TrackMAE: Video Representation Learning via Track Mask and Predict

Renaud Vandeghen* [1,2] Fida Mohammad Thoker* [2] Bernard Ghanem[2] Marc Van Droogenbroeck[1]
[1]University of Liège [2]KAUST
∗ Equal contribution

Abstract

Masked video modeling (MVM) has emerged as a simple and scalable self-supervised pretraining paradigm, but only models motion information implicitly, limiting the encoding of temporal dynamics in the learned representations. As a result, such models struggle on motion-centric tasks that require fine-grained motion awareness. To address this, we propose TrackMAE, a simple masked video modeling paradigm that explicitly uses motion information as a reconstruction signal. In TrackMAE, we use an off-the-shelf point tracker to sparsely track points in the input videos generating motion trajectories. Furthermore, we exploit the extracted trajectories to improve the random tube masking with a motion-aware masking strategy. We enhance video representations learned in both pixel and feature semantic reconstruction space by providing a complimentary supervision signal in the form of motion targets. We evaluate on six datasets across diverse downstream settings and find that TrackMAE consistently outperforms the state-of-the-art video SSL baselines, therefore learning more discriminative and generalizable representations.

1. Introduction

Self-supervised learning (SSL) has become the default pretraining recipe, allowing models to learn diverse yet powerful representations for different modalities, including text [5, 12] image [3, 6–9, 21, 22, 36, 58] and video [23, 49, 51] replacing manual labels with pretext objectives that exploit the structure of the raw data. Among video methods, masked video modeling (MVM) stands out for its simplicity and scalability: a high fraction (often 80–95%) of spatiotemporal tokens is hidden, and a vision transformer is trained to reconstruct them from the visible context [14, 23, 45, 49]. In practice, videos are decomposed into tubelets (*e.g.*, 2×16×16) and only visible tokens are encoded, which makes computation efficient and enables training on large corpora. However, the most common instantiation of MVM is pixel reconstruction with random tube masking, which tends to emphasize low-level appearance statistics (color/texture continuity, local smoothness) and under-utilize temporal structure. Real-world videos also exhibit strong temporal redundancy and sparse foreground motion, so masked pixel reconstruction can often be solved via spatial correlations or short-horizon consistency without learning fine-grained dynamics. This mismatch is observed in the downstream performance of such models, as shown in [46, 47]. They perform well on appearance-dominated datasets like Kinetics-400 [27] or UCF101 [44] but lag on motion-centric tasks such as Something-Something V2 [19] and FineGym[42], where accurate temporal modeling is crucial.

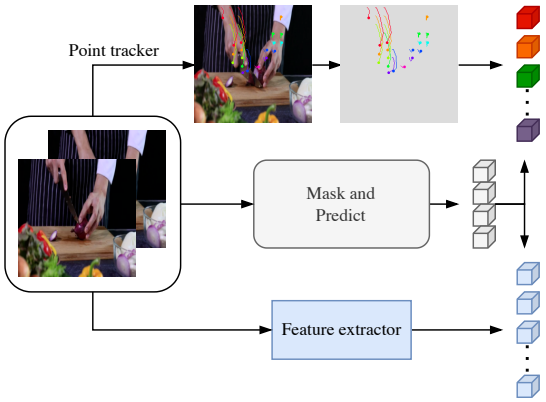

Figure 1. **TrackMAE** improves masked video modeling by jointly predicting spatial features and motion trajectories in a mask-and-predict fashion.

Recent efforts inject more structure into MVM along two orthogonal axes. One line alters *where* to learn by biasing masks toward informative regions, *e.g.*, selecting high-motion tokens via flow [23] or motion vectors [14], or adaptively sampling tokens with learned heuristics [2]. Another line changes *what* to learn by replacing pixels with feature-space targets (*e.g.*, HOG, DINO, CLIP), which mitigates pixel-level shortcuts and encourages learning of high-level semantics like object/part structure, attributes, and ob-

ject–scene relations [45, 48]. While both directions help, they supervise motion only *implicitly*. The model is never asked to predict *how* things move or to maintain identity over time. As a result, improvements in motion sensitivity are limited, especially under high masking ratios where temporal cues must be inferred from few visible tokens. We suggest that temporal correspondence should be a first-hand signal during pretraining, complementing pixel/feature targets rather than competing with them.

In this work, we propose to directly use motion as a training signal by complementing the spatial reconstruction with a motion prediction target. By leveraging the motion prediction produced by a point tracker, our method **TrackMAE** (Fig. 1) aims to jointly reconstruct both the spatial and motion targets. In particular, we use CoTracker3 [26], a robust and high-quality point tracker, to extract motion trajectories. TrackMAE adds two components to masked video modeling: (i) a lightweight *trajectory decoder* that predicts point-track displacements, and (ii) *motion-aware masking* that preferentially samples visible tokens from both high- and low-motion regions. Finally, we show that our motion target complements both pixel- and feature-based spatial targets, leading to on par with or state-of-the-art results on several benchmarks. We summarize our contributions as follows:

1. We propose TrackMAE, a new masked video pretraining scheme with explicit motion awareness based on prediction of tracked point trajectories.
2. We improve the random tube masking strategy with a motion-aware masking that equally samples visible tokens from high- and low-motion regions.
3. Extensive evaluation on multiple datasets demonstrating state-of-the-art performance on motion-centric benchmarks as well as strong generalization capabilities.

2. Related work

Masked video modeling. Masked modeling has emerged as a powerful learning paradigm for representation learning for both visual and non-visual modalities like text [5, 12], audio [10, 18, 24], images [1, 3, 13, 20, 22, 55, 58], and video [14, 16, 23, 45, 49, 50]. The goal is to hide a portion of the input from the model and aim to predict the hidden part from the visible context only. In vanilla masked video modeling [49], a considerable portion of the input pixels are typically kept hidden at random, and the model is trained to predict those hidden pixels from the visible portion, thereby encoding useful video representations. Many follow-up works improve this mask-and-predict task by either leveraging *what to mask* [2, 14, 23, 38] or *what to predict* [31, 40, 45, 48, 54, 56].

In the masking paradigm, MAR [38] designed a new masking strategy tailored for action recognition, based on the running cell masking. MGM [14] leverages the motion information contained in motion vectors of the raw videos to create a motion-aware masking strategy. MGMAE [23] follows the same idea of improving the masking based on motion information extracted from an optical flow instead. AdaMAE [2] learns a sampling network that selects regions with high information, which continually improves the masking strategy over training time.

Beyond the pixel reconstruction paradigm, MaskFeat [54] aims to reconstruct HOG features [11]. In the same manner, MME [45] reconstructs both position changes and the HOG-based shape features. MME builds motion targets based on pre-computed optical flow forming trajectory-like signals, and extracts hand-crafted HOG features around these trajectories for prediction. Such a pipeline requires a heavy pre-processing, camera-motion–sensitive pipeline and produces noisier long-range motion, while we directly extract the trajectories from RGB on the fly. Furthermore, MotionMAE [56] aims to reconstruct temporal frame differences, while recent works like SIGMA [40] rely on clustered DINO [8] features as the reconstruction target. Closely related, SMILE [48] uses CLIP [39] features for reconstruction and improves the motion awareness by injecting synthetic motions by copy-pasting segmented objects onto the videos using randomly generated paths. Different from SMILE, our motion information comes from real trajectories from a tracking module representing actual pixel motion.

Overall, our approach extracts motion trajectories from a tracking module and leverages them as additional motion reconstruction target and reuses them to enhance the motion awareness in the masking, thereby improving the spatial and motion semantics of the learned video representations.

Learning from motion. Motion in video data has long served as free supervision in vision, and prior work exploits it for several tasks. For image representation learning, Wang and Gupta [52] use visual tracking to create positive pairs, encouraging patches that are linked by a track to have similar features. For dense visual correspondence, Wang et al. [53] learn features by tracking backward and then forward in time (cycle consistency), enabling self-supervised correspondence. Li et al. [32] learn dense correspondence by optimizing region tracking and pixel-level matching.

With the emergence of reliable point trackers such as CoTracker3 [26], point tracks have become strong supervisory signals. They can guide attention routing [29] or supervise time-consistent dense features via clustering [41]. Tracktention [29] injects point-track correspondences into the attention layer of image models, yielding temporally consistent features that handle large motion and turning them into strong video models for depth estimation and colorization. Similarly, MoSIC [41] first clusters long-range tracked trajectories via optimal-transport clustering and then propa-

gates the cluster assignments along tracks to enforce temporal coherence under occlusion and viewpoint change, improving dense representations.

Our approach is inspired by these works to use trajectories for motion injection, albeit with a different objective. Instead of temporal consistency or label propagation, we aim to improve the motion semantics in masked video modeling representations.

3. Methodology

In this section, we start by revisiting the masked video modeling frameworks in Sec. 3.1. In Sec. 3.2, we develop our new training scheme, based on motion prediction.

