Supplementary Material for Disentangled Concepts Speak Louder Than Words: Explainable Video Action Recognition

Jongseo Lee[1] **Wooil Lee**[1] **Gyeong-Moon Park**[2†] **Seong Tae Kim**[1†] **Jinwoo Choi**[1†]
[1]Kyung Hee University, Republic of Korea
[2]Korea University, Republic of Korea
{jong980812, lwi2765, st.kim, jinwoochoi}@khu.ac.kr, gm-park@korea.ac.kr

In this supplementary material, we provide additional architecture/implementation/dataset/baseline details, user study setup, additional results, limitations of our method, and broader impact of our method to complement the main paper. We organize the supplementary material as follows:

- Additional details on DANCE (Section 1)
- Implementation details (Section 2)
- Dataset details (Section 3)
- Baseline details (Section 4)
- User study setup and protocol (Section 5)
- Additional results (Section 6)
- Limitations (Section 7)
- Broader impacts (Section 8)

1 Additional Details on DANCE

1.1 Architecture

Video backbone encoder. All three types of disentangled concept layers in DANCE take a video-level feature vector as input to predict their respective concepts. To generate the video-level feature vector $\mathbf{x}_i \in \mathbb{R}^D$, we use a backbone video encoder that takes the input video $V_i \in \mathbb{R}^{T\times C\times H\times W}$ and outputs $\mathbf{x}_i = f(V_i)$. Here, T is the number of frames, C is the number of color channels, and $H \times W$ denotes the spatial resolution of each frame. D is the hidden dimensionality of the extracted video-level feature vector. VideoMAE-B/16 is a transformer-based video encoder built upon the ViT-Base architecture, using a 16×16 spatial patch size and operating on tubelet tokens across time. It includes 12 transformer blocks with a hidden dimension of 768 and 12 attention heads. The model divides the input video into 3D patches (tubelets) and applies space-time attention to learn spatio-temporal representations. After training the backbone on each target dataset, we freeze its parameters and use it solely to extract video-level feature vectors for concept layer training. Unless stated otherwise, we use a pretrained VideoMAE-B/16 [42] model without the classification head as the default video backbone encoder in all experiments. For details on the pretrained weights used, please refer to Section 2.1. Notably, the modular design of the CBM framework allows DANCE to accommodate different backbone encoders without modifying the concept prediction layers. We demonstrate its effectiveness across different video backbone encoders in Table 10.

Concept layer. As described in Section 3 of the main paper, we define three types of concept layers in DANCE: motion dynamics, object, and scene. Each layer is implemented as a bias-free linear layer parameterized by its own weight matrix: $\mathbf{W}_C^m \in \mathbb{R}^{M_m\times D}$ for motion dynamics concepts, $\mathbf{W}_C^o \in \mathbb{R}^{M_o\times D}$ for object concepts, and $\mathbf{W}_C^s \in \mathbb{R}^{M_s\times D}$ for scene concepts, where D is the dimensionality of the

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

video-level feature vector. Following prior CBM-based methods [16, 32], we omit the bias term to enhance interpretability.

Interpretable Classifier To train the final classifier in an interpretable manner, we first obtain concept activations from each concept layer—motion ($\mathbf{z}^m$), object ($\mathbf{z}^o$), and scene ($\mathbf{z}^s$). Then we standardize each activation using its corresponding mean and standard deviation computed over the training set. This normalization step ensures that all concept dimensions are on a comparable scale, preventing bias toward any particular concept type during classifier training. Then we concatenate them to form the combined concept activations $\mathbf{z} = [\mathbf{z}^m; \mathbf{z}^o; \mathbf{z}^s]$. Finally, we the classifier makes the action prediction as follows: $\hat{\mathbf{y}} = h(\mathbf{z}; \mathbf{W}_A)$. To enhance interpretability, we impose sparsity constraints on the classifier weights $h(\cdot; \mathbf{W}_A)$, encouraging the model to rely on a small subset of informative concepts for each prediction.

1.2 Vision-Language Dual Encoder

To supervise language-based concepts and our object/scene concepts without human labeling, we adopt the Label-Free CBM framework [32], which leverages a vision-language dual encoder. Unlike the original Label-Free CBM which operates in the image domain using CLIP [34], we use video-text dual encoders designed for the video domain. Specifically, we experiment with LaViLa [55] and ViCLIP [44], both of which extend CLIP-style contrastive learning to spatio-temporal inputs. We use ViCLIP as the default dual encoder unless stated otherwise. In Table 11, we show the effect of different vision-language dual encoders used to train object and scene concept layers without manual labels.

LaViLa. LaViLa repurposes a Large Language Model (LLM) as a video-conditioned narrator to automatically generate diverse video-language pairs. It then learns a video-language representation via contrastive learning using a video encoder and text encoder. The video encoder is a TimeSformer [2], and the text encoder is initialized from a pretrained LLM.

ViCLIP. ViCLIP extends CLIP to the video domain by replacing the image encoder with a spatiotemporal video transformer (ViT-based). Both the video encoder and text encoder are initialized from CLIP. ViCLIP modifies the native self-attention to incorporate temporal information and applies masked video modeling during pretraining to enhance efficiency. This design allows ViCLIP to maintain CLIP's structure while being tailored for video understanding.

1.3 Concept Discovery

1.3.1 Motion Dynamics Concept

The motion dynamics concept in DANCE is designed to provide temporal explanations for video actions that are disentangled from static cues. In this section, we provide additional details on the construction of motion dynamics concepts that were not fully described in the main paper.

Key frame detection. To extract informative key clips, we first detect key frames from video sequences. One possible approach is to use deep learning-based key frame detection models [38, 41, 29]. However, such models are typically optimized for tasks like action classification or video summarization, where the goal is to select frames that are most discriminative or representative [50, 49], rather than to capture diverse and semantically rich motion dynamics. Moreover, using such models comes with some computational cost, as they often require adaptation to the target dataset

To address this, we employ an off-the-shelf method[1] that detects salient transitions in human movement based on frame-wise pixel differences. This method is a practical implementation inspired by the classic work on automatic video partitioning [53], which proposed detecting transitions based on frame differences. The method first converts video frames to grayscale and applies Gaussian smoothing to reduce noise. It then computes the sum of pixel-wise differences between consecutive frames to construct a temporal signal, and applies peak detection to identify key moments of significant motion change. This simple yet effective algorithm allows us to efficiently partition videos into informative segments without requiring heavy computation, making it well-suited for motion dynamics concept discovery. In Table 3, we empirically validate the effectiveness of the peak detection method in facilitating efficient discovery of motion dynamics concepts.

[1] https://github.com/afreakk/keyframe_detection

Key clip selection. Since the number of detected keyframes varies across videos, we fix the number of key clips per video to S to avoid imbalance in subsequent training. To this end, we uniformly sample S keyframe indices $s \in \{1, \ldots, S\}$ from the detected keyframes. For each selected index, we construct a key clip $\mathbf{V}^s$ by extracting a total of L frames centered around the keyframe. If a keyframe is located near the beginning or end of a video (i.e., within L/2 frames from the boundary), we pad the clip by extending frames from the opposite direction to maintain the clip length. We report the performance variation with respect to different values of S and L in Table 4.

Pose sequence extraction. To extract pose sequences from videos, we use an off-the-shelf ViTPose [48], a transformer-based pose estimator with a lightweight decoder. Given a video $\mathbf{V}$, we apply ViTPose to each frame to extract 17 COCO [23]-format keypoints. Each pose is represented as a matrix $\mathbf{P} \in \mathbb{R}^{17\times 2}$, containing the (x, y) coordinates of body joints. Using the L selected frame indices from each key clip, we obtain a pose sequence $\mathbf{P}^s \in \mathbb{R}^{L\times 17\times 2}$. To ensure consistency across different spatial scales and positions, we normalize each pose in $\mathbf{P}^s$ by centering the joint coordinates and scaling them to the range $[0, 1]$.

Pose sequence filtering. Pose sequences extracted from real-world videos often suffer from two types of quality issues. First, occlusion, truncation, or ambiguous body configurations can lead to low-confidence keypoints. To address this, we discard sequences with low average joint confidence across frames, as provided by the pose estimator. Second, pose estimators may produce temporally inconsistent predictions, especially in crowded scenes or when multiple people perform similar actions. To suppress such tracking failures, we compute the cosine similarity between adjacent pose vectors within a sequence and discard frames whose similarity falls below a predefined threshold. Finally, if all pose sequences $\bigcup_{s=1}^{S} \mathbf{P}^s$ extracted from a video $\mathbf{V}$ are removed by the filtering process, we exclude the $\mathbf{V}$ from concept layer training. This filtering step helps maintain temporal coherence and improves the reliability of motion dynamics representations. We show the effects of these filtering methods in Table 7.

Clustering for concept discovery. To discover motion dynamics concepts, we first collect the pose sequences from the S key clips of each of the N training videos, excluding those removed by filtering. These pose sequences are aggregated into a unified set: $\mathbb{P} = \bigcup_{i=1}^{N} \bigcup_{s=1}^{S} \mathbf{P}_i^s$, where $\mathbf{P}_i^s$ denotes the pose sequence from the s-th key clip of the i-th video. We flatten each pose sequence into a feature vector and cluster the resulting representations using the FINCH [35] algorithm. FINCH produces multiple partitions at different levels of granularity without requiring a predefined number of clusters, allowing us to select the most appropriate concept grouping. We can also specify the number of clusters when needed, which provides a balance between structure discovery and controllability. In Table 6 we show the results of varying the number of motion dynamics concepts.

Representative pose sequence for concept visualization. To visualize each concept, we select the medoid pose sequence within each cluster—that is, the sequence with the smallest average distance to all other members. This representative sequence serves as the canonical example for presenting and interpreting the discovered motion pattern. Each medoid is represented as a pose sequence of skeletal keypoints with shape $L \times 17 \times 2$, where L is the temporal length of the sequence, 17 denotes the number of COCO [23]-format joints, and 2 corresponds to the (x, y) coordinates of each joint. Due to space limitations, we do not display the entire sequence of L frames; instead, we sample a small number of representative frames that best capture the motion dynamics for visualization purposes. In Figure 20–23, showcases a diverse set of motion dynamics concepts discovered by DANCE, highlighting the richness and interpretability of our learned representations.

1.3.2 Spatial Concept

Object and scene concept candidate extraction. As illustrated in Figure 3 (c) of the main paper, we construct spatial concept sets—object and scene concepts—for each action class using large language models, following the approach introduced in LF-CBM [32]. Specifically, we use GPT-4o [13] to generate tangible *object* concepts associated with each action class by prompting with: *"For the action {class name}, list the most important physical objects that commonly appear when this action occurs. Do not include human body parts, poses, or background scenes."* This prompt encourages the model to return concrete, class-relevant object names such as "baseball bat" or "helmet" for actions like *Baseball Swing*. These *object* concepts capture the physical entities that are directly involved in or closely associated with the action, providing strong visual cues.

Similarly, to extract *scene* concepts, we prompt GPT-4o with: *"For the action {class name}, list the most common places or background scenes where this action typically occurs. Do not include objects or equipment."* This prompt is designed to elicit contextual environments—such as "baseball field" or "sports stadium" for the action "playing baseball"—that capture the spatial backdrop in which the action is commonly performed. These scene concepts offer complementary information to object concepts, providing high-level situational cues that support action understanding. We apply this prompt to every class to obtain spatial concept candidates.

Object and scene concept filtering. After obtaining object and scene concept candidates, we apply a series of filtering steps to refine the quality of concepts, following prior work [32]. This filtering process ensures the clarity, diversity, and effectiveness of the resulting object and scene concepts.

