
Supplementary Material - Chirality in Action: Time-Aware Video Representation Learning by Latent Straightening

Anonymous Author(s)

Affiliation

Address

email

1 Contents

2	A Dataset: Chirality in Action (CiA)	1
3	B Model: Linearized Feature Trajectories (LiFT)	5
4	C Experiments	6
5	C.1 Setup details	6
6	C.2 Additional Ablations	6
7	C.3 Qualitative results	7

8 A Dataset: Chirality in Action (CiA)

9 **Metadata and examples.** In Table A.1, we show the chiral groups constructed in SSv2. Similarly,
10 we construct 66 chiral groups in EPIC and 28 groups in Charades. In Table A.2, we show the number
11 of videos in the chiral splits of each of the three datasets. We attach the chiral splits for all three
12 datasets as part of the Supplemental. We also provide a single CSV file that includes the chiral groups
13 for all three combined data sets. We also show more examples of chiral pairs from each of the three
14 datasets in Fig. A.1, Fig. A.2 and Fig. A.3. In general, since SSv2 has single canonical actions, it is
15 a cleaner test bed for chiral action recognition. EPIC and Charades usually have a more cluttered
16 visual context where cues for chiral recognition are more subtle.

17 **Time-sensitivity of CiA.** In Fig. A.4 and Fig. A.5, we repeat the experiments to check time-
18 sensitivity (Fig 5 in the main paper) of the CiA benchmark on all three datasets. Our inferences about
19 time-sensitivity hold for all three datasets.

Verb →	Verb ←	Noun (object)
Pulling [something] from left to right	Pulling [something] from right to left	['something']
Pushing [something] from left to right	Pushing [something] from right to left	['something']
Turning the camera left while filming [...]	Turning the camera right while filming [...]	['something']
Approaching [something] with your camera	Moving away from [something] with your camera	['something']
Closing [something]	Opening [something]	['object']
Closing [something]	Opening [something]	['door']
Closing [something]	Opening [something]	['bottle']
Closing [something]	Opening [something]	['book']
Closing [something]	Opening [something]	['purse']
Closing [something]	Opening [something]	['drawer']
Moving [...] and [...] away from each other	Moving [...] and [...] closer to each other	['something']
Moving [something] away from the camera	Moving [something] towards the camera	['something']
Moving [something] down	Moving [something] up	['something']
Putting [something similar to other things ...]	Taking [one of many similar things on the table]	['something']
Turning the camera downwards while filming [...]	Turning the camera upwards while filming [...]	['something']
Folding [something]	Unfolding [something]	['something']

Table A.1: **Chiral groups in SSv2.** We construct 16 chiral groups in SSv2 by identifying temporally opposite verbs. Note that “opening vs. closing” is split across different objects representing entirely different actions. Noun “[‘something’]” denotes a placeholder which can include any appropriate object that fits with the action verb.

Dataset	Chiral groups	Total videos		Avg. videos per chiral group		Avg. duration (s)
		Train	Validation	Train	Validation	
SSv2	16	12216	1430	763.5	89.4	3.6
EPIC	66	24101	3108	365.1	47.1	1.6
Charades	28	16018	5498	572.1	196.4	8.6

Table A.2: **CiA dataset size.** For each of the constituent datasets, we show the total number of videos in the proposed chiral split and also the average number of videos per chiral group. Note that we train one linear probe for each chiral group.



Figure A.1: **CiA samples from SSv2.** More examples of chiral pairs from the train set of SSv2. The positive direction actions are marked in blue while the negative direction ones are marker in red.



Figure A.2: **CiA samples from EPIC.** More examples of chiral pairs from the train set of EPIC. The positive direction actions are marked in blue while the negative direction ones are marker in red.



Figure A.3: **CiA samples from Charades.** More examples of chiral pairs from the train set of Charades. The positive direction actions are marked in blue while the negative direction ones are marker in red.