3.1. Masked Video Modeling

Input. Masked video modeling, *e.g*. VideoMAE [49], is an extension of standard image masked modeling MAE [22]. The input is a short clip $\mathbf{V} \in \mathbb{R}^{T\times H\times W\times 3}$, which is partitioned into non-overlapping space-time tubelets $\mathcal{C} = \{\mathbf{c}_i\}_{i=1}^{N}$ of size $t\times p\times p$, where t is the temporal size of the tubelet and p is the spatial patch size (*e.g*., $t=2$, $p=16$). This yields $N = \frac{T}{t}\cdot\frac{H}{p}\cdot\frac{W}{p}$ tokens of size 3, each corresponding to a local cubic video volume. A tubelet embedding layer (a 3D conv) maps each tubelet in $\mathcal{C}$ into a set of tokens $\mathcal{T} = \{\tau_i\}_{i=1}^{N}$, $\tau\in\mathbb{R}^D$, and a fixed positional embedding is added to preserve spatial and temporal order.

Masking. Masked video modeling randomly hides a large subset of tokens and asks the model to recover them from the remaining context. A high-ratio masking (typically 90%) is applied at the token level following a Bernoulli distribution. This produces a visible set $\mathcal{T}^{visible}$ and a masked set of tokens $\mathcal{T}^{masked}$. Such aggressive masking makes reconstruction non-trivial, forcing the model to capture meaningful spatio-temporal dependencies in the video data.

Architecture. Masked video modeling relies on a standard ViT-based encoder–decoder architecture. The encoder Φ takes the visible set of tokens as the input to produce latent representations $\mathbf{Z} = \Phi(\mathcal{T}^{visible})$. The decoder Ψ then maps the encoder's output to produce reconstruction predictions. To reconstruct the input clip, a complete sequence is formed by inserting learnable `[MASK]` tokens at the masked positions and adding the same positional embeddings used at input. The decoder Ψ takes this complete sequence and predicts the missing content at every position. The goal of the decoder is to predict the space-time tubelets $\hat{\mathcal{C}} = \Psi([\mathbf{Z}, [\texttt{MASK}]])$, of the same size as $\mathcal{C}$.

Reconstruction objectives. Since the main goal is to employ a mask-and-predict task, the reconstruction is purely performed on masked tokens to prevent any information leakage or shortcut solutions by also predicting the visible tokens. In most masked video modeling methods [14, 23, 49], the reconstruction is done in the pixel space. That is, the model is directly optimized to predict the pixel values of masked space-time tubelets from the set of visible tubelets with the following L2 loss

$$\mathcal{L}_{pixel} = \frac{1}{|\mathcal{T}^{masked}|}\sum_{i\in\mathcal{T}^{masked}} \|\mathbf{c}_i - \hat{\mathbf{c}}_i\|_2^2, \qquad (1)$$

where $\hat{\mathbf{c}}_i$ is the i_{th} token in the decoder. To solve this reconstruction task under a strong information dropout, the model has to encode the spatio-temporal dynamics of the input videos, thereby learning useful video representations.

Beyond the pixel space reconstruction, several works reconstruct in more semantic feature spaces *e.g*., HOG [11], DINO [8, 36], or CLIP [39]. Concretely, video frames are processed to extract per-frame descriptors, which are then aligned to the model's space–time tokens. In other words, each space-time pixel tubelet $\mathbf{c}_i$ is projected onto a feature space $\mathcal{F}$, extracting feature tokens $\mathbf{f}_i$ for each tubelet. The model is then optimized to predict the semantic feature values of masked space-time tubelets from the set of visible pixel tubelets, according to the following loss

$$\mathcal{L}_{feature} = \frac{1}{|\mathcal{T}^{masked}|}\sum_{i\in\mathcal{T}^{masked}} \left\|\mathbf{f}_i - \hat{\mathbf{f}}_i\right\|_2^2, \qquad (2)$$

where $\hat{\mathbf{f}}_i$ is again the i_{th} token in the decoder output $\hat{\mathcal{F}} = \Psi(\Phi(\mathcal{T}^{visible}), [\texttt{MASK}])$. Such reconstruction adds an abstraction to the mask-and-predict task, reducing the chances of any shortcuts in pixel reconstruction, and encourages directly modeling high-level video semantics instead of low-level semantics in the pixel space.

3.2. TrackMAE: Learning from Motion

In this section, we introduce TrackMAE, a masked video pretraining framework that leverages tracked trajectories from CoTracker3 [26] in the form of motion prediction and motion-aware masking to enhance temporal awareness in learned video representations. In particular, we make two additions to the vanilla masked modeling paradigm. First, we integrate motion prediction, *i.e*. predicting a sparse set of trajectories as an additional self-supervision task along with the spatial reconstruction. Second, we replace the random uniform masking with a motion-aware masking that keeps visible tokens from high- and low-motion regions using displacement magnitudes from the extracted trajectories.

Extracting motion targets. As mentioned in Sec. 3.1, the input clip $\mathbf{V}\in\mathbb{R}^{T\times H\times W\times 3}$ is patchified into space-time tubelets $\mathcal{C} = \{\mathbf{c}_i\}_{i=1}^{N}$ of size $t\times p\times p$, where p is the spatial patch size. Our goal is to approximate the motion in the given input clip $\mathbf{V}$ by tracking the center pixels of patches in the first frame in the subsequent frames with an off-the-shelf tracking module like CoTracker3 [26]. To that end, we sample query points from a uniform grid of size $G\times G$ for the first frame, where $G = \frac{H}{p}$, and predict their

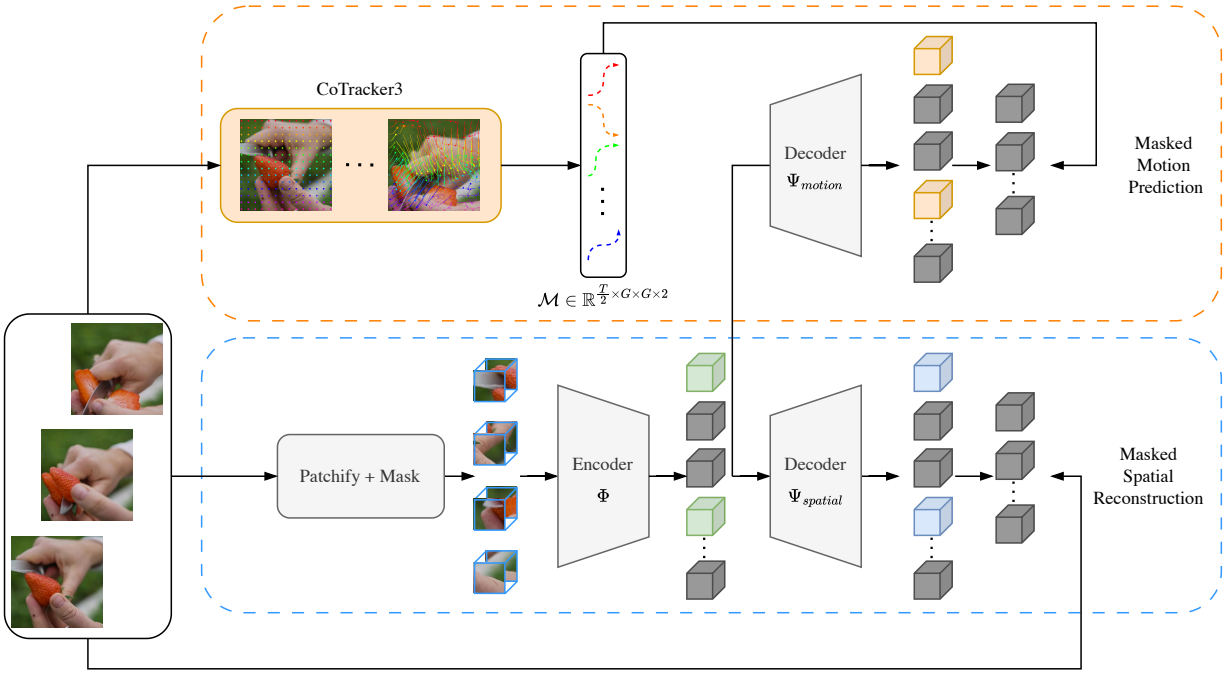

Figure 2. **Overview of TrackMAE.** In the lower branch, a video clip $\mathbf{V}$ is first patchified and masked. The visible tokens are fed to a ViT encoder Φ. Then the decoder $\Psi_{spatial}$ aims to reconstruct spatial features based on the encoder output. In the upper branch, the input video clip is processed by a CoTracker3 module, extracting sparse point trajectories. The encoder output is then passed to a second decoder Ψ_{Motion}, which aims to predict the extracted trajectories. The training objective combines both motion and spatial reconstruction.