1. We discard overly long concepts (over 30 characters) to ensure simplicity and interpretability.
2. We remove concepts that are semantically too similar to the target action classes using text embedding similarity with a threshold of 0.85.
3. We eliminate redundant concepts that are highly similar to other concepts (threshold 0.95). We adopt a relatively high threshold to account for frequent semantic variations in video descriptions—such as "player's bat" and "hitter's bat"—which refer to essentially the same object but appear as distinct textual forms.
4. We remove concepts with low average activations of ViCLIP [44] across the training set, ensuring that only concepts actually represented in the data are retained. We refer to this as the *activation filter*.
5. After optimizing the concept layer, we remove concepts that exhibit low projected concept activations (similarity < 0.45). We refer to this as the *projection filter*.

The remaining set serves as our final spatial concept vocabulary. We empirically evaluate the effectiveness of each filter in Table 12.

Concept pseudo labeling. In this section, we provide the detailed procedure for pseudo labeling of object and scene concepts introduced in Section 3.2.2 of the main paper. For simplicity, we describe the process using the object concept set as an example. We first obtain a filtered object concept set $\mathbf{O} = \{q_1, q_2, \ldots, q_{M_o}\}$ through the steps of concept initialization and filtering. This set consists of M_o textual object concepts. We then encode the concepts using the video-text dual encoder $E_T(\cdot)$, introduced in Section 1.2, which makes text embeddings $E_T(\mathbf{O}) \in \mathbb{R}^{M_o \times D}$, where D is the embedding dimension of the dual encoder. Given an unlabeled video $\mathbf{V}_i$, we obtain its video embedding by passing it through the video encoder $E_V(\cdot)$. Using the similarity formulation defined in Equation (3) of the main paper, we compute similarity scores between the video and all object concepts to generate the pseudo labels.

Similarly, we generate pseudo labels for scene concepts using the same procedure. We start with a filtered scene concept set $\mathbf{S} = \{q_1, q_2, \ldots, q_{M_s}\}$, consisting of M_s textual scene concepts obtained via the concept initialization and filtering steps. Each concept is encoded into a text embedding using the same video-text dual encoder $E_T(\cdot)$, resulting in $E_T(\mathbf{S}) \in \mathbb{R}^{M_s \times D}$. Given the same unlabeled video $\mathbf{V}_i$, we extract its video embedding using $E_V(\cdot)$, and compute similarity scores between the video and all scene concepts based on the similarity formulation in Equation (3) of the main paper. These similarity scores are used as pseudo labels for supervising the scene concept layer.

2 Implementation Details

In this section, we provde ours experimental setup and implementation details. We implement DANCE using PyTorch [33] and build on the official codebases of Label-Free CBM[2] [32] and VideoMAE [42].

2.1 Video Backbone Encoder

In this section, we describe the implementation details of the video backbone encoder used in DANCE. For architectural details, please refer to Section 1.1. We show the dataset-specific hyperparameters used for training VideoMAE-B/16 in Table 1.

[2] https://github.com/Trustworthy-ML-Lab/Label-free-CBM

Table 1: **Hyperparameters used and the fine-tuning configuration for all downstream datasets.**

Config	KTH [36]	Penn Action [54]	HAA-100 [5]	UCF-101 [39]
Optimizer	AdamW [26]	AdamW	AdamW	AdamW
Base learning rate	5e-4	1e-3	5e-3	1e-3
Weight decay	0.05	0.05	0.05	0.05
Optimizer momentum [4]	β_1, β_2=0.9, 0.999	β_1, β_2=0.9, 0.999	β_1, β_2=0.9, 0.999	β_1, β_2=0.9, 0.999
Gpu per batch size	10	10	14	10
Learning rate schedule	cosine decay [25]	cosine decay	cosine decay	cosine decay
Warmup epochs	5	5	5	5
Training epochs	50	50	50	50
Flip augmentation	*yes*	*yes*	*yes*	*yes*
Color jitter	0.4	0.4	0.4	0.4
RandAug [6]	(9, 0.5)	(9, 0.5)	(9, 0.5)	(9, 0.5)
Drop path	0.1	0.1	0.2	0.1
Layer-wise lr decay [1]	0.75	0.75	0.8	0.75

Data preprocessing. Given an input video, we sample 16 frames for model input. Specifically, we use dense sampling [8] for UCF-101 [39] and uniform sampling [43] to KTH [36], Penn Action [54], and HAA-100 [5]. After sampling, we apply random cropping and resize each frame to a spatial resolution of 224×224. We then apply data augmentations including horizontal flipping, color jittering, and RandAugment [6], as detailed in Table 1.

Pretrained weights. We use off-the-shelf pretrained VideoMAE weights from the official repository.[3] Specifically, we use weights pretrained on Something-Something-V2 [9] for KTH, Penn Action, and HAA-100, and weights pretrained on UCF-101 for experiments on UCF-101. Something-Something-V2 is a large-scale and temporally fine-grained dataset, and the self-supervised VideoMAE weights trained on the dataset are particularly effective for datasets where temporal dynamics play a crucial role, such as KTH, Penn Action, and HAA-100.

Training. We initialize the VideoMAE-B/16 model with pretrained weights and perform supervised learning on each target dataset. We conduct all backbone training experiments using 4 NVIDIA GeForce RTX 3090 GPUs. By default, we train the model for 50 epochs with a batch size of 10 per GPU, and use a base learning rate of $1e$-3. We linearly scale the base learning rate with respect to the total batch size using the rule: $lr = \text{base learning rate} \times (\text{batch size})/256$ following VideoMAE [42]. We use the AdamW optimizer [26] with momentum [4] parameters $(\beta_1, \beta_2) = (0.9, 0.999)$, apply cosine learning rate decay [25], set the weight decay to 0.05, and use a layer-wise learning rate decay of 0.75 [1]. We provide the detailed training settings for each dataset in Table 1. After training, we extract and store the video-level feature vectors from the backbone encoder in advance. This strategy enables efficient concept layer training across all experiments by eliminating the overhead of loading video inputs during training.

2.2 DANCE

After training the backbone encoder, we freeze its parameters and pre-extract feature vectors for all training and validation samples. This strategy allows for efficient and reproducible training of the concept layers and the final classifier. We conduct all concept layer and classifier training using a single NVIDIA GeForce RTX 3090 GPU.

Pose estimation. To extract pose sequences from videos, we use ViTPose [48], a transformer-based human pose estimator with a lightweight classification decoder. Specifically, we adopt the ViTPose-L variant pretrained with the masked autoencoding (MAE [12]) strategy. The model is initialized with weights trained on the MS-COCO [23] dataset, and is publicly available from the official repository.[4] We perform pose estimation on all videos from every dataset using an input spatial resolution of 256×192.

Motion dynamics concept layer training. We train the motion dynamics concept layer using video-level feature vectors as input to produce motion dynamics concept activations. By default, we use the Adam [15] optimizer with a learning rate of 1e-3 and a batch size of 512. Although we

[3] https://github.com/MCG-NJU/VideoMAE

[4] https://github.com/ViTAE-Transformer/ViTPose

set the training step limit to 10K steps, we apply early stopping based on validation performance. Specifically, when training motion dynamics concepts, we exclude filtered-out training samples and use only the remaining ones, as described in Section 1.3.1. After training the concept layer, we compute and store the mean and standard deviation of the training set to standardize the concept activations during classifier training.

Object and scene concept filtering. We apply two filtering stages to refine the discovered object and scene concepts: *activation filtering* and *projection filtering* described in Section 1.3.2. The activation filter computes the average similarity of each concept's pseudo label (obtained via the video–text multimodal encoder) across the training set. We use a default threshold of 0.22 for both object and scene concepts. The projection filter is applied after training the concept layer, based on the average activation of each projected concept. We use a default threshold of 0.45 to remove underutilized or non-informative concepts.

Object and scene concept layer training. For training the object and scene concept layers, we use the same optimizer, batch size, and early stopping criteria as in the motion dynamics concept layer. The main difference lies in the loss function: while motion dynamics concepts are trained using binary cross-entropy (BCE) loss, object and scene concepts are trained using the *cosine cubed loss* proposed in Label-Free CBM [32]. This loss measures the cosine similarity between the predicted and target vectors after raising both to the third power, encouraging sharper alignment. For more details, please refer to the original Label-Free CBM paper.

Concept Configuration. We describe the default concept configurations used across all datasets in our experiments. For motion dynamics concepts, we use $L = 20$ for KTH and HAA-100, and $L = 25$ for Penn Action and UCF-101. The number of key clips per video (S) is set to 17 for KTH, 12 for Penn Action, 7 for HAA-100, and 10 for UCF-101. The number of discovered motion concepts varies by dataset: 71 for KTH, 79 for Penn Action, 223 for HAA-100, and 500 for UCF-101. In Table 6, we show how we determine the number of motion dynamics concepts for each dataset. We use the number of object and scene concepts obtained through the concept set filtering procedure described in Section 1.3.2. For object concepts, we retain 44 out of 59 for Penn Action, 23 out of 24 for KTH, 165 out of 218 for HAA-100, and 172 out of 481 for UCF-101. For scene concepts, we retain 41 out of 79 for Penn Action, 55 out of 56 for KTH, 208 out of 279 for HAA-100, and 110 out of 418 for UCF-101. Unless otherwise noted, we use these filtered concept sets as the default configuration in all experiments, including the results reported in Table 1 of the main paper.

Interpretable classifier training. We employ the GLM-SAGA solver [46] to optimize the sparse classifier in the Equation (5) of the main paper. We use $\alpha = 0.99$, $\lambda = 0.05$, a step size of 5K, and a batch size of 512. Early stopping is applied based on validation loss to avoid overfitting.

Inference. During inference, we resize each frame such that the shorter side is 224 pixels and apply a center crop of 224×224. While many video action recognition methods [19, 42, 51] report performance using ensemble predictions over multiple spatial crops and temporal clips, our focus is not on maximizing accuracy. For fairness and simplicity, we do not apply any test-time augmentation and report all results using a single-view setting (1 clip × 1 crop)

3 Dataset Details

3.1 In-domain Setting.

We evaluate DANCE on four datasets: KTH [36], Penn Action [54], HAA-100, and UCF-101 [39]. KTH, Penn Action, and HAA-100 datasets primarily require a model to reason over temporal motion patterns rather than static spatial context. In contrast, UCF-101 includes many classes where spatial cues (e.g., scene or object appearance) are sufficient for classification, and motion dynamics may play a less critical role. This selection enables us to evaluate DANCE under both temporally demanding and spatially biased conditions, demonstrating its effectiveness across diverse video understanding scenarios.

KTH The KTH Action dataset [36] consists of 6 action classes performed by 25 subjects in 4 different scenarios. It includes 1,528 training videos and 863 test videos. It consists of six human action classes: walking, jogging, running, boxing, hand waving, and hand clapping. All videos are

grayscale and have a resolution of 160×120 pixels at 25 FPS with minimal background clutter. This makes the dataset particularly suitable for evaluating models' ability to recognize motion-centric actions from low-level visual signals.

Penn Action. Penn Action [54] is a large-scale dataset designed for video-based human action recognition. It consists of 2,326 video sequences across 15 action categories (e.g., baseball swing, bench press), capturing a wide range of articulated human motions. Each frame is annotated with 2D human pose keypoints for 13 joints, including both body and limb parts.

Following the official split, we use 1,256 videos for training and 1,067 for testing.

