION TARGETS

HOG. In our experiment, we utilize the successful application of the HOG feature (Dalal & Triggs, 2005) in masked video modeling (Wei et al., 2022) to represent appearance information. HOG is

[1] https://github.com/yjxiong/dense_flow

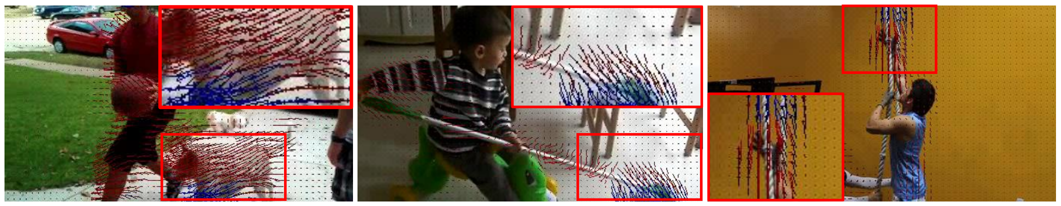


Figure 5: **Illustration of the inaccurate trajectories**, which are caused by objects moving out of the camera FOV.

the statistic histogram of image gradients in RGB channels. To speedup the calculation, we re-implement the HOG extractor by convolution operator with sobel kernel and the scatter method in the PyTorch (Paszke et al., 2019) Tensor API, and thus the calculation process can be conducted on the GPU. With the exception of the L2 normalization, which does not work in our practice, we precisely follow the recipe in Wei's work (Wei et al., 2022) to calculate a feature map of the entire image and then fetch the 9-bins HOG feature in each RGB channel.

HOF and MBH. These two features are spatial-temporal local features that have been widely used in the action recognition task since the 2010s. Similar to the HOG feature, these two features are the histogram of optical flow (Laptev et al., 2008), the only difference is that MBH is the second order histogram of the optical flow (Dalal et al., 2006). We implement them based on the HOG extractor that deals with optical flow.

Table 13: **Pre-training setting.**

config	SSV2	K400
optimizer	AdamW (Loshchilov & Hutter, 2019)	
base learning rate	1.5e-4	
weight decay	0.05	
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$ (Chen et al., 2020)	
batch size	512	
learning rate schedule	cosine decay (Loshchilov & Hutter, 2017)	
warmup epochs	40	
training epochs	400	
flip augmentation	*no*	*yes*
augmentation	MultiScaleCrop	

Table 14: **Fine-tuning setting.**

config	SSV2	K400	UCF101	HMDB51
optimizer	AdamW			
base learning rate	1e-3			
weight decay	0.05			
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$			
layer-wise lr decay	0.75 (Bao et al.)			
batch size	512			
learning rate schedule	cosine decay			
warmup epochs	5			
training epochs	50	100	100	100
flip augmentation	*no*	*yes*	*yes*	*yes*
RandAug	(9,0.5) (Cubuk et al., 2020)			
label smoothing (Szegedy et al., 2016)	0.1			
mixup (Zhang et al., 2018)	0.8			
cutmix (Yun et al., 2019)	1.0			
drop path	0.1			

Table 15: **Training speed-up versus performance decrease**. We report the total time cost of entire pre-training and fine-tuning procedure.

Data ratio	Input patches	Acc@1	Speed-up
100%	100%	64.8	1.0×
25%	50%	61.1	2.5×

Table 16: Without the additional short-term motion targets features draws competitive result.

Additional targets	Acc@1
Traj. + MBH + HOF	63.5
Traj. (ours)	63.5

A.4 Pre-training and Fine-tuning Settings

Something-Something V2.[2] The default settings for pre-training and fine-tuning on Something-Something V2 (SSV2) are shown in Tab 13 and Tab 14. We take the same recipe as in Tong et al. (2022).

Kinetics-400.[3] The default settings for pre-training and fine-tuning on Kinetics-400 (K400) are shown in Tab 13 and Tab 14. We take the same recipe as in Tong et al. (2022).

UCF101.[4] We pre-train the model on the Kinetics-400 for 400 epochs and then transfer to the UCF101. The default settings of fine-tuning are shown in Tab 14.

HMDB51.[5] The default setting is the same as in UCF101.