2D positions (x, y) in subsequent frames, extracting a set of motion tracks of shape $T \times \frac{H}{p} \times \frac{W}{p} \times 2$. For efficiency and to match the shape of input video tokens, we feed the tracking module every other frame, producing motion tokens $\mathcal{M} = \{\mathbf{m}_i\}_{i=1}^{N}$ of size 2, matching the size of $\mathcal{C}$. Finally, this set of motion tokens $\mathcal{M}$ is used as the reconstruction target. In practice, we predict the displacement of the point tracks instead of absolute trajectory values.

By sparsely tracking only one point per 16 by 16 patch, we may not accurately capture fine motion displacement in the extracted motion tokens. Ideally, we would like to track as many points per patch as possible to generate denser motion targets. However, dense tracking is expensive, with computational cost proportional to the query grid size G. To overcome this issue, we introduce a simple yet effective upsampling trick. Assuming that nearby pixels in a patch behave similarly in terms of motion, we can spatially interpolate extracted sparse motion tokens to simulate denser trajectories per patch. In other words, we can spatially upsample the sparse motion tokens $\mathcal{M}$ into a dense set of motion tokens $\mathcal{U}$ of size $\frac{T}{2} \cdot \frac{\upsilon H}{p} \cdot \frac{\upsilon W}{p}$, where υ is the upsampling factor. This is equivalent to tracking υ^2 points per patch, generating more dense motion targets for reconstruction. We show, in Sec. 4.3, that upsampled targets demonstrate better downstream performance without any added cost.

Architecture. In TrackMAE, our goal is to jointly solve two prediction tasks in a mask-and-predict manner; one is spatial reconstruction, and the other is motion prediction. As in Sec. 3.1, we employ an encoder-decoder framework with a common encoder and two separate decoders. As before, encoder Φ takes visible tokens $\mathcal{T}^{visible}$ as the input to produce latent features $\mathbf{Z}$. The two decoders $\Psi_{spatial}$ and Ψ_{motion} take the encoded features $\mathbf{Z}$ with learnable `[MASK]` tokens and positional embeddings to predict the spatial and motion tokens, respectively, as $\hat{\mathcal{C}} = \Psi_{spatial}([\mathbf{Z}, [\texttt{MASK}]])$ and $\hat{\mathcal{M}} = \Psi_{motion}([\mathbf{Z}, [\texttt{MASK}]])$.

Objectives. In addition to the spatial reconstruction objective, we additionally optimize for the motion prediction. We follow the masked reconstruction strategy to only predict motion tokens of the hidden portion on the input as

$$\mathcal{L}_{motion} = \frac{1}{|\mathcal{T}^{masked}|} \sum_{i \in \mathcal{T}^{masked}} \|\mathbf{m}_i - \hat{\mathbf{m}}_i\|_2^2, \quad (3)$$

which supervises the motion decoder on masked positions only. The final objective is a weighted sum of the two objectives as

$$\mathcal{L} = \mathcal{L}_{spatial} + \lambda * \mathcal{L}_{motion}, \quad (4)$$

where $\mathcal{L}_{spatial}$ is either $\mathcal{L}_{pixel}$ (Eq. (1)) or $\mathcal{L}_{feature}$ (Eq. (2)). As shown in the experiments, our proposed

motion prediction loss $\mathcal{L}_{motion}$ is complementary to both $\mathcal{L}_{pixel}$ and feature reconstruction $\mathcal{L}_{feature}$. The overview of the method is shown in Fig. 2.

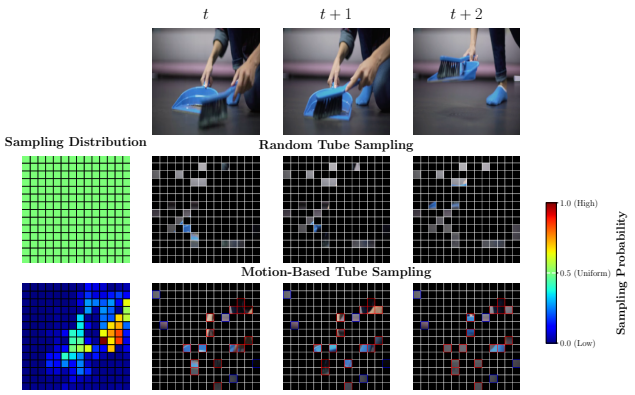

Figure 3. **Masking comparison.** We show how our motion-based tube masking compares to random tube masking. By explicitly sampling visible tokens, our motion-based sampling distribution ensures that visible tokens cover both motion and static regions. In the motion-based tube sampling, **red** squares are sampled from the high-motion and **blue** squares from the low-motion bin.

Motion aware masking. The vanilla tube masking used in VideoMAE does not assume any information of motion, leading to random masking maps. Since the tracker module already gives us motion prediction for trajectory reconstruction, we can also leverage that information to create a motion-guided masking map. In particular, based on the full trajectory predictions, we compute the average displacement $\bar{\mathbf{M}}$, of size $p \times p$, for each query point over the temporal dimension. We then use $\bar{\mathbf{M}}$ as a sampling distribution for the visible tokens. Such distribution can be seen in the first column of the last row in Fig. 3. To sample visible tokens from that distribution, we first create 2 uniform bins, containing high- and low-motion samples. We uniformly sample visible tokens from each bin, using a motion ratio ρ_{motion} to control the number drawn per bin. This masking formulation allows to control where the masking should be done and in what proportion. Such a masking example is depicted in Fig. 3, as well as the random tube masking, which can be expressed as a special case of our sampling distribution, using a uniform distribution instead of $\bar{\mathbf{M}}$.

4. Experiments

In Sec. 4.1, we describe the pretraining setup that we use in our experiments. In Sec. 4.2, we compare our method against prior methods in different linear probing and full finetuning setups. We run some ablation studies in Sec. 4.3 and assess the generalization performance in Sec. 4.4.

4.1. Implementation Details

Following previous methods, we pretrain a video-based ViT-B model on the Kinetics-400 (K400) [27]. We replace the original tube masking with our motion-guided tube masking, using a motion ratio of $\rho_{motion} = 50\%$, and equally balance the losses with λ=1. For pixel reconstruction, we use the default offline CoTracker3 module with a grid size of 14×14, and use upsampling with υ=2. With feature reconstruction, we use a CLIP ViT-B model to extract features. In that setup, we use a grid size of 28×28, without upsampling and without our motion-guided masking. Unless stated otherwise, we follow the same hyperparameters as in [49] and pretrain our model for 800 epochs. More details can be found in the supplementary materials.

Table 1. **Linear probing comparison.** We evaluate TrackMAE on different spatial- and motion-centric benchmarks. For both pixel and feature reconstruction, our method is on par with previous methods for spatial-centric tasks. For motion-centric tasks, our method largely improves previous methods. We report the Top-1% accuracies of ViT-B models pretrained on K400, and highlight in **bold** best and underline second best results.

Method	Target	Spatial-centric		Motion-centric	
		K400	HMDB	SSv2	GYM
Pixel Reconstruction					
VideoMAE [49]	Pixel	20.7	37.7	17.5	23.9
MVD [51]	Pixel	18.7	28.6	12.2	22.7
MGMAE [23]	Pixel	24.9	41.3	16.8	26.1
EVEREST [25]	Pixel	14.1	30.3	14.5	23.3
MGM [14]	Pixel	19.8	40.3	21.7	25.8
TrackMAE (ours)	Pixel	25.7	40.6	23.6	29.0
Feature Reconstruction					
MME [45]	HOG	19.1	37.1	16.6	29.0
SIGMA [40]	DINO	47.5	52.3	20.8	30.1
SMILE [48]	CLIP	**56.2**	**53.4**	23.7	30.2
TrackMAE (ours)	CLIP	55.2	53.1	**27.3**	**31.8**

4.2. Comparison with State-of-the-Art

Linear Probing. To evaluate the quality of learned video representations, we conduct linear probing experiments across four standard action recognition benchmarks. In this setup, the pretrained encoder is frozen, and a linear classifier is trained on top of it. This isolates the effect of pretraining and removes the influence of fine-tuning dynamics, providing a proper measure of representation quality. We compare TrackMAE with prior methods based on both pixel and feature reconstruction. For a fair comparison with prior methods, we evaluate using their publicly released ViT-B checkpoints pretrained on K400. All downstream results are obtained under a unified evaluation protocol.

Table 1 presents the linear probing results across multiple benchmarks. In the pixel reconstruction setting, Track-

Table 2. **Full finetuning comparison.** We evaluate TrackMAE finetuned on SSv2 and K400 after pretraining the models on K400, outperforming all prior works. We highlight in **bold** best and underline second best results.