1. arm_wave
2. atlatl_throw
3. axe_throwing
4. backflip
5. badminton_overswing
6. badminton_serve
7. badminton_underswing
8. baseball_bunt
9. baseball_catch_groundball
10. baseball_pitch
11. baseball_swing
12. basketball_dribble
13. basketball_dunk
14. basketball_hookshot
15. basketball_jabstep
16. basketball_layup
17. basketball_pass
18. basketball_shoot
19. battle-rope_jumping-jack
20. battle-rope_power-slam
21. battle-rope_rainbow
22. bench_dip
23. bowing_fullbody
24. bowling
25. bowls_throw
26. burpee
27. canoeing_sprint
28. cast_net
29. climb_ladder
30. CPR
31. dips
32. floss_dance
33. football_throw
34. frisbee_throw
35. gangnam_style_dance
36. golf_part`
37. golf_swing
38. gym_lift
39. gym_lunges
40. gym_plank
41. gym_pull
42. gym_push
43. gym_ride
44. gym_run
45. gym_squat
46. hammer_throw
47. handstand
48. high_knees
49. hurdle_jump
50. javelin_run
51. javelin_throw
52. jump rope
53. jumping_jack
54. leg_hold_back
55. leg_hold_front
56. leg_split
57. moonwalk
58. neck_side_pull_stretch
59. one_arm_push_up
60. pirouette
61. pull_ups
62. punching_sandbag
63. pushup
64. quadruped_hip-extension
65. roller-skating_forward
66. rowing_boat
67. salute
68. shuffle_dance
69. side_lunge
70. situp
71. soccer_throw
72. soccer_shoot
73. split_leap
74. sprint_start
75. taekwondo_low_block
76. taekwondo_middle_block
77. tennis_backhand
78. tennis_forehand
79. tennis_serve
80. volleyball_underhand
81. volleyball_set
82. volleyball_pass
83. volleyball_overhand
84. weightlifting_stand
85. weightlifting_overhead
86. weightlifting_hang
87. workout_chest-pull
88. workout_crunch
89. yoga_bridge
90. yoga_cat
91. yoga_dancer
92. yoga_firefly
93. yoga_fish
94. yoga_gate
95. yoga_locust
96. yoga_lotus
97. yoga_pigeon
98. yoga_tree
99. yoga_triangle
100. yoga_updog

Figure 1: **Full list of action classes in HAA-100.**

HAA-100. The HAA-500 dataset [5] provides 500 fine-grained action classes involving human-object and human-human interactions, with only 20 video clips per class (16 for training, 4 for testing). This limited data per class makes it difficult to train models that require diverse spatio-temporal patterns or to evaluate concept-based explanations reliably. In addition, many classes exhibit strong background or object bias, which can confound the interpretability analysis. To enable more interpretable and controlled evaluation, we construct the HAA-100 subset by selecting 100 classes that exhibit less bias, clearer temporal pose dynamics, and overlap semantically with classes from other datasets. This curated subset allows for more robust and interpretable evaluation of concept-based video understanding models. The full list of classes included in HAA-100 is shown in Figure 1.

UCF-101. The UCF-101 [39] dataset is a large-scale action recognition dataset consisting of 13,320 video clips spanning 101 human action classes. Each class contains at least 100 video clips with varying camera motion, background complexity, and lighting conditions, making the dataset highly challenging. The official split provides three training/testing partitions and we report results using the first split. In addition, we use UCF-101-Attributes [14], a set of binary attribute annotations for each action class released as part of the TUMOS Challenge[5]. These attributes cover semantic cues such as involved body parts, motion types, and interaction contexts, and are manually defined by domain experts.

3.2 Cross-domain Setting.

We include UCF-101-SCUBA [21] and Mimetics [45] to evaluate DANCE under severe domain shifts. DANCE enables transparent and post hoc intervention—empowering users to inspect, debug, and recover model behavior by reasoning about concept-level decisions. This setup allows us to probe how well our disentangled explanations generalize beyond the training domain and support actionable feedback even under distribution shifts. Specifically, we replace test videos of selected classes in UCF-101 with their SCUBA or Mimetics variants to simulate domain shift scenario [52] at the class level, enabling fine-grained evaluation of model robustness and intervention capability. In Figure 2, we show example videos from each dataset.

UCF-101-SCUBA SCUBA [21] is a synthetic dataset designed to evaluate model robustness under spatial distribution shifts. It is constructed by overlaying action regions from source videos onto diverse background scenes drawn from three sources: (i) validation images from Places365 [56], (ii) 2,000 images generated by VQGAN-CLIP conditioned on random Places365 categories and artistic styles, and (iii) synthetic stripe patterns generated via sinusoidal functions. In this work, we evaluate

[5] http://www.thumos.info/download.html

(a) An example video of UCF-101-SCUBA *Tennis Swing* class

(b) An example video of Mimetics *Archery* class

Figure 2: **Example videos from cross-domain datasets.** (a) An example video from the *Tennis Swing* class in UCF-101-SCUBA, where the original background is replaced with a scene image from Places365 to simulate domain shift. (b) An example from the *Archery* class in Mimetics, where the action is performed through miming, without the use of actual objects, representing a shift in visual modality.

on the Places365 variant of UCF-101-SCUBA, which pairs 910 videos from the first official split of the UCF-101 validation set with five randomly sampled Places365 backgrounds per video, yielding 4,550 synthetic test videos. This setup allows for controlled assessment of explanation robustness by inducing domain shift through background alterations, while keeping motion dynamics unchanged.

Mimetics Mimetics [45] contains 713 short video clips spanning 50 human action classes selected from the Kinetics dataset. Each clip shows mimed actions—such as brushing teeth, juggling, or playing instruments—performed without real object interaction in staged or casual environments. Since it excludes physical context, the dataset introduces a strong domain shift. We use Mimetics only for testing and leverage it to evaluate generalization under object-deprived, out-of-distribution conditions.

4 Baseline Details

4.1 Saliency-based explanations

VTCD. We include VTCD [18] as a representative post-hoc concept discovery method. VTCD discovers spatio-temporal tubelet concepts through intra-class clustering. To obtain a VTCD explanation for the selected class, we manually chose the most reasonable explanation from the discovered concepts in the last transformer layer of the VideoMAE-B/16 [42] encoder used in DANCE. Since VTCD provides class-level explanations, we compare it to the model-level concept-to-class weights of DANCE for a fair evaluation. We use the same backbone encoder (VideoMAE-B/16) for both VTCD and DANCE to ensure architectural consistency. We use this selected explanation in both Figure 1 and the user study presented in the main paper.

Saliency Tubes. We use Saliency Tubes [40], a video-level spatio-temporal saliency explanation method based on gradient backpropagation. The saliency outputs highlight the most influential regions across space and time for each prediction. As visualized in Figure 1 and Figure 4 of the main paper, we train Saliency Tubes directly on the downstream action recognition datasets. We use the

GradCAM. We also include GradCAM [37], a widely used saliency-based explanation method originally designed for image models. We apply GradCAM [37] to a ResNet-152 [11] model pretrained on ImageNet-1K [7] to compare sample-level explanations in Figure 4 of the main paper. We generate saliency maps for each frame independently and concatenate them along the temporal axis for visualization.

4.2 Language-based Explanations

GPT-4o-derived concepts. To compare with language-based explanations, we query GPT-4o with purpose-specific prompts to generate concept sets tailored to video action recognition. For details on how we obtain and filter GPT-4o-derived concepts using prompt-based queries, please refer to Section 1.3.2. As baselines, we use two types of language-derived concepts: *temporal descriptions*

and *spatio-temporal descriptions*. For the *GPT-4o-derived temporal description concepts*, we use the following prompt: *"For the action {class name}, describe how the action typically unfolds over time using short, simple sentences. Do not include objects, background scenes, or human body parts."* For the *GPT-4o-derived spatio-temporal description concepts*, we use the prompt: *"For the action {class name}, write several short descriptive sentences that capture the important visual concepts involved. Include objects, scenes, and key actions."* We apply the same concept filtering procedure following the same process used for the object and scene concepts in DANCE. Note that, we do not apply the *discard overly long concepts* step, as the language-derived descriptions are inherently longer and structured as complete phrases. Note that the *GPT-4o-derived object and scene concepts* are those used in DANCE, while the temporal and spatio-temporal descriptions are used only for comparison in language-based baselines.

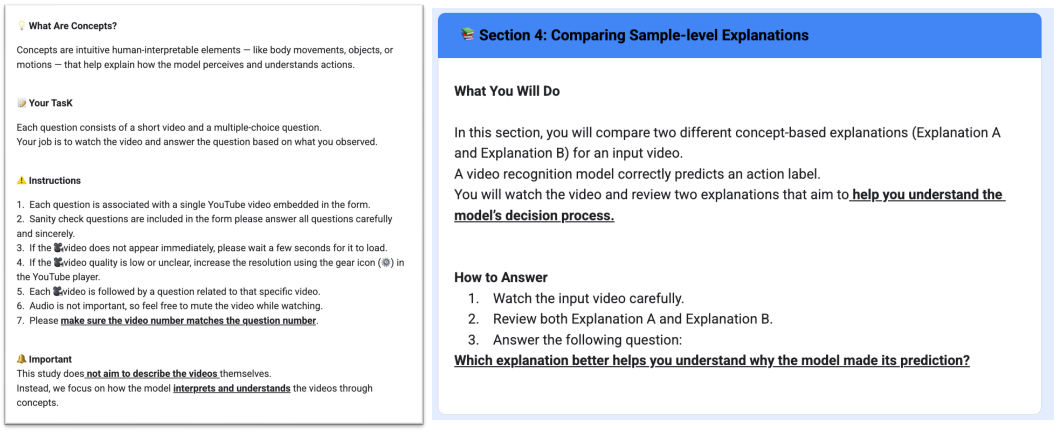
Training. We use the same experimental setup as described for the object and scene concept layers in Section 2. The only difference lies in the initialized concept texts. We use a pretrained VideoMAE-B/16 [42] as the video backbone encoder for processing video inputs, and a video–text multimodal encoder (ViCLIP [44]) to obtain pseudo labels for supervision via the Label-Free CBM approach [32]. Since all other components are identical across experiments, we refer to each model by the name of its concept set for simplicity and clarity.

4.3 Expert knowledge.

UCF-101-attributes. UCF-101-attributes [14] provides semantic attribute annotations that define relationships between 101 action classes and a set of binary attributes. Each class is associated with one or more attributes, offering high-level semantic information despite the absence of video-level annotations. Since these attributes are manually defined by domain experts, we treat this concept set as *expert knowledge*. To ensure a fair comparison of spatial concepts in Table 5, we use only the 26 object- and scene-related attributes (e.g., *Object = Ball*) from the full set.

Training We train a Concept Bottleneck Model (CBM) [16] using these attributes as binary label vectors for each class, following the standard CBM formulation.

5 User Study Setup and Protocol



(a) Task overview instruction interface

(b) Section-specific instruction interface

Figure 3: **Instruction interfaces shown to participants.** (a) We provide overview instruction at the beginning of the study. (b) We provide section-specific instructions before each task to guide participants throughout the study.

To assess the interpretability of the proposed DANCE, we conducted a user study to evaluate the interpretability of our motion dynamics concepts and the effectiveness of DANCE explanations. The results of this study are summarized in Figure 6 of the main paper, and we provide further details in this section. Specifically, our study aimed to answer two key questions:

- Do pose sequence-based representations effectively convey the temporal context of human actions in the video?
- Does our proposed DANCE explanation better help users understand the model's decision compared to other explanation methods?

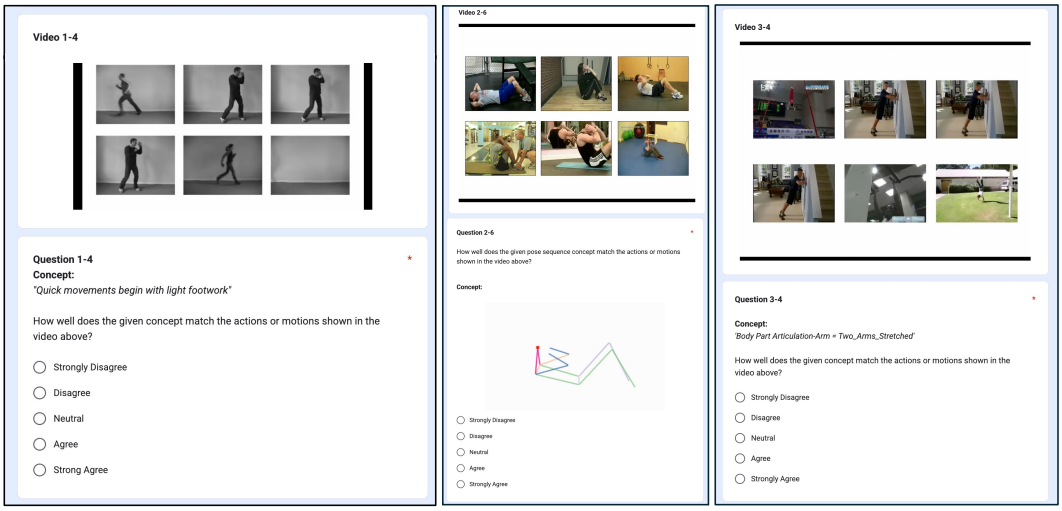
5.1 Setup

To address the two key questions, we conducted a user study using the Prolific platform. The study consisted of a series of simple and intuitive multiple-choice questions designed to evaluate users' understanding of motion-based explanations. We provided clear instructions to participants prior to the task, as illustrated in Figure 3. This study was reviewed and approved as exempt by the Institutional Review Board (IRB) at our institution.