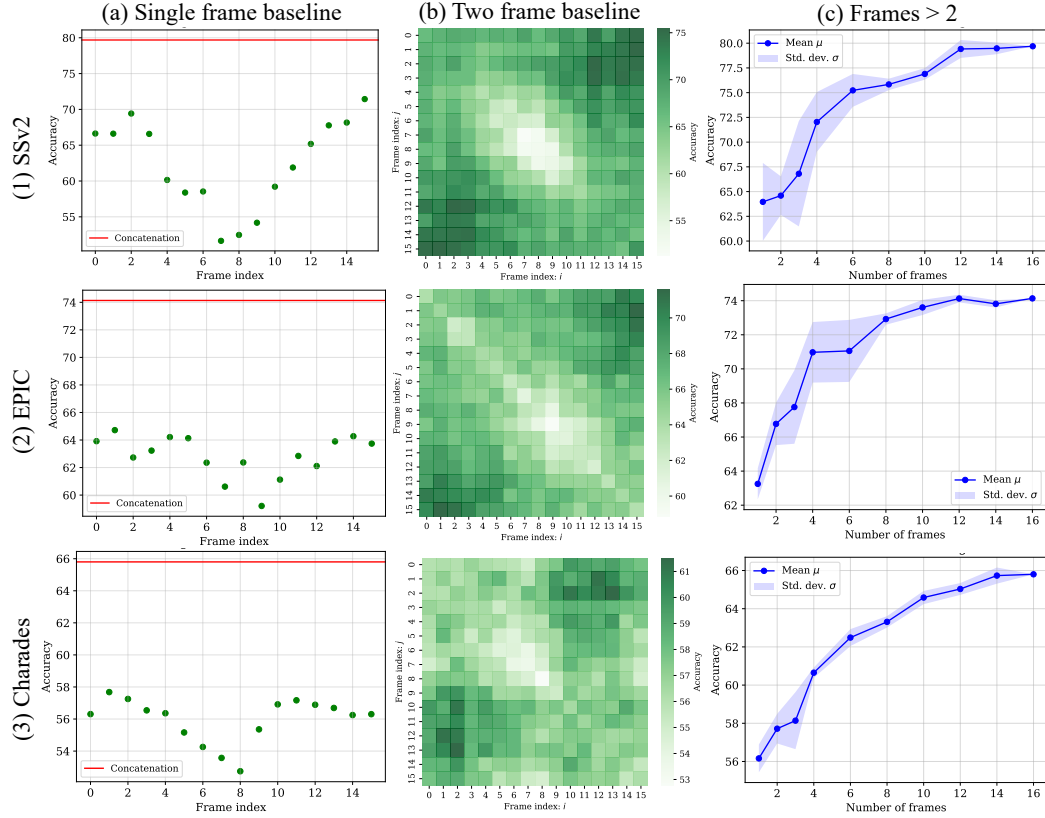


Figure A.4: **Time-sensitivity of CiA: Part 1.** We repeat the experiment shown in Fig 5 (a)/(b) of the main paper for all datasets. Rows represent datasets while columns represent different properties of the task. (a) A single-frame baseline tends to do well on frames at the end of the video sequence since those usually encode either the start or end state of the action. (b) A two-frame baseline usually does best if the frames are picked at the two ends of the video. (c) As more frames are considered in the context, accuracy on chiral action recognition improves. Overall, these demonstrate that chiral action recognition is time-sensitive: it benefits predictably from more frames, especially at the ends.