Method	Targets	SSv2 Top-1	K400 Top-1
VideoMAE [49]	Pixel	68.5	80.0
OmniMAE [17]	Pixel	69.0	80.8
MGM [14]	Pixel	71.1	80.8
MGMAE [23]	Pixel	68.9	81.2
TrackMAE (ours)	Pixel	70.1	80.8
MME [45]	HOG	70.5	81.5
SIGMA [40]	DINO	71.1	81.5
SMILE [48]	CLIP	72.1	83.1
TrackMAE (ours)	CLIP	**72.8**	**83.6**

MAE consistently outperforms the VideoMAE baseline by a margin of ∼5% across all datasets. This affirms our hypothesis that predicting motion provides a strong self-supervisory signal for learning discriminative video representations. Moreover, TrackMAE surpasses other motion-aware pixel reconstruction methods such as MGM [14] and MGMAE [23], particularly on motion-centric datasets like SSv2 and FineGym. This performance gap underscores TrackMAE 's superior ability to encode fine-grained temporal dynamics. In the feature reconstruction setting, TrackMAE again yields consistent gains and outperforms all prior methods, including the state-of-the-art SMILE for the motion-centric tasks. These results demonstrate that motion prediction is not only effective in isolation but also complements semantically rich targets (*e.g.*, CLIP features), enhancing representation learning beyond the pixel space. In summary, these findings confirm that our approach leads to more temporal awareness in the learned representations.

Full Finetuning. To fully leverage the learned spatiotemporal representations, we perform end-to-end finetuning of both the pretrained backbone and the classification head on downstream datasets. This setup is crucial for assessing the transferability and task-specific adaptability of representations learned through our trajectory-guided pretraining. In contrast to linear probing, which freezes the encoder, full finetuning enables the model to refine its temporal and semantic understanding based on the target task distribution. We evaluate our method on two representative benchmarks: Kinetics-400 and Something-Something V2 (SSv2), covering both appearance- and motion-focused benchmarks. We use the same evaluation protocol as the prior masked video modeling works [14, 23, 49], ensuring a fair comparison. Further details, are provided in the supplementary material.

We evaluate TrackMAE under two standard transfer regimes: in-domain transfer (pretraining and finetuning on the same dataset *i.e.*, K400) and cross-domain transfer (pretraining on K400 and finetuning on SSv2). The results, summarized in Tab. 2, show that TrackMAE consistently achieves strong performance across both settings. As with linear probing, we again observe that our TrackMAE with pixel reconstruction improves the VideoMAE baseline by +0.8% for in-domain transfer, and +1.6% for cross-domain transfer. Moreover, TrackMAE with pixel targets is on par or outperforms other prior methods with pixel targets, OmniMAE (1.1% on SSv2) MGMAE (1.2% on SSv2), even in full-finetuning settings. This again validates that adding our motion prediction task to pixel reconstruction is effective for learning better transferable video representations.

Finally, TrackMAE with CLIP targets outperforms all other prior methods, achieving state-of-the-art performance in all settings. In particular, TrackMAE outperforms SMILE, which also uses CLIP targets along with synthetic motion infusion for learning motion-aware representations by 0.7% and 0.5% on SSv2 and K400, demonstrating better motion encoding without the need for any synthetic motion priors. To summarize, adding motion prediction targets complements the spatial reconstruction targets to enrich the learned video representations for both appearance and motion-focused downstream tasks even in full-finetuning settings. We show a more detailed SOTA comparison with other pretraining settings in the supplementary material.

Table 3. **Reconstruction targets.** Trajectory reconstruction provides a strong supervisory signal to learn useful video representations and complements both pixels and CLIP reconstruction, consistently improving the downstream results.

Reconstruction Targets	$K400_s$	$SSv2_s$
Trajectory only	46.5	53.1
Pixels	46.0	52.2
Pixels + Trajectory	**48.9**	**55.7**
CLIP	52.7	57.1
CLIP + Trajectory	**55.8**	**61.1**

4.3. Ablations

To assess the contribution of each design component in our TrackMAE framework, we conduct a series of ablation experiments, as summarized in Tabs. 3 to 6. For computational efficiency, we adopt the ViT-S architecture and pretrain on a subset of Kinetics-400, referred to as $K400_s$, which includes $1/3$ of the total training videos. Evaluation is performed on both $K400_s$ and $SSv2_s$, also a reduced version of SSv2. By default we use pixel and motion trajectories as reconstruction targets, employ a grid size of 14×14, use random tube masking, set λ=1, and train for 200 epochs, unless specified otherwise.

Impact of reconstruction targets. Table 3 analyzes the impact of the different target reconstructions used. We first observe that trajectory reconstruction as a standalone task

Table 4. **Masking strategy.** Replacing random tube masking with our motion-based masking consistently improves the results for both pixels and pixels + trajectory reconstruction.

Target	Masking	$K400_s$	$SSv2_s$
Pixel	Tube	46.0	52.2
Pixel	Motion-aware	**46.6**	**52.6**
Pixel + Trajectory	Tube	48.9	55.7
Pixel + Trajectory	Motion-aware	**49.4**	**56.2**

Table 5. **Impact of dense tracking.** Increasing the tracker grid size consistently improves the results, but at a higher computational cost. Upsampling the tracker's prediction from 14×14 to 28×28 yields better performance without any added cost.

Grid Size	Upsampling	$K400_s$	$SSv2_s$
14×14	None	48.9	55.7
28×28	None	49.5	56.7
56×56	None	**50.0**	**57.0**
14×14	$14 \rightarrow 28$ (v=2)	**50.6**	**57.6**
14×14	$14 \rightarrow 56$ (v=4)	50.4	57.4

is very effective for learning useful video representations, indicating that it can act as a strong self-supervisory signal. Next, we observe that combining trajectory reconstruction with pixel reconstruction outperforms both trajectory-only and pixel-only reconstruction by ($+2.4\%$ and $+2.9\%$) on $K400_s$ and ($+2.6\%$ and $+3.5\%$) on $SSv2_s$, respectively. Finally, when combined with more high-level semantic targets like CLIP instead of pixel targets, we observe a significant improvement of ($+2.9\%$ and $+4.0\%$). One of the reasons for this is that trajectory targets are closer to pixel targets in the sense that they represent movement of pixels and might rely on the same semantics for solving the reconstruction tasks. On the other hand, CLIP targets are highly semantic and largely encode what is present but not how things move, and trajectory prediction fills that gap with a more complimentary reconstruction task. In summary, this clearly indicates that exploiting motion trajectories as a complementary target strengthens video representation learning.

Impact of motion-aware masking. Table 4 shows the impact of our proposed motion-aware masking over random tube masking. We compare our motion-aware masking against random tube masking under two regimes: pixel-only and pixel+trajectory reconstruction. Motion-aware masking yields consistent gains of roughly +0.5% at no extra computation, as the same trajectories used for supervision are repurposed to construct the masking prior.

Impact of dense tracking. As mentioned in Sec. 3.2, we track one point per patch from the first frame and initialize query points accordingly on a coarse grid of size 14×14 to extract sparse motion trajectories. In this ablation, we show the impact of extracting and predicting denser trajectories.

Table 6. **Motion ratio masking.** Equally sampling (50%) visible tokens from the high motion and low motion bins yields the best results compared to asymmetric sampling.

ρ_{motion}	$K400_m$	$SSv2_m$
25	48.7	55.4
50	**49.4**	**56.2**
75	49.0	55.9

Table 7. **Balancing the losses.** Equally balancing the loss ($\lambda = 1.0$) yields the best results compared to unbalanced setups.

λ	$K400_s$	$SSv2_s$
0.1	47.1	54.0
0.5	48.2	54.6
1.0	**48.9**	**55.7**
2.0	48.8	55.7

In particular, we initialize query points on denser grids (28×28 and 56×56) to extract denser trajectories. The results in Tab. 5 show that moving from a coarse to denser prediction consistently improves the results.

However, these gains come at a cost, since the computational cost of CoTracker3 is directly proportional to the query grid size. Table 5 also shows that by upsampling the sparse trajectories to denser trajectories via spatial interpolation from $14 \rightarrow 28$ (v=2) yields the strongest gains on both datasets at no cost ($+1.7\%$ and $+2.0\%$). However, upsampling from $14 \rightarrow 56$ (v=4) does not improve over $14 \rightarrow 28$ (v=2) setting. One of the reasons could be that spatial interpolation exploits the piecewise-smooth nature of local motion and densifies supervision at token locations, strengthening gradients without introducing tracker noise.

Impact of ρ_{mask}. Table 6 indicates how our motion-guided masking behaves when we vary the proportion of masked tokens sampled from high- and low-motion regions. Sampling too few (25%) or too many (75%) motion locations slightly degrades performance compared to 50% motion ratio, showing that equally biasing the mask towards both motion and static parts is a good balance.

Impact of λ. Table 7 shows that using the same weight for the losses yields the best results, indicating that both signals are useful during training.