Participants and Compensation. We collected responses from 50 adult participants from English-speaking regions, regardless of gender. Each participant answered 40 questions, which included embedded two sanity checks to ensure engagement and attention. Participants were compensated \$0.1875 per question, totaling \$7.50 for the entire study. On average, participants completed the study in about 24 minutes, corresponding to an hourly rate of approximately \$18.75. We did not collect any personal information from the participants, nor did we disclose any details about the model or research objectives during the study.

Baselines. We compared our method against three representative types of concept-based explanation methods: (1) saliency-based concepts, using VTCD [18] for post-hoc spatio-temporal concept discovery; (2) language-based concepts, where concepts generated by GPT-4o [13] were used to train a label-free CBM [32]; and (3) expert-knowledge-based concepts, using predefined fine-grained attributes [14] from UCF-101 [39] to train a standard CBM [16]. For a fair comparison, we used the same backbone encoder—VideoMAE-B/16 [42] fine-tuned on each dataset—for all methods, including DANCE.

Datasets used. We conducted the user study using a diverse set of action recognition datasets, including KTH [36], Penn Action [54], HAA-100 [5], and UCF-101 [39]. These datasets cover a wide range of human actions with varying levels of motion complexity, enabling us to evaluate the generalizability and robustness of each explanation method. For more detailed descriptions of each dataset, please refer to Section 3.



(a) Language-based (b) Pose sequence (c) Expert-knowledge-based

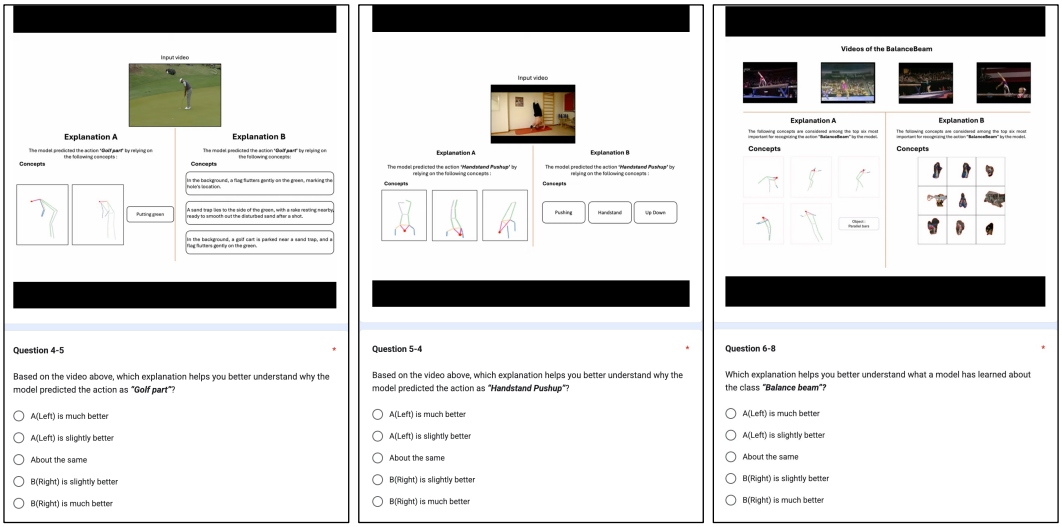
Figure 4: **Example question and user interface used in Task 1.** We show how participants evaluate the interpretability of a single concept using several highly activated video clips. Each interface corresponds to one of the three concept types.

5.2 Task1: Do pose sequences effectively convey the temporal context of human actions?

Task 1 aims to address the key question: *Do pose sequence-based representations effectively convey the temporal context of human actions in video?* In other words, this task evaluates the interpretability of the proposed motion dynamics concepts.

Experimental setting. To evaluate this, we perform an absolute evaluation comparing three types of concepts: (i) *Language-based* refers to a Label-Free CBM [32] trained on temporal concepts generated by GPT-4o [32], (ii) *Pose sequence* refers to DANCE trained solely on the our proposed motion dynamics concepts, and (iii) *Expert-knowledge-based* refers to a CBM [16] trained on temporal concepts derived from UCF-101-attributes [14], excluding concepts related to static cues (e.g., *Object: Ball Like*). This task is conducted exclusively on the UCF-101 dataset, as it is the only dataset that provides such expert annotations. For each method, we trained a Concept Bottleneck Model (CBM) [16] using the corresponding set of concepts. For the language-based method, we adopted a Label-free CBM [32] with video-text dual encoder [44].

User evaluation protocol. For each model, we randomly select one concept and visualize the top six video clips whose top-1 concept activation corresponds to the selected concept. We then presented the concept alongside the activated video clips, and asked participants the following question: *"How well does the given concept match the actions or motions shown in the video above?"* Participants rated six such concept-video sets per model using a five-point Likert scale (1: Strongly Disagree to 5: Strongly Agree). We independently evaluated each concept set, without directly comparing it to others. Each participant answered 6 questions per concept set, resulting in a total of 18 questions. We show an example of the evaluation question and interface in Figure 4.



(a) DANCE vs Language-based (b) DANCE vs Expert-knowledge-based (c) DANCE vs VTCD

Figure 5: **Example question and user interface used in Task 2.** Participants are asked to compare two explanations (A and B) for each sample or class. (a) Language-based [13], (b) Expert-knowledge-based [39], and (c) VTCD [18]. In (a) and (b), we show a single video sample with top-3 sample-level concept contributions. In (c), VTCD provide a class-level explanation with nine representative video clips.

5.3 Task2: Does DANCE improve users' understanding of model compared to other explanations?

Task 2 aims to address the second key question: *Does our proposed DANCE explanation better help users understand the model's decision compared to other explanation methods?* To evaluate the interpretability of DANCE, we compare its explanations with those of three baseline methods: language-based, expert-knowledge-based, and saliency-based approaches.

Experimental setting. To evaluate this, we conduct a relative evaluation comparing DANCE with three types of explanations: (i) *Language-based* refers to a Label-free CBM [32] trained on entangled spatio-temporal concepts generated by GPT-4o [13] , (ii) VTCD [18], a saliency-based spatio-temporal concept discovery method, and (iii) *Expert-knowledge-based*, refers to a CBM [16] trained on spatio-temporal concepts derived from UCF-101-attributes [14]. Since VTCD provides only class-level explanations through intra-class clustering, we use the class-level explanation of DANCE for a fair comparison. Specifically, we use the concept-to-class weights of DANCE as its class-level explanation as shown in the Figure 5 of the main paper. In contrast, for the other two baselines, we used the top-3 contributing concepts at the sample level as their explanations.

User evaluation protocol For the comparison with VTCD [18], we selected a single action class for evaluation. To obtain a VTCD explanation for the selected class, we manually chose the most reasonable explanation from the discovered concepts in the last transformer layer of the VideoMAE-B/16 [42] encoder used in DANCE. We used the default hyperparameters provided in the official VTCD implementation [6], originally optimized for Something-Something-V2 [9]. Since VTCD presents nine representative videos per explanation, we also selected the top-9 weighted concepts from DANCE as its class-level explanation to ensure a fair comparison. For the other two baselines (*Language-based* and *Expert-knowledge-based*), we randomly selected a single video sample and presented the top-3 sample-level concept contributions as the explanation. Participants were then asked to compare our explanation with the baseline for that sample.

For each question, we presented either four randomly selected videos from a given class (for VTCD) or a single input sample (for the other baselines), along with two explanations labeled A and B. Participants were asked: *"Which explanation better helps you understand why the model made its prediction?"* All questions in Task 2 were formulated as pairwise comparisons using a five-point scale ranging from "A is much better" to "B is much better," including a neutral "About the same" option. We show an example of the evaluation question and interface in Figure 5. In total, 50 participants completed Task 2. Each participant answered 8 questions comparing DANCE with VTCD and 6 questions each for the other two baselines, resulting in 20 questions per participant.

6 Additional Results

6.1 Ablation Study

For fair comparison, all experiments in this section use the same video backbone encoder, VideoMAE-B/16 [42]. Unless otherwise specified, all ablation studies are conducted on the UCF-101 [39] dataset. By default, we use FINCH [35] for motion dynamics concept clustering with 500 clusters, and obtain object and scene concepts via GPT-4o [13]. For more implementation details and hyperparameters, please refer to the Section 2.

6.1.1 Concept Discovery

Table 2: **Effect of clustering method on motion dynamics concept discovery.**

Method	Acc.
FINCH [35]	87.5
Self-Organizing Map [17]	86.9
K-Means [28]	87.5

Table 3: **Comparison of keyframe detection methods.**

Method	Acc.
Random selection	86.9
Keyframe detector [38]	87.0
Peak detection [53]	87.5

Effect of clustering method on motion dynamics concept discovery. To validate the effect of clustering strategy on concept quality, we compare different clustering algorithms for the discovery of motion dynamics concepts of DANCE in Table 2. We evaluate FINCH [35], SOM(Self-Organizing Map) [17], and K-Means [28], each applied to the same set of pose sequences extracted from training videos. To ensure a fair comparison, we fix the number of motion dynamics concepts to 500 across all methods and use the same object and scene concept sets in every configuration For the SOM method, we use 529 concepts instead, as it requires the number of clusters to be a perfect square. All methods

[6] https://github.com/YorkUCVIL/VTCD

yield high and comparable accuracy, ranging from 86.9% to 87.5%, indicating that DANCE is robust to the choice of clustering strategy. Among these, we select FINCH as the default clustering method due to its ability to automatically infer the number of clusters from the data hierarchy—eliminating the need for manual hyperparameter tuning. These results confirm that the concept discovery process in DANCE is not only effective but also flexible across unsupervised clustering strategies.

Table 4: **Effects of pose sequence length (L) and number (S).**

$S \backslash L$	10	25	50
5	87.4	86.4	86.8
10	87.0	87.5	87.5
20	87.4	86.6	87.2

Comparison of keyframe detection methods. To discover motion dynamics concepts, we extract key clips from selected keyframes and obtain pose sequences from them. In Table 3, we compare three keyframe selection strategies: Random selection, a deep learning-based method (CSTA [38]), and our peak detection approach [53]. CSTA selects keyframes by applying 2D convolutions over stacked frame features to capture spatiotemporal saliency, identifying frames with high visual importance. The *Peak detection* method detects keyframes by treating frame-wise differences as a signal and identifying peaks. Details are provided in Section 1.3.1. The *Random selection* strategy selects frames arbitrarily. For fair comparison, all methods are configured to select 10 keyframes per video. Among these, peak detection method yields the highest accuracy (87.5%), outperforming random selection (86.5%) and the deep learning-based method CSTA (87.0%). While CSTA achieves competitive performance, it requires significantly higher computational cost due to CNN-based spatiotemporal modeling, whereas the peak detection method is lightweight and efficient, relying only on simple frame-wise differences. These results show that the peak detection method provides efficient motion cues for discovering motion dynamics concepts without relying on external models.