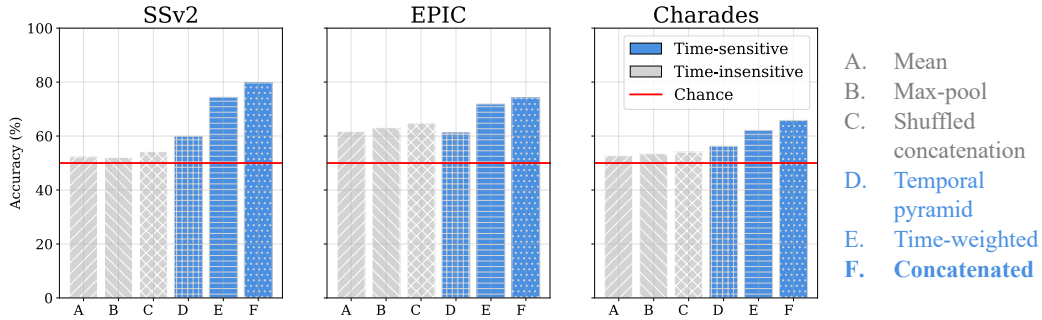


Figure A.5: **Time-sensitivity of CiA: Part 2.** We repeat the experiment shown in Fig 5 (c) of the main paper across all three datasets. We show that time-insensitive pooling of per-frame features (*e.g.*, average pooling) leads to much worse performance than with time-sensitive pooling (*e.g.*, concatenation) on chiral action recognition. Note that all the pooling methods considered are non-parametric. This demonstrates the time-sensitivity of chiral action recognition since incorporating the time order of frames substantially improves performance.

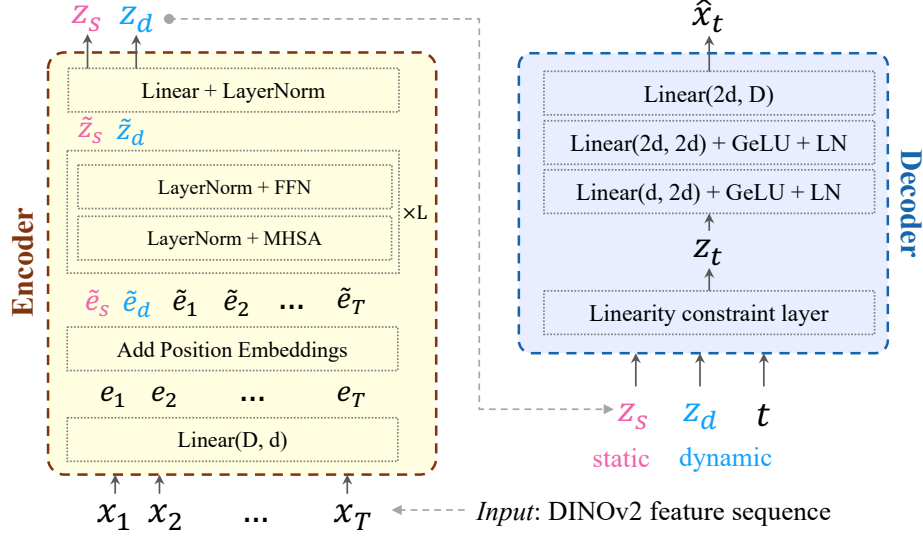


Figure B.1: **LIFT architecture details.** The encoder takes in a sequence of DINOv2 features and outputs a video descriptor disentangled into static and dynamic vectors. The decoder reconstructs the feature sequence with linearity baked in the latent space.

20 B Model: Linearized Feature Trajectories (LiFT)

21 **Architecture.** A detailed sketch of the architecture is provided in Fig. B.1. In LiFT, the Encoder
 22 takes in a sequence of features $\{x_t\}_t$ and outputs two descriptor tokens z_s, z_d . First, a linear layer
 23 projection is applied $\mathbb{R}^D \rightarrow \mathbb{R}^d$. Then, Sinusoidal position encoding is added representing frame
 24 index t . The two CLS tokens e_s, e_d are initialized randomly. Then, the CLS tokens along with the
 25 sequence tokens are passed through a Transformer with $L=4$ blocks and $H=8$ heads each. Each
 26 block has a multi-head self-attention (MHSA) layer followed by a FFN layer. Both the layers are
 27 preceded by LayerNorm layers. Then, the outputs for the two CLS tokens are projected with a linear
 28 layer ($\mathbb{R}^d \rightarrow \mathbb{R}^d$) followed by LayerNorm. This gives the latent descriptors z_s and z_d .