4.4. Downstream Generalization on SEVERE

Next, following [40, 48], we assess the robustness of TrackMAE beyond standard action recognition on the SEVERE benchmark proposed in [46] and extended in [47]. It consists of a controlled suite of eight evaluations designed to probe four aspects of generalization: *domain shift*, *sample efficiency*, *action granularity*, and *task shift*. Concretely, we measure transfer under distribution shift on Something-Something V2 and FineGym (Gym99), low-data finetuning with 1K examples on UCF101 and FineGym, fine-grained discrimination on FX-S1 and UB-S1 splits of FineGym, and non-standard objectives via temporal repetition counting on UCFRep [57] and multi-label classification on Charades [43]. All experiments follow the official SEVERE protocols and are reported in Tab. 8. Implementation and training details are reported in the supplementary material.

Domain shift. On SSv2 and Gym99, TrackMAE with

Table 8. **Comparison on SEVERE generalization benchmark [46].** We compare prior video SSL methods on four generalization factors of SEVERE benchmark spanning a total of eight downstream settings. TrackMAE delivers consistently strong or superior performance across these diverse settings in both pixel and feature reconstruction modes, indicating that trajectory-guided masked pretraining improves robustness and generalization of learned video representations.

Method	Domain shift		Sample efficiency (10^3)		Action granularity		Task shift		Mean
	SSv2	Gym99	UCF	GYM	FX-S1	UB-S1	UCF-RC↓	Charades	
Pixel Reconstruction									
VideoMAE [49]	68.6	86.6	74.6	25.9	42.8	65.3	0.172	17.8	57.6
MVD [51]	70.0	82.5	67.1	17.5	31.3	50.5	0.184	16.1	52.1
MGMAE [23]	68.9	87.2	77.2	24.1	33.7	79.5	0.181	17.9	58.8
MGM [14]	71.1	89.1	78.4	26.4	38.6	86.9	**0.152**	22.5	62.2
TrackMAE	70.3	88.7	79.8	31.0	41.6	85.5	0.162	20.8	62.9
Feature Reconstruction									
MME [45]	70.1	89.7	79.2	29.8	55.5	87.2	0.155	23.6	65.0
SIGMA [40]	70.9	89.7	84.1	28.0	55.1	79.9	0.169	23.1	64.2
SMILE [48]	72.1	90.8	86.4	**35.1**	55.1	88.3	0.170	**32.5**	67.9
TrackMAE w/o motion	71.9	90.0	85.5	31.4	55.1	74.6	0.170	27.1	64.8
TrackMAE	**72.8**	**91.1**	**86.7**	34.4	**59.0**	**90.1**	0.170	30.5	**68.4**

CLIP+motion targets attains the strongest performance among masked video modeling approaches, slightly improving over SMILE and other feature-based baselines. Importantly, even the pixel-only TrackMAE variant improves over VideoMAE and remains competitive with motion-aware designs such as MGM and MGMAE, showing that explicit trajectory prediction enhances robustness to distribution shifts even when training purely in pixel space.

Sample efficiency. In the 1K-sample setting (UCF10^3, Gym10^3), TrackMAE maintains strong performance, confirming that trajectory-guided pretraining produces features that adapt well under limited supervision. The pixel-based TrackMAE variant already surpasses its pixel-only counterparts, indicating that injecting motion structure into masked prediction benefits low-shot recognition without relying solely on high-level feature targets.

Action granularity. For fine-grained splits FX-S1 and UB-S1, TrackMAE with CLIP+motion achieves the best results, and the pixel-based TrackMAE variant also improves over or is on par with motion-guided baselines like MGM and MGMAE. These gains underline that TrackMAE is particularly effective for capturing subtle temporal and spatial differences required in fine-grained action classification.

Task shift. For the task shift, TrackMAE shows improvements over baselines VideoMAE and MGMAE in the pixel space but is on par or slightly worst than the current best method SMILE in the feature reconstruction.

Summary. In the pixel space, TrackMAE outperforms all prior works with a significantly improving mean results by +5.3% over VideoMAE and by +4.1% over MGMAE, showing strong generalization capability. Furthermore, adding our motion prediction targets to CLIP-only reconstruction (denoted as TrackMAE w/o motion) results in a mean improvement of 3.4%, indicating the generalization capability is infact enhanced by the prediction of motion trajectories. Finally, TrackMAE with CLIP targets improves the previous state-of-the-art mean results by 0.5%.

5. Discussions

Computational cost. Point tracking comes at a non-negligible cost. In practice, we observe that the pretraining time increases by 50%. However, with stronger performance on linear probing and generalization, the tradeoff is reasonable. Furthermore, we are able to mitigate larger cost due to denser tracking with our upsampling strategy enabling our method to extract motion targets on the fly in a scalable manner.

Motion robustness. Point trackers are prone to incorrect prediction, often due to very high motion or occlusion. To evaluate the robustness of our method under jittered trajectories, we trained our model to reconstruct noisy predictions, either spatially or temporally, using Gaussian noise. In that setting, the performance degradation is around -0.5%, showing that even under noisy motion targets, our method is still able to learn meaningful motion-based representations.

6. Conclusion

We introduce TrackMAE, a new masked video modeling paradigm based on motion prediction. In particular, we leverage a sparse point tracker to a create motion-based reconstruction signal. To mitigate the computational cost needed to obtain denser trajectories, we show that spatially interpolating the tracker output yields better performance without additional cost. We also use the motion trajectories to propose a new motion-aware masking strategy that further improves the downstream performance. Across linear

probing, full fine-tuning, and SEVERE, TrackMAE shows consistent gains on appearance- and motion-centric tasks, translating into stronger discrimination and broader generalization than prior video SSL methods.

Acknowledgments

The present research benefited from computational resources made available on Lucia, the Tier-1 supercomputer of the Walloon Region, infrastructure funded by the Walloon Region under the grant agreement n°1910247. The research reported in this publication was supported by funding from King Abdullah University of Science and Technology (KAUST) - Center of Excellence for Generative AI, under award number 5940. For computing time, this research used Ibex managed by the Supercomputing Core Laboratory at King Abdullah University of Science & Technology (KAUST) in Thuwal, Saudi Arabia. We acknowledge EuroPC JU for awarding the project ID EHPC-DEV-2025D10-008 access to Leonardo on Leonardo Booster hosted by CINECA, Italy and access to MareNostrum5 on MN5 ACC hosted by BSC, Spain.

References

[1] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 15619–15629, Vancouver, Can., 2023. 2

[2] Wele Gedara Chaminda Bandara, Naman Patel, Ali Gholami, Mehdi Nikkhah, Motilal Agrawal, and Vishal M. Patel. AdaMAE: Adaptive masking for efficient spatiotemporal learning with masked autoencoders. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 14507–14517, Vancouver, Can., 2023. 1, 2

[3] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *Int. Conf. Learn. Represent. (ICLR)*, pages 1–18, Virtual conference, 2022. 1, 2

[4] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Int. Conf. Mach. Learn. (ICML)*, pages 813–824. ML Res. Press, 2021. 2

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hessei, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Sutskever Ilya, and Dario Amodei. Language models are few-shot learners. *arXiv*, abs/2005.14165, 2020. 1, 2

[6] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 139–156, 2018. 1

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 9912–9924, Virtual conference, 2020. Curran Assoc. Inc.

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Herve Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pages 9630–9640, Montréal, Can., 2021. 2, 3

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Int. Conf. Mach. Learn. (ICML)*, pages 1597–1607, 2020. 1

[10] Dading Chong, Helin Wang, Peilin Zhou, and Qingcheng Zeng. Masked spectrogram prediction for self-supervised audio pre-training. In *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, pages 1–5, Rhodes Island, Greece, 2023. 2

[11] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 886–893, San Diego, CA, USA, 2005. 2, 3

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. Conf. North Am. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, pages 4171–4186, Minneapolis, MN, USA, 2019. Assoc. Comput. Linguistics. 1, 2

[13] Alexandre Eymaël, Renaud Vandeghen, Anthony Cioppa, Silvio Giancola, Bernard Ghanem, and Marc Van Droogenbroeck. Efficient image pre-training with siamese cropped masked autoencoders. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 348–366, 2024. 2

[14] David Fan, Jue Wang, Shuai Liao, Yi Zhu, Vimal Bhat, Hector Santos-Villalobos, Rohith M V, and Xinyu Li. Motion-guided masking for spatiotemporal representation learning. In *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pages 5596–5606, Paris, Fr., 2023. 1, 2, 3, 5, 6, 8

[15] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 6804–6815, Montréal, Can., 2021. 2

[16] Christoph Feichtenhofer, haoqi fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 35946–35958, New Orleans, LA, USA, 2022. Curran Assoc. Inc. 2

[17] Rohit Girdhar, Alaaeldin El-Nouby, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. OmniMAE: Single model masked pretraining on images and videos. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 10406–10417, Vancouver, Can., 2023. 6, 1, 2

[18] Yuan Gong, Cheng-I Lai, Yu-An Chung, and James Glass. SSAST: Self-supervised audio spectrogram transformer. In *AAAI Conf. Artif. Intell.*, pages 10699–10709, Virtual venue, 2022. 2