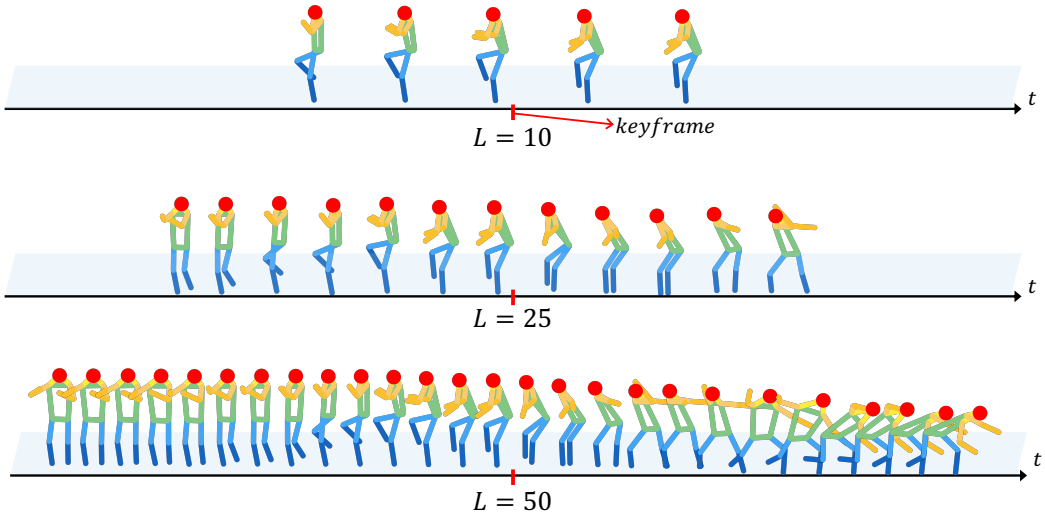


Figure 6: **Effect of pose sequence length L on motion dynamics concept interpretability.** We visualize the motion dynamics concept discovered from a *BaseballPitch* video, using different sequence lengths (L=10, 25, 50). To enhance clarity, we visualize one out of every two frames from each pose sequence. Red tick marks denote the keyframes around which each motion concept is constructed. Short sequences (L=10) often fail to capture meaningful motion and tend to appear static — for example, only showing the pitcher lifting a knee without subsequent movement. Long sequences (L=50) span the full action and often encode class-specific patterns, such as the entire pitching motion, which may hinder interpretability by overfitting to class labels. In contrast, intermediate sequences (L=25) tend to capture representative motion fragments that are both discriminative and generalizable, which allows us to use *primitive* pose sequences for better interpretability.

Effect of pose sequence length L and number S. In Table 4, we analyze the effect of pose sequence length L and the number of key clips per video S on classification accuracy and interpretability. When $L = 10$, the model achieves 87.0–87.4% accuracy depending on S, but the short sequence length limits the temporal context available to the model, making it difficult to capture meaningful motion dynamics. In contrast, when $L = 50$, the model achieves the best accuracy of 87.5%, but the resulting concepts tend to represent the entire action class itself, rather than reusable, interpretable motion units shared across classes. For example, it is like classifying an image of an "apple" using the concept *apple*—correct, but not informative. Therefore, we set the sequence length to 25 to use *primitive* pose sequences for better interpretability. In Figure 6, we visualize how different values of L affect the interpretability of the discovered motion dynamics concepts. Additionally, increasing S, the number of key clips per video, improves coverage but significantly increases the computational cost of clustering. Considering both performance and interpretability, we adopt $L = 25$ and $S = 10$ as a balanced setting. This configuration achieves the highest accuracy (87.5%) and we use this configuration as the default in all experiments.

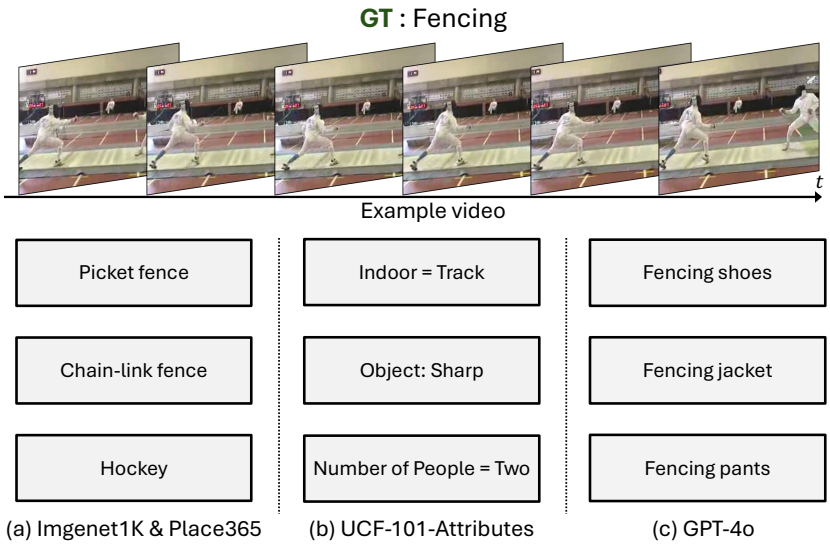


Figure 7: **Comparison of object and scene text concepts from different sources.** We compare the top-3 object and scene concepts (based on concept-to-class weights) used for the *Fencing* class across three different concept generation methods.

Table 5: **Effect of different object and scene concepts.**

Method	# Concepts	Acc.
GPT-4o [13]	899	84.6
LLaVa [24]	648	82.9
IN1K [7] & Place365 [56]	1,365	83.7
WordNet [30] noun	67,893	84.2
UCF-101-attritbutes [39, 14]	26	81.8

Effect of different object and scene concepts. To validate the object and scene concepts used in DANCE, we compare the GPT-4o-derived [13] object and scene concept set with other alternative concept sets in Table 5. For the UCF-101-attributes [39, 14] baseline, we train a standard CBM [16] using the available binary concept labels. The *# Concepts* column indicates the total number of initialized object and scene concepts before filtering. For a fair comparison, we apply filtering to ensure that the final number of concepts remains similar across all settings. Notably, UCF-101-attributes contains only 26 object- and scene-related concepts, limiting its expressiveness compared to other sources. For the ImageNet-1K [7] and Places365 [56] baselines, we directly use the class names of each dataset as object and scene concepts, respectively. Finally, for the WordNet [30] noun baseline, we extract a broad set of general-purpose concepts by collecting common noun entries from the WordNet hierarchy. WordNet nouns yield reasonable performance, but the large number of concepts makes interpretation challenging. The UCF-101-attributes performs poorly due to its

limited set of only 26 object and scene concepts. In contrast, the GPT-4o-derived [13] concept set achieves the highest accuracy (84.6%) with a relatively small number of concepts, demonstrating an effective balance between performance and interpretability. These results motivate our choice to use GPT-4o-derived concepts for object and scene supervision in DANCE, balancing accuracy and interpretability.

To qualitatively compare concept quality, Figure 7 visualizes the object and scene concepts generated by each method for the same video. Concepts from ImageNet-1K & Place365, and UCF-101-attributes often include generic or unrelated items (e.g., "Picket fence", "Hockey", "Sharp") that fail to reflect the specific action context. In contrast, GPT-4o-derived obejct and scene concepts produces more contextually appropriate concepts such as "Fencing shoes", "A fencing jacket", and "Fencing pants," which better reflect the scene. Based on these observations, we adopt GPT-4o as the default source of object and scene concepts in all experiments.

Table 6: **Effect of the number of motion dynamics concepts.**

FINCH Hierarchy	KTH		Penn Action		HAA-100		UCF-101	
	# Concepts	Acc.	# Concepts	Acc.	# Concepts	Acc.	# Concepts	Acc.
Partition 1	1712	91.9	1887	98.1	1,208	71.8	6,159	87.9
Partition 2	71	91.1	79	98.1	223	70.7	500	87.5
Partition 3	21	90.2	19	97.9	42	68.5	154	87.0

Effect of the number of motion dynamics concepts. To assess the impact of the number of motion dynamics concepts on performance, we report the accuracy under different concept counts in Table 6. We conduct experiments on four datasets —KTH, Penn Action, HAA-100, and UCF-101— using the number of motion dynamics concept obtained from partition 1,2, and 3 of the FINCH [35] clustering results. The *# Concepts* column indicates the number of motion dynamics concepts. Overall, we observe that increasing the number of concepts generally improves accuracy. However, overly large concept sets lead to a significant increase in training and inference costs. To balance performance and efficiency, we use the number of motion dynamics concept obtained from partition 2 for all datasets.

Table 7: **Effect of pose sequence filtering.**

Method		Acc.
Cosine similarity	Joint confidence	
✓	✓	87.5
✗	✗	86.7

Filtering unreliable poses improves concept quality. To validate the effectiveness of the pose sequence filtering used in DANCE, we compare the classification performance with and without the filtering process. The filtering strategy follows the method described in Section 1.3.1. As shown in Table 7, the model achieves 86.7% accuracy without any filtering. In contrast, applying a cosine similarity threshold of 0.92 and a confidence threshold of 0.5 improves the accuracy to 87.5%. In Figure 8, we illustrate the effect of the filtering process by visualizing pose sequences before and after filtering, highlighting how noisy poses are removed to produce smoother and more consistent motion representations. This filtering process allows us to obtain cleaner pose sequences, which can be more effectively clustered to discover consistent and interpretable motion dynamics concepts.

Comparison of DANCE vs Non-CBM. To isolate the source of DANCE's performance, we conduct an architecture-controlled experiment on UCF-101 in the Table 8. We modify VideoMAE-B/16 [42] to match DANCE's architectural components by adding a non-interpretable dump layer, which is structurally identical to our concept layer, along with a sparse linear classifier. The standard VideoMAE-B/16 achieves 88.7% Top-1 accuracy. Adding only the sparse linear layer reduces accuracy to 86.8%. Using the dump layer alone yields 87.1%, and combining both further drops performance to 86.5%. All configurations perform worse than DANCE. These results indicate that the performance gains of DANCE are not due to architectural changes alone, but rather emerge from its interpretable, concept-based design. We also observe that applying sparsity improves DANCE from

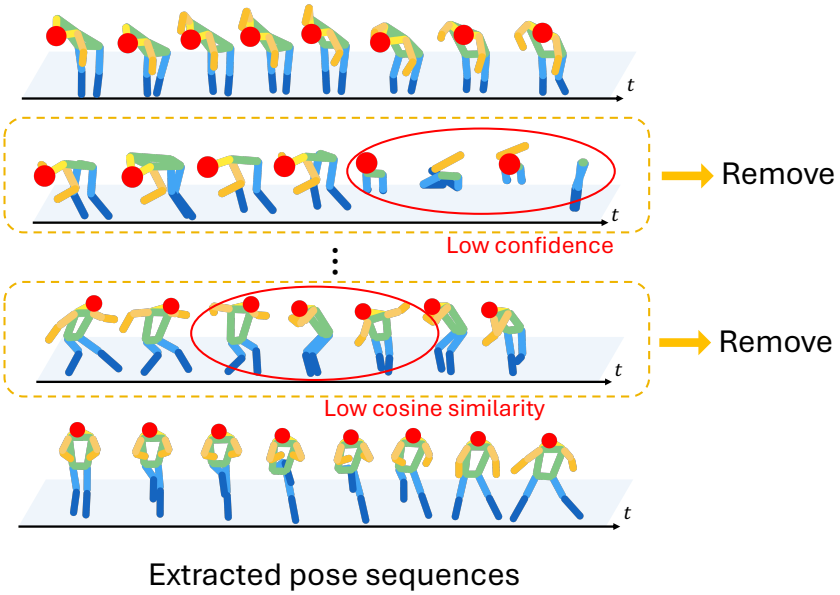


Figure 8: **Effect of pose sequence filtering.** Extracted pose sequences may contain some noisy poses caused by pose estimation failures (low confidence scores) and tracking errors (low inter-frame cosine similarity). To address this, we apply a filtering step that removes unreliable poses based on average joint confidence scores and average inter-frame similarity.