B Additional Results

B.1 Training speed-up versus performance decrease

To speed up the training procedure during the ablation study, we (1) randomly select 25% videos from the training set of Something-Something V2 dataset to pre-train the model, (2) randomly drop 50% patches following Qing et al. (2022), specifically, we adopt a random masking strategy to select 50% input patches to reduce computation of self-attention. We provide a comparison across the speed-up and performance decrease due to these techniques in Tab 15. Notice that with an acceptable performance drop (-3.7%), we speed up the ablation experiment by 2.5 times.

B.2 Comparing with the original version of dense trajectory.

The original version of dense trajectories (Wang et al., 2013; Wang & Schmid, 2013) combines short-term motion targets, including HOF and MBH features to represent motion. However, we found that only using the position changes to represent objects' movement is enough, see Tab 16.

C More Visualization results

We provide more visualization results of our trajectory motion targets. Videos are sampled from the Something-Something V2 validation set. For each 3D cube patch shape $2 \times 16 \times 16$, we visualize the first frame and the trajectories start from it. All the trajectories are with length $L = 6$ as our default setting.

[2] https://developer.qualcomm.com/software/ai-datasets/something-something
[3] https://www.deepmind.com/open-source/kinetics
[4] https://www.crcv.ucf.edu/data/UCF101.php
[5] https://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/

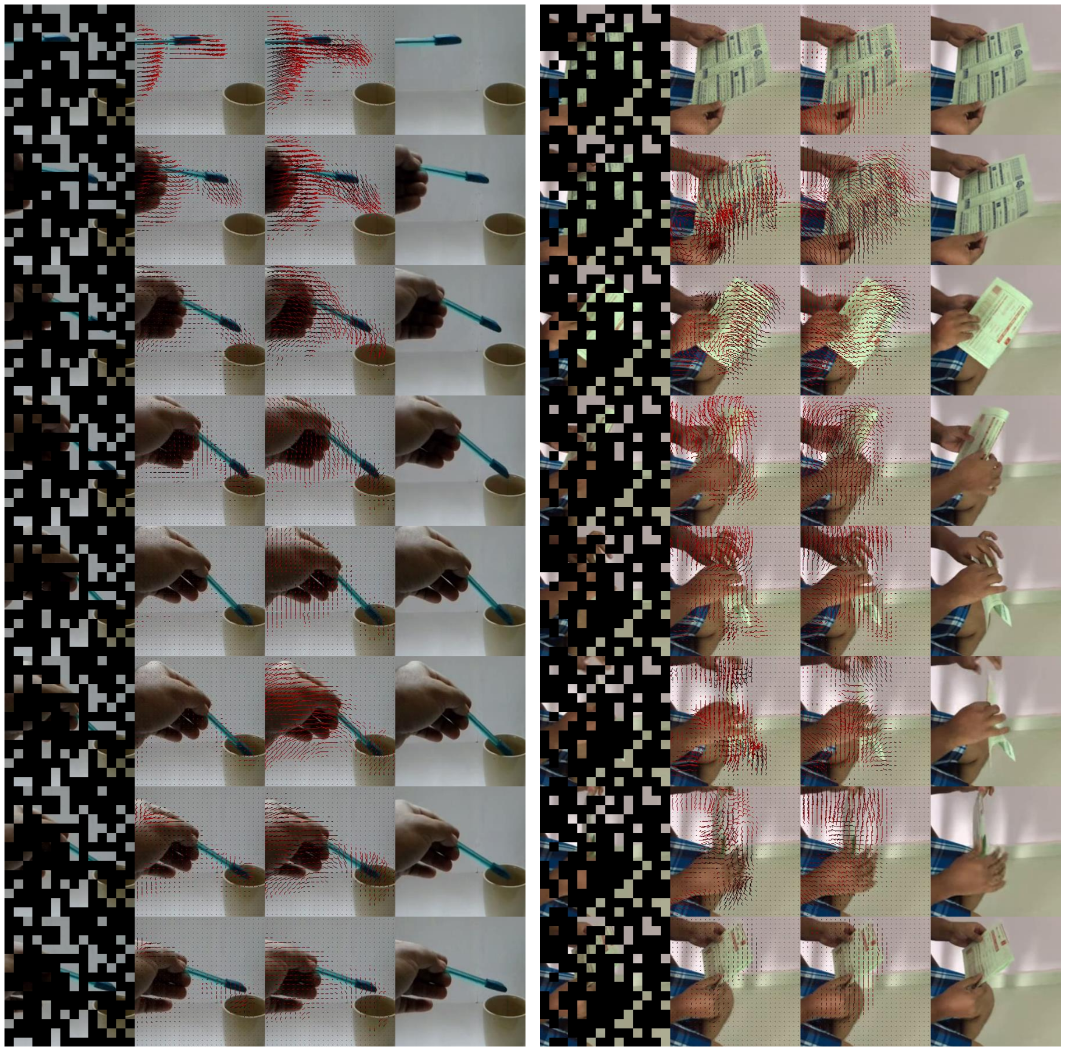


Figure 6: **Visualization of trajectory motion targets.** Each column represents: (a) Masked frames; (b) ground-truth; (c) prediction; (d) original frame. Trajectories are colored dark to light in chronological order.