[19] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 5843–5851, Venice, Italy, 2017. 1, 4

[20] Agrim Gupta, Jiajun Wu, Jia Deng, and Li Fei-Fei. Siamese masked autoencoders. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 1–18, New Orleans, LA, USA, 2023. Curran Assoc. Inc. 2

[21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 9726–9735, Seattle, WA, USA, 2020. 1

[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollar, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conf. Comput. Vis. Pattern*

Recognit. (CVPR), pages 15979–15988, New Orleans, LA, USA, 2022. 1, 2, 3

[23] Bingkun Huang, Zhiyu Zhao, Guozhen Zhang, Yu Qiao, and Limin Wang. MGMAE: Motion guided masking for video masked autoencoding. In *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pages 13447–13458, Paris, Fr., 2023. 1, 2, 3, 5, 6, 8

[24] Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 28708–28720, New Orleans, LA, USA, 2022. Curran Assoc. Inc. 2

[25] Sunil Hwang, Jaehong Yoon, Youngwan Lee, and Sung Ju Hwang. EVEREST: Efficient masked video autoencoder by removing redundant spatiotemporal tokens. In *Int. Conf. Mach. Learn. (ICML)*, pages 20889–20907, Vienna, Austria, 2024. 5

[26] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. In *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pages 6013–6022, Honolulu, HI, USA, 2025. 2, 3, 4

[27] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv*, abs/1705.06950, 2017. 1, 5, 4

[28] Hilde Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso A. Poggio, and Thomas Serre. HMDB: A large video database for human motion recognition. In *IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 2556–2563, Barcelona, Spain, 2011. 4

[29] Zihang Lai and Andrea Vedaldi. Tracktention: Leveraging point tracking to attend videos faster and better. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 22809–22819, Nashville, TN, USA, 2025. 2

[30] Kunchang Li, Yali Wang, Gao Peng, Guanglu Song, Yu Liu, Hongsheng Li, and Yu Qiao. UniFormer: Unified transformer for efficient spatial-temporal representation learning. In *Int. Conf. Learn. Represent. (ICLR)*, pages 1–19, Virtual conference, 2022. 1, 2

[31] Kunchang Li, Yali Wang, Yizhuo Li, Yi Wang, Yinan He, Limin Wang, and Yu Qiao. Unmasked teacher: Towards training-efficient video foundation models. In *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pages 19891–19903, Paris, Fr., 2023. 2

[32] Xueting Li, Sifei Liu, Shalini De Mello, Xiaolong Wang, Jan Kautz, and Ming-Hsuan Yang. Joint-task self-supervised learning for temporal correspondence. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 1–11, Vancouver, Can., 2019. Curran Assoc. Inc. 2

[33] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. MViTv2: Improved multiscale vision transformers for classification and detection. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 4794–4804, New Orleans, LA, USA, 2022. 1, 2

[34] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 3192–3201, New Orleans, LA, USA, 2022. 1, 2

[35] Cheng-Ze Lu, Xiaojie Jin, Zhicheng Huang, Qibin Hou, Ming-Ming Cheng, and Jiashi Feng. CMAE-V: Contrastive masked autoencoders for video action recognition. *arXiv*, abs/2301.06018, 2023. 1, 2

[36] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 1:1–32, 2024. 1, 3

[37] Mandela Patrick, Dylan Campbell, Yuki Asano, Ishan Misra, Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and João F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 12493–12506, Virtual conference, 2021. Curran Assoc. Inc. 2

[38] Zhiwu Qing, Shiwei Zhang, Ziyuan Huang, Xiang Wang, Yuehuan Wang, Yiliang Lv, Changxin Gao, and Nong Sang. MAR: Masked autoencoders for efficient action recognition. *IEEE Trans. Multimedia*, 26:218–233, 2024. 2

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Int. Conf. Mach. Learn. (ICML)*, pages 8748–8763, Virtual Conf., 2021. ML Res. Press. 2, 3

[40] Mohammadreza Salehi, Michael Dorkenwald, Fida Mohammad Thoker, Efstratios Gavves, Cees G. M. Snoek, and Yuki M. Asano. SIGMA: Sinkhorn-guided masked video modeling. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 293–312, 2024. 2, 5, 6, 7, 8, 1, 4

[41] Mohammadreza Salehi, Shashanka Venkataramanan, Ioana Simion, Efstratios Gavves, Cees G. M. Snoek, and Yuki M. Asano. MoSiC: Optimal-transport motion trajectory for dense self-supervised learning. In *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Honolulu, HI, USA, 2025. 2

[42] Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. FineGym: A hierarchical video dataset for fine-grained action understanding. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 2613–2622, Seattle, WA, USA, 2020. 1, 4, 5

[43] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 510–526, 2016. 7, 5

[44] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv*, abs/1212.0402, 2012. 1, 4, 5

[45] Xinyu Sun, Peihao Chen, Liangwei Chen, Changhao Li, Thomas H. Li, Mingkui Tan, and Chuang Gan. Masked motion encoding for self-supervised video representation learning. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 2235–2245, Vancouver, Can., 2023. 1, 2, 5, 6, 8

[46] Fida Mohammad Thoker, Hazel Doughty, Piyush Bagad, and Cees G. M. Snoek. How severe is benchmark-sensitivity in video self-supervised learning? In *Eur. Conf. Comput. Vis. (ECCV)*, pages 632–652, 2022. 1, 7, 8, 4, 5

[47] Fida Mohammad Thoker, Letian Jiang, Chen Zhao, Piyush Bagad, Hazel Doughty, Bernard Ghanem, and Cees G. M. Snoek. SEVERE++: Evaluating benchmark sensitivity in generalization of video representation learning. *arXiv*, abs/2504.05706, 2025. 1, 7

[48] Fida Mohammad Thoker, Letian Jiang, Chen Zhao, and Bernard Ghanem. SMILE: Infusing spatial and motion semantics in masked video learning. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 8438–8449, Nashville, TN, USA, 2025. 2, 5, 6, 7, 8, 1, 4

[49] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, pages 10078–10093, New Orleans, LA, USA, 2022. Curran Assoc. Inc. 1, 2, 3, 5, 6, 8, 4

[50] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. VideoMAE V2: Scaling video masked autoencoders with dual masking. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 14549–14560, Vancouver, Can., 2023. 2

[51] Rui Wang, Dongdong Chen, Zuxuan Wu, Yinpeng Chen, Xiyang Dai, Mengchen Liu, Lu Yuan, and Yu-Gang Jiang. Masked video distillation: Rethinking masked feature modeling for self-supervised video representation learning. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 6312–6322, Vancouver, Can., 2023. 1, 5, 8

[52] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *IEEE Int. Conf. Comput. Vis. (ICCV)*, pages 2794–2802, Santiago, Chile, 2015. 2

[53] Xiaolong Wang, Allan Jabri, and Alexei A. Efros. Learning correspondence from the cycle-consistency of time. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 2561–2571, Long Beach, CA, USA, 2019. 2

[54] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 14648–14658, New Orleans, LA, USA, 2022. 2

[55] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. SimMIM: a simple framework for masked image modeling. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 9643–9653, New Orleans, LA, USA, 2022. 2

[56] Haosen Yang, Bin Huang, Deng andi Wen, Jiannan Wu, Hongxun Yao, Yi Jiang, Xiatian Zhu, and Zehuan Yuan. MotionMAE: Self-supervised video representation learning with motion-aware masked autoencoders. In *Br. Mach. Vis. Conf. (BMVC)*, pages 1–14, Glasgow, UK, 2024. Br. Mach. Vis. Assoc. (BMVA). 2

[57] Huaidong Zhang, Xuemiao Xu, Guoqiang Han, and Shengfeng He. Context-aware and scale-insensitive temporal repetition counting. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 667–675, Seattle, WA, USA, 2020. 7, 5

[58] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. Image BERT pre-training with online tokenizer. In *Int. Conf. Learn. Represent. (ICLR)*, pages 1–29, Virtual conference, 2022. 1, 2

TrackMAE: Video Representation Learning via Track Mask and Predict

Supplementary Material

7. Detailed SOTA Comparison

In the main paper, we restricted comparisons to self-supervised masked video models with the same backbone and similar pretraining schedule. We now broaden that comparison to include both self-supervised (MVM variants with pixels/HOG/DINO/CLIP targets) and supervised baselines on K400 and SSv2, reported with the ViT-B backbone and a range of pretraining epochs. We evaluate in the full finetuning setup on K400 and SSv2 datasets to assess in-domain and cross-domain transfer. The results are shown in Table 9.