Table 8: **Effects of additional layer and sparsity in baseline and DANCE architectures.**

Method	Add. layer	Sparsity	Acc.
Baseline	✗	✗	88.7
	✗	✓	86.8
	✓	✗	87.1
	✓	✓	86.5
DANCE	✓	✗	86.9
	✓	✓	87.5

86.9% to 87.5%, as it encourages structured reasoning over semantically meaningful concepts. In contrast, sparsity consistently degrades the performance of the non-interpretable baseline, suggesting that sparsity serves as an effective inductive bias when guided by a concept bottleneck structure.

Concept combination. To evaluate the contribution of different concept types by testing various combinations. In Table 9, using only motion dynamics concepts yields 86.4% accuracy, confirming that temporal information alone provides strong predictive signals. Adding object or scene concepts improves performance, with (motion dynamics & object) achieving 86.8% and (motion dynamics & scene) achieving 86.5%. The slightly higher gain from object concepts suggests that they offer more action-relevant cues than static scene context. In contrast, using only object and scene concepts without motion dynamics results in a notable performance drop, highlighting the limitations of relying solely on static context to represent dynamic actions. Using all three concept types yields the best performance, demonstrating the complementary nature of motion and context cues. These findings validate the importance of disentangled concept representations for both interpretability and recognition performance.

Effect of video backbone encoder. In Table 10, we evaluate the impact of different video backbone encoders while keeping the video-text multimodal dual encoder fixed to ViCLIP [44]. Regardless of the encoder used, the model consistently maintains strong performance. These results demonstrate that DANCE is robust to the choice of video backbone encoder and generalizes well across diverse visual representations, reinforcing its modularity and broad applicability.

Table 9: **Effects of individual concept types on DANCE performance.**

Concept			Acc.
Motion dynamics	Object	Scene	
✓	✗	✗	86.4
✓	✓	✗	86.7
✓	✗	✓	86.5
✗	✓	✓	84.6
✓	✓	✓	87.5

Table 10: **Evaluation of different video backbone encoder.**

Video backbone encoder	Acc.
VideoMAE [42]	87.5
CLIP [34]	70.2
R3d [10]	81.9

Table 11: **Evaluation of different vision-language dual encoder.**

Vision-language dual encoder	Acc.
ViCLIP [44]	87.5
CLIP [34]	86.8
LAVILA [55]	87.4

Effect of video-text multimodal dual encoder. In Table 11, we assess the effect of vision-language dual encoders while fixing the video backbone encoder to VideoMAE [42]. We compare ViCLIP [44], CLIP [34], and LAVILA [55], and observe that all configurations yield comparable performance. These results confirm that DANCE is robust to the choice of vision-language dual encoder and can integrate seamlessly with various multimodal representations.

Table 12: **Effects of activation and projection filtering.**

Method		Acc.
Activation filter	Projection filter	
✓	✓	87.5
✗	✓	87.2
✓	✗	86.9
✗	✗	87.2

Effect of object and scene concept filtering. In Table 12, we analyze the effects of two text concept filtering methods—*activation filter* and *projection filter*—on classification accuracy. Both filters follow the text concept filtering strategy described in Section 1.3.2, and we keep the motion dynamics concepts fixed throughout this experiment. The results show that applying both filters reduces the number of object and scene concepts to 282 and achieves the highest accuracy (87.5%), whereas using no filters results in a much larger concept set of 899 but significantly lower accuracy. These findings suggest that combining the activation and projection filters is more effective for improving concept quality and downstream performance.

6.2 Analysis

Sample level explanation. DANCE enables interpretation of the reasoning behind individual predictions by leveraging interpretable concept activations, which highlight how each concept contributes to the final decision. For a given input video, we compute each concept's contribution to the predicted class by multiplying its activation with the corresponding classifier weight. In Figure 11 ,we visualize the top-3 contributing concepts and their contribution to provide an intuitive understanding of the model's decision process.

Model level explanation. We visualize concept-to-class weights using Sankey diagrams to reveal how our model distinguishes between semantically similar actions. For each class pair, we show both shared and discriminative concepts, highlighting the model's reasoning pathway. In Figure 12–13, we show additional visualizations of model weights of similar action class pairs.

Failure case analysis. The transparency of DANCE's decision-making process enables us to analyze the reasoning behind incorrect predictions. Even when the model fails, the concept-level contributions

allow us to understand which concepts led to the misclassification. This interpretability helps identify whether the error stems from misleading motion dynamics, ambiguous object or scene cues, or dataset-specific biases. We illustrate several representative failure cases in Figure 14, covering diverse error sources across different concept types.

Sanity check. We evaluate whether DANCE properly interprets motion dynamics sequences by conducting two distinct configurations. First, we reverse the input video and input it into DANCE to observe how the prediction changes. This setting helps determine whether the model is sensitive to temporal direction, as reversing the motion should alter the concept activations if motion dynamics are properly captured. We show an example of this reversed input scenario in Figure 15. Second, we randomly select a frame from the video, replicate it to match the input format required by the inference pipeline, and observe how the model responds to the resulting static video. We show an example of this static input scenario in Figure 16.

Qualitative analysis of motion dynamics concepts. To evaluate whether our motion dynamics concepts effectively capture temporal context, we qualitatively compare them against concepts from UCF-101-attributes [39, 14] and GPT-4o-derived [13] temporal descriptions. In Figure 9, we visualize the top-3 contributing concepts for a *Hammer Throw* video, using CBMs trained separately with each concept set. The UCF-101-attributes concept "Two Arms Swinging" is vague and fails to specify how the arms are moving. The explanation by GPT-4o-derived temporal descriptions confuses the action with a discus throw and provides vague temporal cues (e.g., "knees slightly bent...") which are difficult to interpret due to their tacit knowledge. In contrast, our motion dynamics concepts capture clear temporal cues, such as the spinning motion and the final throwing action. Thanks to the explicitly disentangled motion dynamics concepts, DANCE provides more interpretable explanations of human actions in video. We quantitatively validate this through a user study presented in Section 4.1 of the main paper.

Limitation of video-text multimodal models. Recent video-text multimodal models often struggle to capture genuine temporal understanding due to the strong correlation between static visual cues and action labels in existing datasets [22, 20, 47, 27]. Many benchmarks allow models to align video-text pairs based on object or scene appearances alone, without requiring comprehension of dynamic motion or temporal progression. This limitation makes it difficult to leverage VidLMs for temporal concept grounding. While we adopt video-text multimodal models [55, 44] to obtain object and scene concepts, we find them less effective for modeling temporal descriptions associated with specific action classes. This is because the alignment between the input video and fine-grained temporal descriptions is often weak or inconsistent.

To validate this limitation, we evaluate how well a recent video-text multimodal model, ViCLIP [44], trained on 200M video-text pairs from InternVid [44], aligns with temporal descriptions in UCF-101 [39]. Specifically, we pass a video from the *BasketballShoot* class of UCF-101 through the video encoder to obtain a video embedding. We then pass GPT-4o-derived [13] temporal descriptions through the text encoder to obtain text embeddings, and compute the similarity between the video embedding and text embeddings. As illustrated in Figure 10, we observe that despite both input videos clearly depicting basketball shooting actions, ViCLIP assigns high similarity scores to unrelated descriptions such as those related to "Hulahoop". Even when provided with a precise and relevant description like "They jump and release the ball towards the hoop," the model fails to rank it highest, highlighting its difficulty in capturing temporal alignment.

6.3 Model Editing

Sample-level intervention. We demonstrate how DANCE's predictions change depending on the contribution of specific concepts. For each input video, we identify the predicted class and the top contributing concepts, then deactivate the concept that interferes with the correct prediction and observe how the output changes. We perform deactivation by setting the activation value of the selected concept to zero. In Figure 17, we show additional results of sample level intervention.

In domain class-level intervention. In Figure 18, we present several examples where user intervention on the DANCE's concept-to-class weights leads to improved predictions. Using the same dataset for both training and testing, we demonstrate that modifying the learned weights between concepts and classes in DANCE can successfully alter the prediction outcomes. Specifically, due to the sparsity of the model's classification layer, some concepts may have high activation but contribute

nothing to the prediction because their corresponding weights are zero. To address this issue, we allow user intervention in frequently confused class pairs by: (i) identifying concepts with high activation (ii) checking which of them are not connected to the predicted class (iii) modifying the weights of potentially helpful concepts to establish a connection with the correct class.

Cross-domain class-level intervention. We demonstrate that DANCE can correct counterintuitive predictions caused by domain shifts or bias through intervention at the concept-to-class weight level—without requiring additional data or retraining. To demonstrate this, we examine examples from the domain-shifted datasets UCF-101-SCUBA [21] and Mimetics [45]. We show additional results in Figure 19, where DANCE misclassifies inputs due to incorrect concept correlations, and we successfully correct the predictions through targeted interventions.

7 Limitations

While DANCE provides effective and interpretable explanations by disentangling motion dynamics, object, and scene concepts, it has a few limitations. First, due to the absence of explicit object and scene annotations in most video datasets, DANCE relies on a vision-language model to generate pseudo labels. Consequently, the resulting object and scene representations may inherit limitations of the underlying model, such as language priors and static visual biases [31, 3, 22, 47, 20, 27]. Second, the motion dynamics concepts depend on the pose estimation quality. Inaccurate pose estimates can lead to misleading or non-intuitive motion explanations. In scenarios where no humans are visible or relatively irrelevant-such as object-centric activities-DANCE might not leverage motion concepts and must rely solely on object and scene cues, which may reduce the completeness or fidelity of the explanation.

To reduce reliance on video-language models for spatial concept supervision, a promising direction is to develop methods that directly discover object and scene attributes from video, without requiring a pretrained multimodal encoder. To mitigate sensitivity to pose estimation quality, future work could explore more robust pose tracking algorithms or alternative representations of motion dynamics that do not depend on explicit keypoints. Finally, extending DANCE to human-absent or object-centric videos may involve generalizing motion dynamics into broader visual dynamics—such as using object keypoints or modeling temporal interactions among visual entities—to capture meaningful motion patterns beyond human pose.

8 Broader Impacts

This work aims to advance explainability in video action recognition by explicitly disentangling and modeling spatial and temporal concepts. In contrast to prior approaches that rely on entangled or opaque feature representations, our framework provides semantically grounded explanations that promote greater transparency and interpretability. We believe this contributes to the foundation for developing trustworthy and accountable video understanding systems, particularly in high-stakes domains such as healthcare, surveillance, education, and human–AI interaction. Additionally, our concept-based formulation can be extended beyond full-body human action to partially observed or non-human action scenarios, such as egocentric videos involving only hand gestures or object manipulations. By generalizing motion dynamics into broader visual dynamics—e.g., using gesture keypoints or modeling temporal dependencies among scene elements such as objects—our approach holds potential for applications in instructional video analysis, assistive technologies, and human-robot collaboration. We hope this line of research fosters safer, more interpretable, and human-aligned video AI systems. We anticipate no direct negative societal impacts, as the focus of this work is on enhancing interpretability.

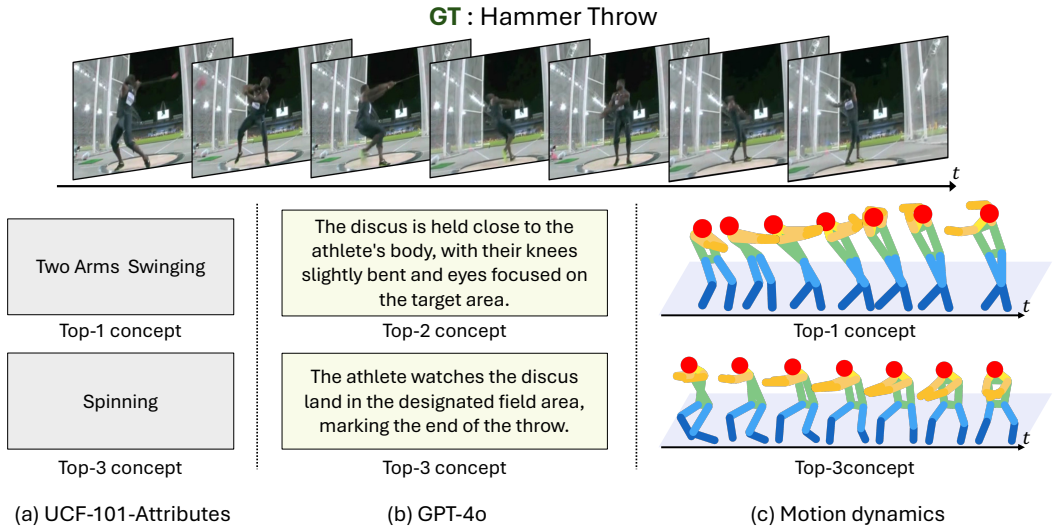


Figure 9: **Qualitative visualization of motion dynamics concepts.** We visualize the top-3 contributing concepts for a *Hammer Throw* video, using CBMs trained separately with UCF-101-Attributes, GPT-4o-generated concepts, and our proposed motion dynamics concepts.