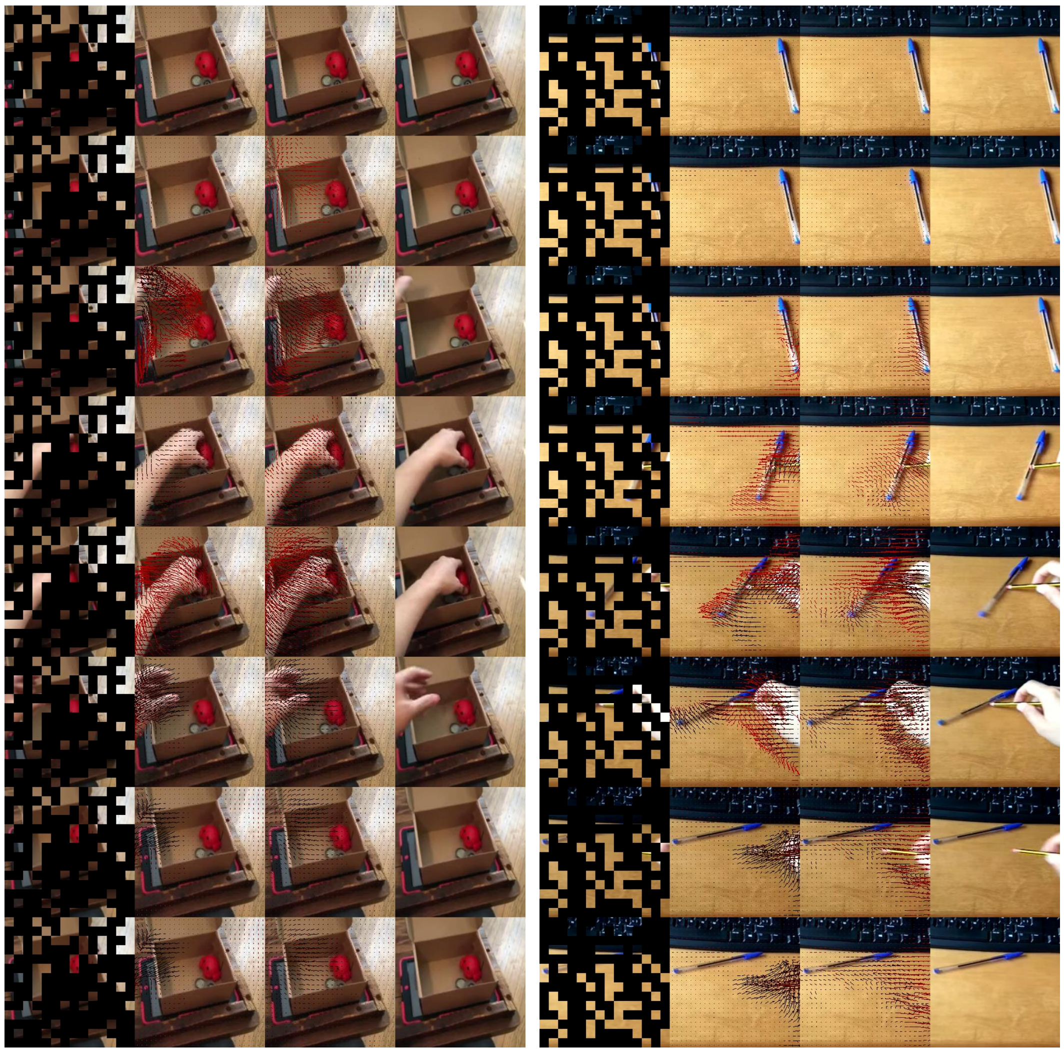


Figure 7: **Visualization of trajectory motion targets.** Each column represents: (a) Masked frames; (b) ground-truth; (c) prediction; (d) original frame. Trajectories are colored dark to light in chronological order.