Kinetics-400 (in-domain pretraining and finetuning). On K400, TrackMAE attains 83.6 Top-1 with 600 epochs, outperforming all listed masked-video baselines trained for equal or longer schedules: VideoMAE [49] (80.0/81.5 at 800/1600), CMAE-V [35] (80.2/80.9), OmniMAE (80.8), MGM (80.8/81.7), MGMAE [23] (81.2/81.8), MME [45] (81.5/81.8), and SIGMA [40] (81.5). TrackMAE also edges SMILE [48] (83.1 at 600 and 83.4 at 1200), indicating a stronger accuracy–compute trade-off at shorter schedules. Compared to supervised architectures, TrackMAE surpasses MViTv2-B [33] (82.9) and Uniformer-B [30] (83.0), despite using the standard ViT-B backbone and a self-supervised objective.

SSv2 (cross-domain: K400→SSv2). When pretrained on K400 and finetuned on SSv2, TrackMAE reaches 72.8, improving over SMILE [48] (72.1 at 600; 72.4 at 1200) and clearly ahead of VideoMAE [49] (68.5), SIGMA [40] (71.1), MGM [14] (71.1), and MGMAE [23] (68.9). It also exceeds the supervised counterparts reported under the same column (e.g., VideoSwin [34] 69.6, MViTv2-B [33] 70.5, Uniformer-B [30] 71.2). These results indicate better domain transfer to motion-centric SSv2, consistent with the hypothesis that explicit motion supervision complements CLIP-space semantics.

SSv2 (in-domain: SSv2→SSv2). With SSv2 pretraining, TrackMAE attains 72.5, matching the best reported MVM result in the table (SMILE [48] at 72.5) and exceeding other MVM baselines like MGMAE [23] (72.0 at 1600), MGM [14] (71.8 at 1600), CMAE-V [35] (70.5 at 1600), VideoMAE [49] (69.6 at 800), and OmniMAE [17] (69.5). This suggests TrackMAE's motion signal remains beneficial even when the pretraining domain already contains substantial temporal variation.

In summary, TrackMAE achieves state-of-the-art results among masked-video models under comparable settings and is competitive with, or outperforms, strong supervised models. The pattern supports the central claim that explicit motion supervision paired with CLIP-space reconstruction produces video representations that are both semantically strong and motion-aware, yielding robust transfer in-domain and under-domain shift.

8. Additional Ablations

Impact of masking. Table 10a shows the impact of the masking ratio used during pretraining on the finetuning performance. We observe that a masking ratio of 90%, similar to previous methods [23, 49], shows that learning from highly masked spatial and *trajectory* tokens works well in practice.

Impact of separate decoding. We study in Tab. 10b the impact of training with a joint or separate decoders to reconstruct both the spatial and trajectory targets. As shown in the table, separate decoders work best, which may indicate that involving the same decoder for both tasks may lead to information leakage, degrading the signal.

Impact of noisy tracks. As mentioned in the main paper that our method shows robustness against noisy target tracks. The results are shown in Tab. 10c. We observe that our method is robust for both spatial and temporal noise, meaning that even under a noisy tracker, our model would still be able to learn meaningful features.

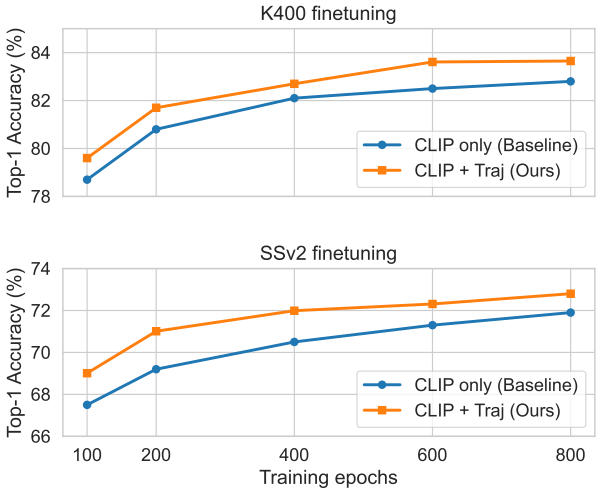

Figure 4. **Training evolution.** We report the Top-1 Accuracy for K400 and SSv2 finetuning at different pretraining epochs. Our model trained with both CLIP and trajectory reconstructions consistently outperforms the CLIP reconstruction only.

Table 9. **Detailed SOTA comparison of masked video modeling methods on Something-Something V2 and Kinetics-400 for full finetuning action recognition.** Our TrackMAE outperforms many supervised approaches and achieves the best performance masked video modeling methods with similar pretraining setups.

				SSv2 Pretraining	K400 Pretraining	
Method	**Backbone**	**Targets**	**Epochs**	**SSv2 Top-1**	**SSv2 Top-1**	**K400 Top-1**
Supervised						
Mformer [37]	Mformer-B	-	-	-	66.7	79.7
VideoSwin [34]	Swin-B	-	-	-	69.6	80.6
TimeSformer [4]	ViT-B	-	-	-	59.5	80.7
MViTv1 [15]	MViTv1-B	-	-	-	67.7	80.2
MViTv2 [33]	MViTv2-B	-	-	-	70.5	82.9
Uniformer-B [30]	Uformer-B	-	-	-	71.2	83.0
Self-supervised						
VideoMAE [49]	ViT-B	Pixel	800	69.6	68.5	80.0
VideoMAE [49]	ViT-B	Pixel	1600	69.6	-	81.5
CMAE-V [35]	ViT-B	Pixel	800	69.7	-	80.2
CMAE-V [35]	ViT-B	Pixel	1600	70.5	-	80.9
OmniMAE [17]	ViT-B	Pixel	800	69.5	69.0	80.8
MGM [14]	ViT-B	Pixel	800	70.6	71.1	80.8
MGM [14]	ViT-B	Pixel	1600	71.8	-	81.7
MGMAE [23]	ViT-B	Pixel	800	71.0	68.9	81.2
MGMAE [23]	ViT-B	Pixel	1600	72.0	-	81.8
MME [45]	ViT-B	HOG	800	70.0	70.5	81.5
MME [45]	ViT-B	HOG	1600	-	-	81.8
SIGMA [40]	ViT-B	DINO	800	71.2	71.1	81.5
SMILE [48]	ViT-B	CLIP	600	**72.5**	72.1	83.1
SMILE [48]	ViT-B	CLIP	1200	-	72.4	83.4
TrackMAE (Ours)	ViT-B	CLIP	600	**72.5**	**72.8**	**83.6**

Table 10. **Additional ablations on our proposed TrackMAE.** The default setting uses pixel reconstruction and motion prediction with a grid size of 14, masking ratio of 90%, $\lambda = 1$, and two separate decoders.

Ratio	$K400_s$	$SSv2_s$
95%	48.1	54.4
90%	**49.4**	**56.2**
80%	49.2	56.0

(a) **Masking ratio.** Masking 90% of the tokens, as in previous methods, works best.

Decoder	$K400_s$	$SSv2_s$
Joint	48.7	55.5
Separate	**49.4**	**56.2**

(b) ***Joint* or *Separate* decoders**. Using separate decoder for motion prediction outperforms using the joint decoder.

Noise	$K400_s$	$SSv2_s$
None	**49.4**	**56.2**
Spatial	48.9	55.7
Temporal	48.8	55.8

(c) **Impact of noisy tracks.** Adding spatial or temporal noise to the tracker trajectories does not affect our method much.

9. Convergence Results

We analyze the convergence behavior of our approach compared to the baseline. We evaluate our CLIP + Trajectory and CLIP-only reconstruction at different pretraining epochs on both K400 and SSv2 finetuning tasks. Figure 4 shows that our model trained with both the spatial and trajectory targets consistently outperforms the model trained with the spatial targets only. We observe that with fewer epochs, our performance is much better than the baseline; however, the gap is narrowed with more pretraining epochs.

10. Discussion on Motion Masking

In this section, we expand the discussion on our motion-aware strategy. In particular, we expand the discussion on different sampling distributions Sec. 10.1, and the different sampling strategies Sec. 10.2 based on motion trajecotories.

Figure 5. **Sampling strategies.** We show how we can use the motion information to create different sampling distribution.

10.1. Sampling Distribution in Masking

As mentioned in the main paper, we use the motion trajectories to create a sampling distribution. We explore two different ways to do this: (1) we accumulate the displacement made by each point of the grid through time with respect to the first frame, (2) we accumulate the displacement made by each point of the grid through time in the next consecutive frame. In other words, the first strategy gives the average motion information over time of *how much* a given point is moving, without taking into account the trajectory. The second strategy gives the average motion information over time of *how much* and *where* a given point is moving. Those 2 sampling strategies are depicted in Fig. 5.

Both sampling strategies have their own merits, *i.e*. the first strategy will put some emphasis towards tokens that are likely to move, without guarantee that they are seen in the following frames. Our intuition is that it forces the model towards learning some motion dynamics of the different moving objects. For the second strategy, our intuition follows the same reasoning, but with visible tokens sampled along the trajectory. In practice, we see both version works on par, with slightly better results for the first strategy, which is used for the results in the main paper. It is also worth mentioning that whatever the strategy used, it is always impacted by the input data. We observe that the K400 dataset is prone to have erratic movements, which may lead to poor motion information. In such a case, our sampling distribution tends to behave more like a uniform sampling, thus falling back to the original random tube masking strategy.