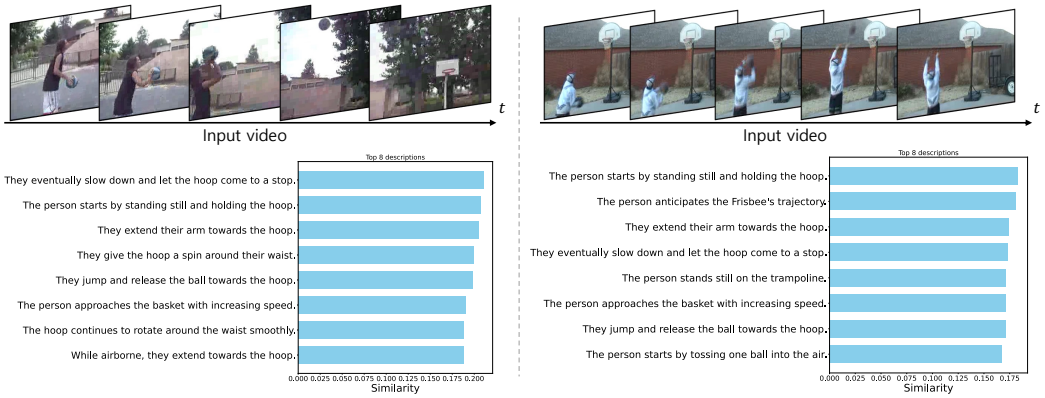


Figure 10: **Two examples showing misalignment between video and text embeddings in ViCLIP.** We input two videos from the UCF101 "Basketball" class into the ViCLIP video encoder and compute similarities with temporal descriptions passed through the text encoder. Although the input videos clearly depict basketball shooting actions, ViCLIP assigns high similarity scores to unrelated descriptions (e.g., involving hoop or trampoline), while more relevant ones such as "They jump and release the ball towards the hoop" are ranked lower. This suggests that the model relies on static object cues like the word "hoop", which appears in both basketball and hula hoop contexts, without properly capturing the temporal dynamics of the action. These results highlight the limitation of video-text multimodal models in aligning visual content with fine-grained temporal descriptions.

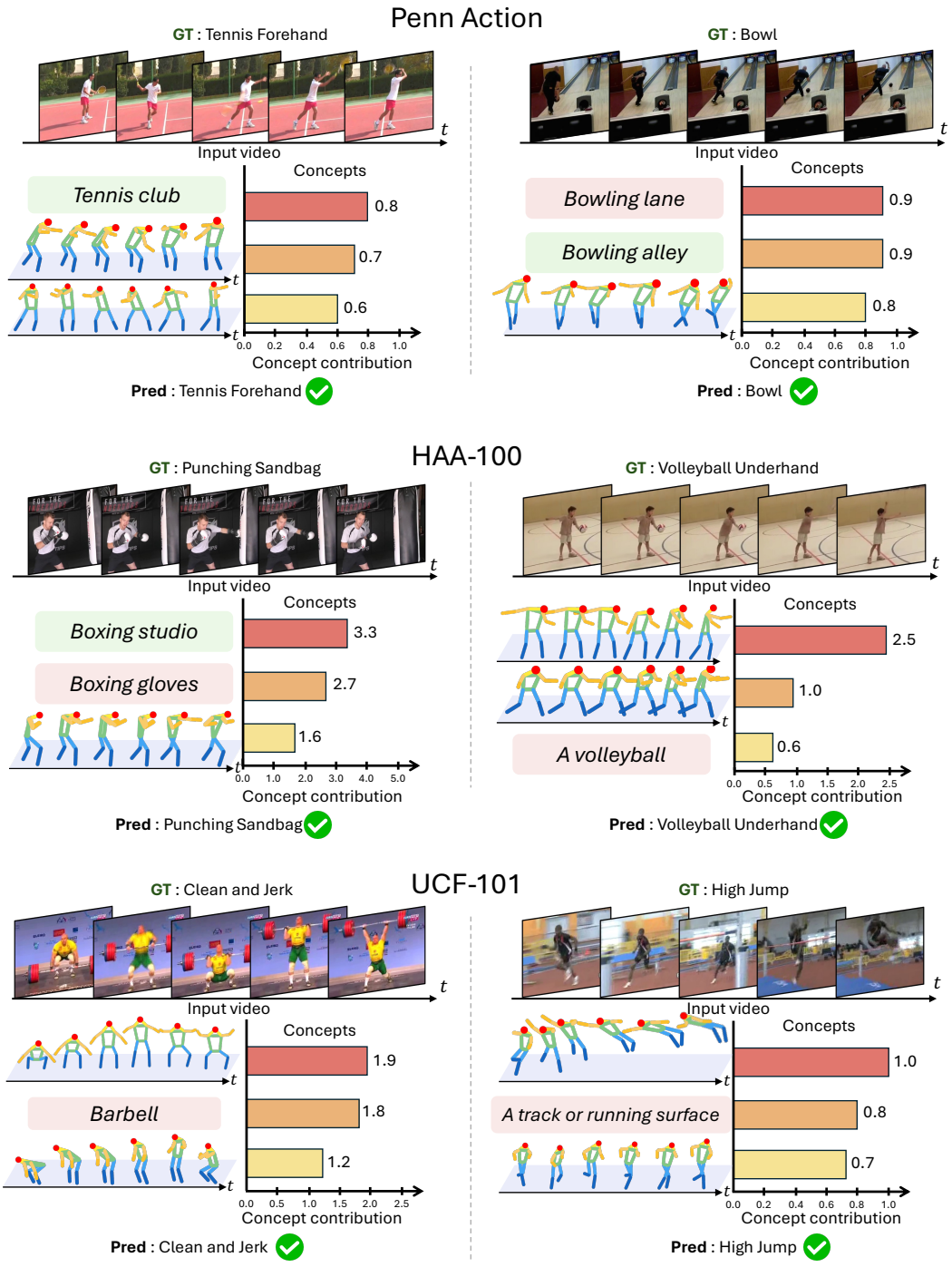


Figure 11: **Additional sample-level explanations of DANCE.**

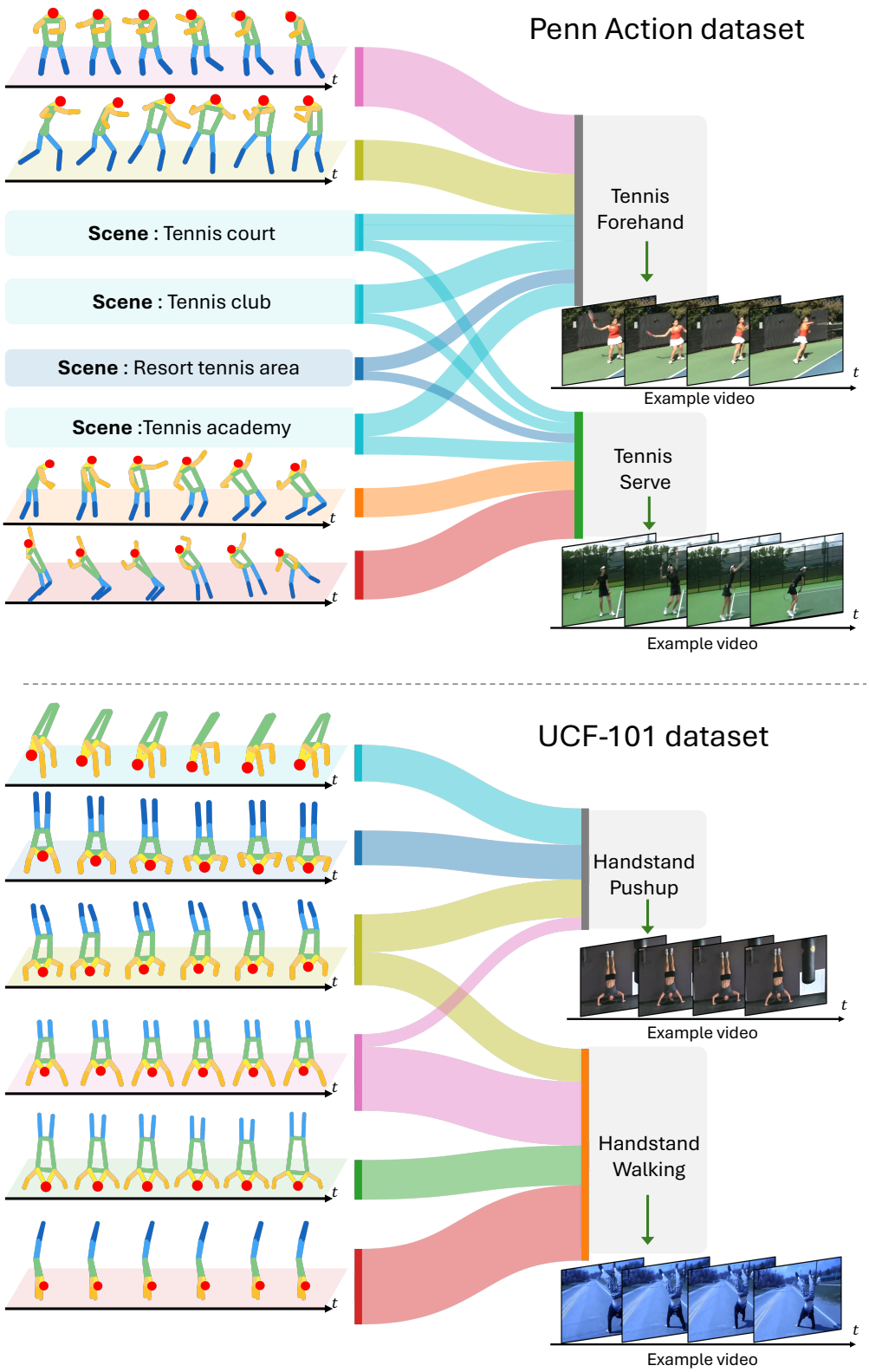


Figure 12: **Additional model-level explanations of DANCE on Penn Action and UCF-101.** we visualize concept-to-class weights for pairs of similar action classes using Sankey diagrams. For each class, we also provide one example video to illustrate the corresponding motion pattern.