29 The decoder takes in z_s, z_d, t and outputs \hat{x}_t . First, we construct an intermediate representation for
 30 the frame at index t using our linearity constraint in the latent space.

$$z_t = z_s + (t/T) \cdot z_d \quad (1)$$

31 Then, this is passed to an MLP network with two hidden layers each followed by GeLU activation
 32 and LayerNorm. The first hidden layer maps $\mathbb{R}^d \rightarrow \mathbb{R}^{2d}$ and the second layer maps $\mathbb{R}^{2d} \rightarrow \mathbb{R}^{2d}$.
 33 This is followed by a linear projection ($\mathbb{R}^{2d} \rightarrow \mathbb{R}^D$) back to the DINOv2 space.

34 **Compute resources.** In order to train LiFT, we first compute and store feature vectors for DINOv2
 35 ViT-S/14. This feature computation is run on 4 NVIDIA RTX A4000 16GB GPUs in parallel. It
 36 takes about 12 GPU hours to compute features for 250K videos in Kinetics-400. Once features are
 37 computed, LiFT is trained on a single consumer-grade GPU (e.g., NVIDIA RTX A4000, Tesla P40,
 38 Quadro RTX 8000, NVIDIA RTX A6000). A single training run takes about 15 GPU hours.

C Experiments

C.1 Setup details

Details for chiral action recognition. To benchmark a given video model for chiral action recognition, we require a single descriptor vector for a video. There are two important details here: (i) *input processing pipeline*: Different methods differ in the way they sample frames, apply cropping operations, etc. (ii) *pooling*: existing methods [1, 8, 2] usually only represent short clips (sequence of frames with a fixed stride), so we need a way of pooling clip-level descriptors into a video-level descriptor. For (i), we follow the data pipeline for each model as provided. For (ii), depending on the method, we either average pool per-clip representations following [5] (e.g., for VideoMAEv2 [9]) or concatenate them (e.g., for 3D ResNet methods like TCLR [2]), or we hand-craft a pooling mechanism (e.g., averaging spatial tokens for each frame and concatenating across time for InternVideo2.5 [10]). Investigating a general pooling method that gives more time-aware descriptors is an avenue we leave for future work. For image-based model, we sample T frames linearly and simply concatenate per-frame features to represent the video.

Details for standard action recognition. For the experiments with probing video models [4, 8, 11] with LiFT, we sample a single clip of $T=16$ frames with a stride of $s=4$, resize the short side and center crop to (224, 224). Since VideoMAE, VideoJEPA and InternVideo2.5 all produce a sequence of space-time tokens without any global CLS token, we compute the average of all tokens to represent the video in case of linear/non-linear probing. Then, we concatenate the LiFT descriptor with this descriptor and train a classifier head on top to output the action class. In case of *linear probe*, the classifier head is a linear layer. In case of *non-linear probing*, it is a two layer MLP with 512 hidden dimensions with ReLU non-linearity and Dropout of 0.1. In case of an attentive probe, following [1], we train a single attention layer with a learnable query to pool the space-time tokens into a single descriptor. Then, LiFT is concatenated with the query vector and a linear classifier layer is added on top of the concatenation. We train the probe for 100 epochs using Adam optimizer with learning rate of $1e^{-5}$ and LRPlateau scheduler.

C.2 Additional Ablations

Varying the latent dimension d . In Tab. 3(a), we vary the latent dimension of the Encoder in LiFT. While the number of parameters increases with d , we find that with $d = 384$, LiFT achieves the best performance while still being compact and containing only 8.7M parameters. Note that we do not account for the fixed DINOv2 parameters (22.1M) in this experiment.

Varying the amount of training data. In Tab. Appendix C.2 (b), we vary the amount of training data used to train LiFT with the unsupervised reconstruction loss. We fix the latent dimension to be $d = 384$ and note that the model capacity is fixed. For each row, we run the experiment with three different random seeds and report the average and standard deviation in accuracy. Surprisingly, even with 10% of the data, LiFT gets to 83.3% accuracy. With increasing data samples, the mean accuracy increases marginally. We hypothesize that this is due to two reasons. (i) The model size remains fixed (8.7M) and may not have the capacity to significantly benefit from more samples, (ii) since Kinetics-400 is known to be biased to static understanding (single frame or unordered set of frames [3, 6]), for videos with little visual change, reconstructing the per-frame feature trajectories may not have sufficient signal to inform LiFT. This experiment raises some interesting research questions. Is it possible to achieve time-sensitive video representations by training on selected, *temporally hard* samples only? Is using synthetic data with hand-crafted temporal patterns sufficient [7, 12]? We leave these questions for future work.