10.2. Sampling Strategy in Masking

Besides different sampling strategy, there are also many different ways to sample visible tokens from it. For computational reasons and for its intuitive soundness, we only evaluated the motion bins strategy described in the main paper. However, we describe below some other strategies.

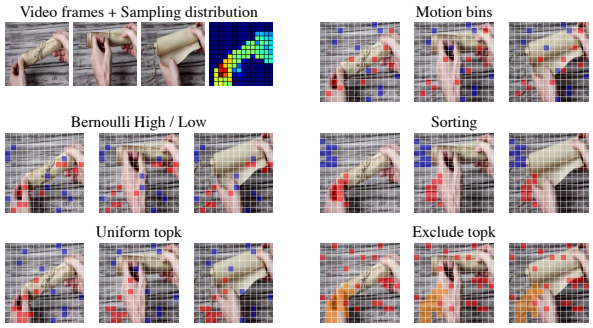

Figure 6. **Masking strategies.** We show how we can use our sampling distribution to create different masking maps. Red squares are high-motion visible tubes, blue squares are low-motion visible tokens. For the "exclude topk" strategy, we also show orange the tokens excluded from being sampled.

Motion bins. We sample visible tokens from high- and low-motion regions of the sampling distribution using 2 uniform bins. Depending on the value of ρ_{motion}, we control the number of tokens sampled coming from each bins. Our intuition is that this strategy gives, on average, a good balance between high- and low-motion visible and masked tokens. This is the strategy used in the main paper.

Bernoulli High/Low. Instead of uniformly sampling from high- and low-motion regions, we can sample using a Bernoulli distribution. For the high-motion tokens, we directly sample from the sampling distribution, and for the low-motion token, we sample from the "1-motion" distribution, excluding tokens already sampled from the high-motion part. Similarly as for the motion bins strategy, we can use the same parameter to control the number of tokens for each high- and low-motion regions. However, we visually observe that this strategy tends to create blob regions, which would usually hurts training.

Sorting. Based on the previous strategy, we can directly sort high- and low-motion tokens and follow their ordering, instead of sampling from the distribution. This strategy creates bigger blobs, which we believe would hurt even more the training.

Uniform topk. To find a better balance between Bernoulli and sorting, we can uniformly sample from the Top-k most moving tokens. However, in order to sample from a smaller set than in the motion bins strategy, we need to specify how many samples are used in the Top-k operation, adding a new hyperparameter to tune.

Exclude topk. In contrast to previous strategies, which use the motion to guide the sampling of visible tokens, we can use the motion information to decide where *we should not* sample visible tokens. From the Top-k most moving tokens, we can chose to explicitly remove them from the sampling

Table 11. **Datasets details.** Splits of datasets used in full finetuning and linear probing.

Dataset	Abbrev.	#Classes	#Train	#Test
Kinetics-400	K400	400	240K	19K
UCF-101	UCF	101	9.5K	3.8K
HMDB-51	HMDB	51	4.8K	2K
Something-Something V2	SSv2	174	169K	24.8K
FineGYM	GYM	99	20.5K	8.5K

distribution, as learning to reconstruct those tokens may be interesting. Then, we can use any strategy to sample the visible tokens. All sampling strategies are shown in Fig. 6

We leave the exploration of such masking strategies, their impact on different pretraining data and downstream tasks for future work.

11. Experimental Details

11.1. Datasets

Kinetics-400 [27] (K400) is a large-scale YouTube-sourced corpus with 400 human action categories and over 306k short clips. It remains a canonical benchmark for learning generalizable video representations.

Something-Something V2 [19] (SSv2) emphasizes object-centric, first-person interactions that differ markedly from K400's Internet footage. It comprises 168,913 training and 24,777 test samples spanning 174 categories, stressing temporal reasoning and commonsense dynamics.

UCF-101 [44] (UCF) collects 9,537 training and 3,783 test clips from YouTube across 101 action classes. Although coarser in granularity and overlapping with K400 categories, it is widely used for transfer evaluation in self-supervised video learning.

HMDB-51 [28] (HMDB) contains 6,766 clips from varied sources (films, archives, web videos) covering 51 classes (at least 100 videos per class). Its heterogeneity challenges models to cope with diverse cinematic and real-world content.

FineGYM [42] (GYM) targets fine-grained action understanding in gymnastics. We use the Gym-99 subset (99 classes) with 20,484 training and 8,521 test samples, focusing on subtle motion differences within highly structured routines.

SEVERE Benchmark [46] SEVERE aggregates eight evaluation settings across SSv2, UCF, FineGYM, and Charades to stress sample efficiency, granularity, and task shift. Table 12 details each subset and metric.

11.2. Training and Evaluation Details

Pretraining Details. We pretrain on K400 [27] and SSv2 [19]. Following VideoMAE [49], we sample clips of 16 frames at 224×224 with a temporal stride of 2 on SSv2 and 4 on K400. We compute space–time tube tokens via a 3D convolution, treating each $2\times16\times16$ cube as a token. For each sampled 16-frame clip, we temporally downsample it with a stride of 2 to extract the trajectories from CoTracker3 [26], matching the total number of trajectory tokens with space-time cubes. In practice, we use the normalized temporal differences of the extracted motion trajectories as the target, rather than absolute values.

All hyperparameters are shown in Table Tab. 13, following [48, 49]. All pretraining runs use $16\times$NVIDIA A100 GPUs. For downstream tasks, we discard the decoders and use only the pretrained encoder with a task-specific head (*e.g.*, a linear classifier for action recognition).

Linear probing details. We strictly follow the settings in [48] and train the linear head on top of the frozen backbone with the target dataset, as shown in Tab. 14. Experiments are run on $4\times$V100 GPUs.

Full finetuning details, We strictly follow the settings in [48] and train the backbone + head with the taget dataset, as shown in Tab. 15. Experiments are run on $4\times$V100 GPUs.

SEVERE benchmark details. Following [40, 48], we evaluate with the official SEVERE codebase [46], strictly reusing the provided training and evaluation configurations to ensure a fair comparison, also shown in Tab. 15. Experiments are run on $4\times$V100 GPUs.

Table 12. **SEVERE benchmark.** Subsets, protocols, and metrics following [46].

Dataset	Experiment	Setup Group	Task	#Classes	Finetune	Test	Metric
FineGym [42]	Gym99	Full	Action Class.	99	20,484	8,521	Top-1 Acc.
UCF 101 [44]	UCF (10^3)	Sample Efficiency	Action Class.	101	1,000	3,783	Top-1 Acc.
FineGym [42]	Gym (10^3)	Sample Efficiency	Action Class.	99	1,000	8,521	Top-1 Acc.
FineGym [42]	FX-S1	Action Granularity	Action Class.	11	1,882	777	Mean-per-class
FineGym [42]	UB-S1	Action Granularity	Action Class.	15	3,511	1,471	Mean-per-class
UCFRep [57]	UCF-RC	Task Shift	Repetition Counting	–	421	105	Mean Error
Charades [43]	Charades	Task Shift	Multi-label Class.	157	7,985	1,863	mAP

Table 13. **Pretraining configuration.**

Shared		
Optimizer	AdamW	
Base learning rate	1.5×10^{-4}	
Weight decay	0.05	
Momentum (Betas)	β_1=0.9, β_2=0.95	
Batch size	512	
LR schedule	cosine decay	
Warmup epochs	40	
Augmentation	MultiScaleCrop(1, 0.875)	
Dataset-specific	**Epochs**	**FlipAug.**
SSv2	800	no
K400	600	yes

Table 14. **Linear probing configuration.**

config	K400	HMDB	SSv2	GYM
optimizer	AdamW			
base learning rate	1×10^{-3}			
weight decay	0.05			
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$			
layer-wise lr decay	0.75			
batch size	128			
learning rate schedule	cosine decay			
training epochs	30	100	50	100
flip augmentation	*yes*	*yes*	*no*	*yes*

Table 15. **Full finetuning configuration.**

Parameter	**Value**		
Optimizer	AdamW		
Base learning rate	1.0×10^{-3}		
Weight decay	0.05		
Momentum (Betas)	$\beta_1 = 0.9,\ \beta_2 = 0.999$		
Layer-wise LR decay	0.75		
LR schedule	cosine decay		
Warmup epochs	5		
RandAug	(9, 0.5)		
Label smoothing	0.1		
Mixup	0.8		
CutMix	1.0		
Drop path	0.1		
Dataset-specific	**Batchsize**	**Epochs**	**FlipAug.**
SSv2	32	40	no
K400	16	100	yes
SEVERE	16	100	yes