Error bars. To compute error bars, we train LiFT on Kinetics-400 with five different random seeds. The training configuration is kept constant across all runs except the change in seed. Then, we evaluate the trained models on our main task: chiral action recognition as described in the main paper across the three datasets, SSv2, Charades and EPIC-Kitchens. We report the mean and standard deviation in accuracy in

Dataset	Accuracy (%)
SSv2	86.1 \pm 0.3
EPIC	76.5 \pm 0.8
Charades	70.3 \pm 0.6

Figure C.1: **Error bars for LiFT.** Mean \pm standard deviation of accuracy across five random seeds. LiFT remains fairly stable and the error bars emphasize the difference between LiFT and other video models.

Latent dim d	Accuracy	Parameters (M)
192	85.9	2.3
256	85.8	4.0
384	86.6	8.7
512	84.9	15.3

(a) Varying the latent dimension d of Encoder.

Data frac.	Accuracy
0.1	83.3 ± 0.1
0.2	84.4 ± 0.2
0.4	85.9 ± 0.4
0.6	85.6 ± 0.6
0.8	85.8 ± 0.8
1.0	86.2 ± 0.4

(b) Varying the % train data.

Table C.1: **Ablations.** Both ablations are conducted on the chiral subset of SSv2. In (a) we vary the latent dimension of the LiFT encoder. We find the best performance with $d = 384$. In (b), we vary the amount of training data (Kinetics-400) used to adapt LiFT. The given % is uniformly randomly chosen from the entire dataset. Surprisingly, even with 10% of the data, LiFT gets to 83% accuracy. We hypothesize that at fixed model capacity, scaling up to more samples gives diminishing returns.

91 Fig. C.1. The table illustrates these results, highlight-
 92 ing the consistency of the model’s performance.

93 C.3 Qualitative results

94 In Fig. C.2, we show more examples with tSNE em-
 95 beddings of the LiFT reconstructed feature trajectories. In most cases, LiFT reconstructs a smoother,
 96 continuous approximation of the original trajectory. Note that the original trajectory points seem more
 97 scattered than they actually are because tSNE optimizes for local neighborhood distances, which are
 98 dominated by closeness of points in the reconstructed trajectory. In case of Fig. C.2(f), we observe
 99 a divergence between the true and reconstructed trajectories. In this case, the model likely fails to
 100 capture the (subtle) visual change which likely causes \mathbf{z}_d to be inaccurate leading to the discrepancy.

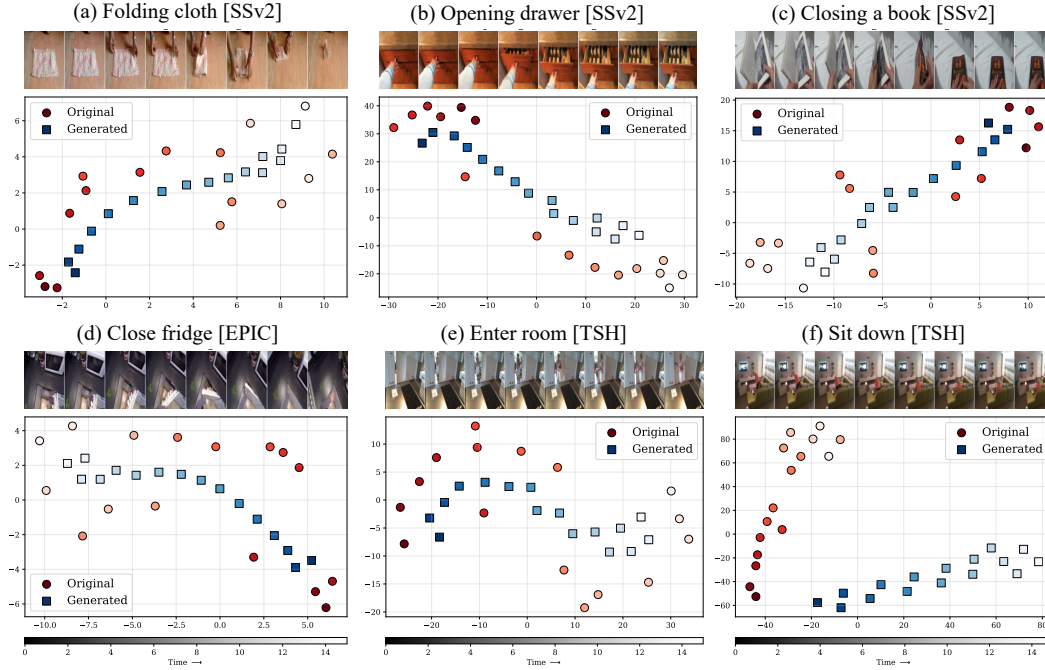


Figure C.2: **More qualitative samples of reconstructed features.** We show tSNE embeddings of original and reconstructed features for six videos. The red circles represent original features while blue squares represent reconstructions. Gradient of the color encodes the frame index (time). In general, LiFT tends to output a smooth, continuous approximation of the original feature trajectories in DINO space. (f) is an example failure case where the static token seems reasonable but the direction token is inaccurately predicted causing the direction of original and reconstructed trajectories to differ.

References

- [1] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL <https://openreview.net/forum?id=QaCCuDfBk2>. Featured Certification. 6
- [2] Ishan Dave, Rohit Gupta, Mamshad Nayeem Rizve, and Mubarak Shah. Tclr: Temporal contrastive learning for video representation. *Computer Vision and Image Understanding*, 219:103406, 2022. 6
- [3] De-An Huang, Vignesh Ramanathan, Dhruv Mahajan, Lorenzo Torresani, Manohar Paluri, Li Fei-Fei, and Juan Carlos Niebles. What makes a video a video: Analyzing temporal information in video understanding models and datasets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7366–7375, 2018. 6
- [4] Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2019. 6
- [5] Hyeongmin Lee, Jin-Young Kim, Kyungjune Baek, Jihwan Kim, Hyojun Go, Seongsu Ha, Seokjin Han, Jiho Jang, Raehyuk Jung, Daewoo Kim, et al. Twlv-i: Analysis and insights from holistic evaluation on video foundation models. *arXiv preprint arXiv:2408.11318*, 2024. 6
- [6] Laura Sevilla-Lara, Shengxin Zha, Zhicheng Yan, Vedanuj Goswami, Matt Feiszli, and Lorenzo Torresani. Only time can tell: Discovering temporal data for temporal modeling. In *Winter Conference on Applications of Computer Vision (WACV)*, 2021. 6
- [7] Fida Mohammad Thoker, Hazel Doughty, and Cees GM Snoek. Tubelet-contrastive self-supervision for video-efficient generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13812–13823, 2023. 6
- [8] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35: 10078–10093, 2022. 6

- 126 [9] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao.
127 Videomae v2: Scaling video masked autoencoders with dual masking. In *Proceedings of the IEEE/CVF*
128 *conference on computer vision and pattern recognition*, pages 14549–14560, 2023. 6
- 129 [10] Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Zun
130 Wang, Yansong Shi, et al. Internvideo2: Scaling foundation models for multimodal video understanding.
131 In *European Conference on Computer Vision*, pages 396–416. Springer, 2024. 6
- 132 [11] Yi Wang, Xinhao Li, Ziang Yan, Yinan He, Jiashuo Yu, Xiangyu Zeng, Chenting Wang, Changlian Ma,
133 Haian Huang, Jianfei Gao, et al. Internvideo2. 5: Empowering video mllms with long and rich context
134 modeling. *arXiv preprint arXiv:2501.12386*, 2025. 6
- 135 [12] Xueyang Yu, Xinlei Chen, and Yossi Gandelsman. Learning video representations without natural videos.
136 *arXiv preprint arXiv:2410.24213*, 2024. 6