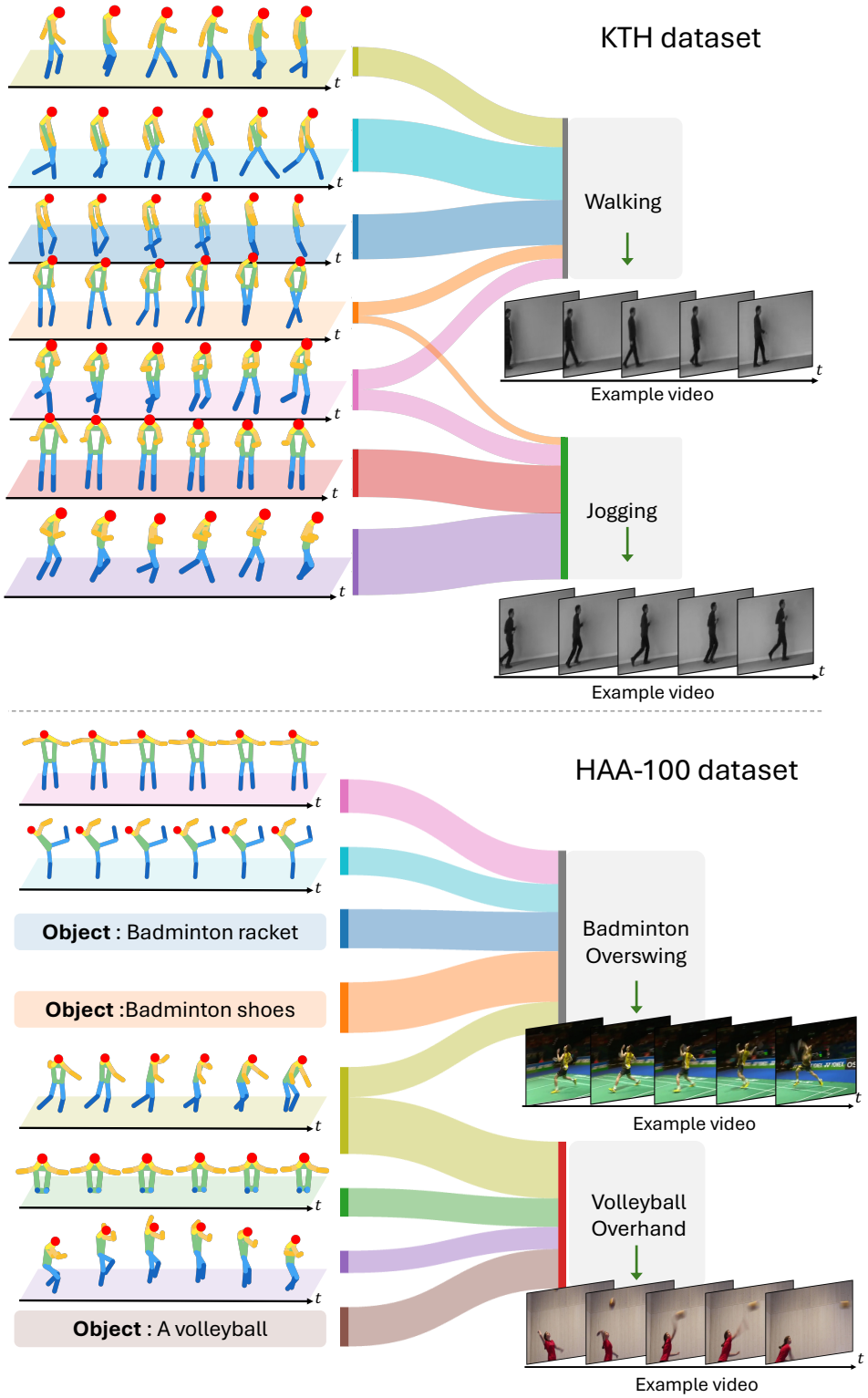


Figure 13: **Additional model-level explanations of DANCE on KTH and HAA-100.** we visualize concept-to-class weights for pairs of similar action classes using Sankey diagrams. For each class, we also provide one example video to illustrate the corresponding motion pattern.

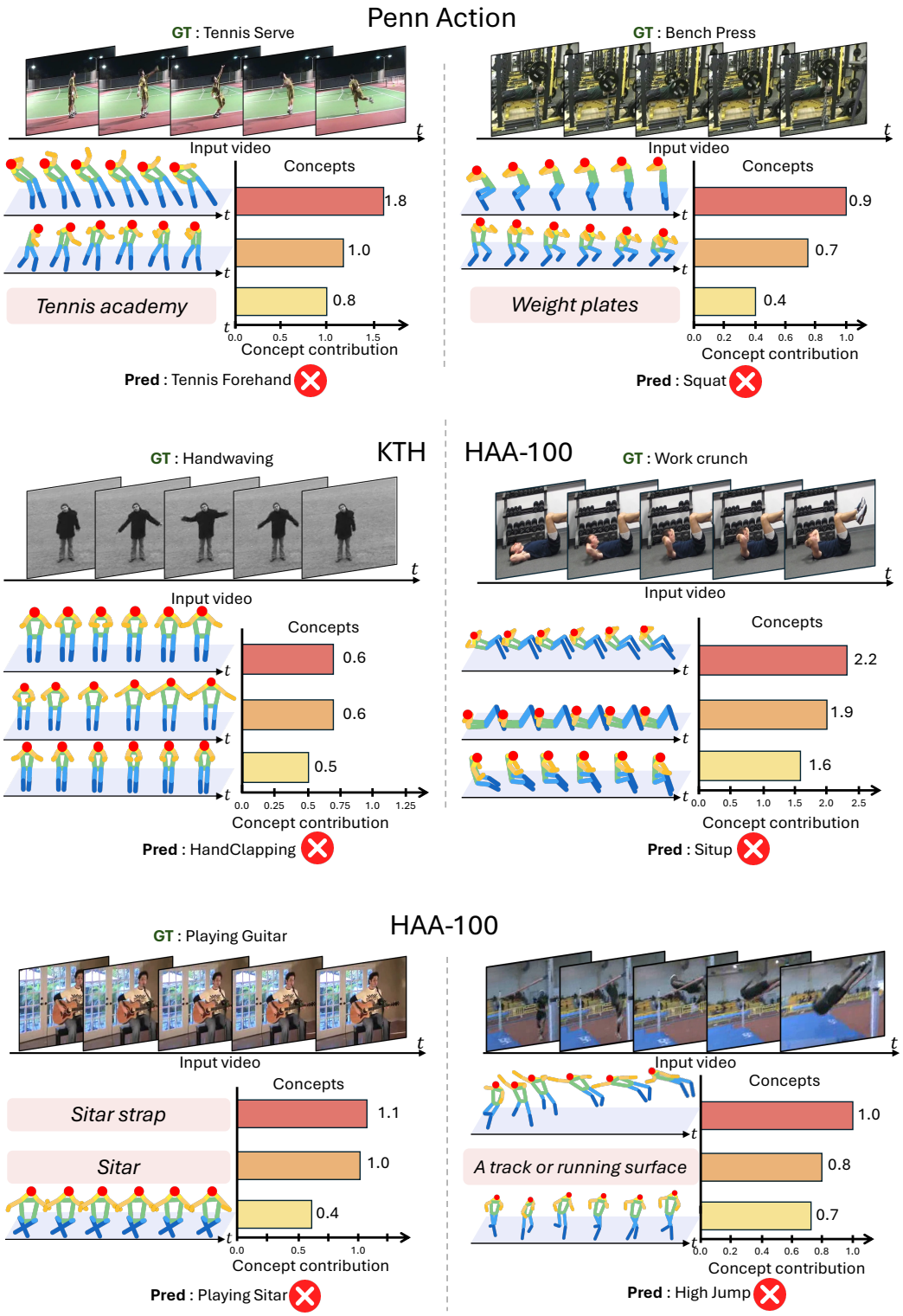


Figure 14: **Additional failure case analyses using DANCE.**

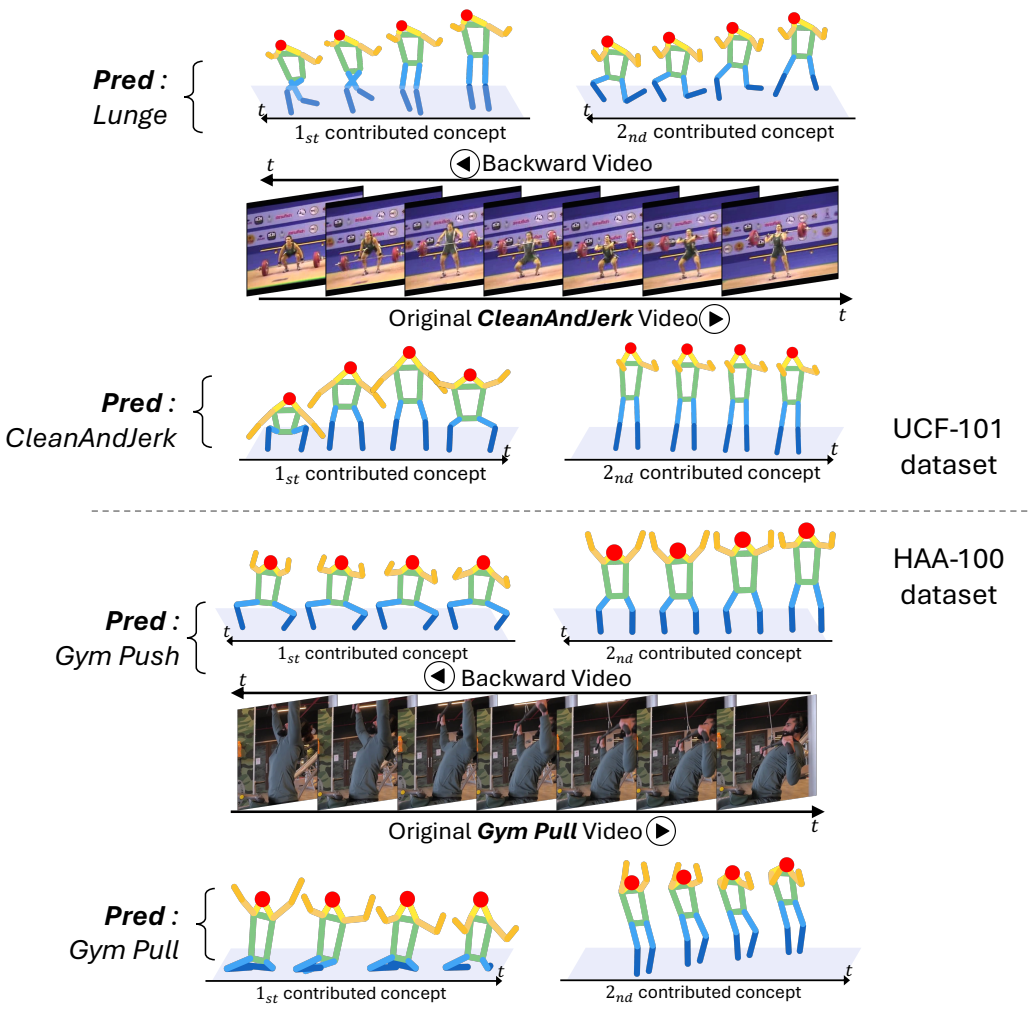


Figure 15: **Sanity check of DANCE with a backward video.** We compare predictions and top-2 contributing concepts of DANCE for (i) the original video, and (ii) the same video played backward.

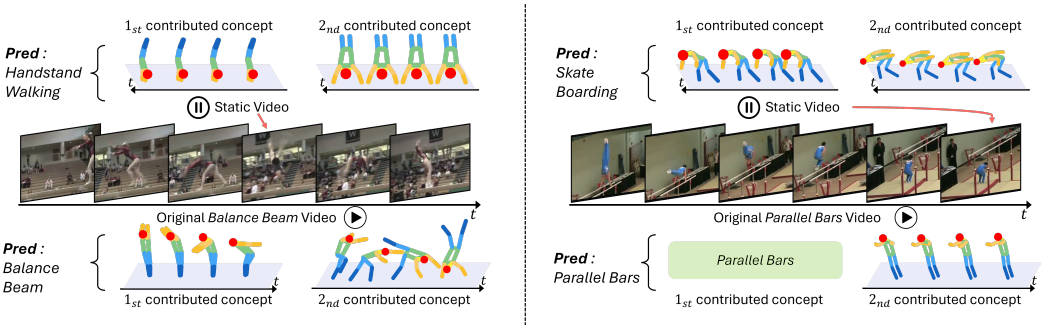


Figure 16: **Sanity check of DANCE with a static video.** We compare predictions and top-2 contributing concepts of DANCE for (i) the original video, and (ii) a static version created by repeating a single frame from the original.

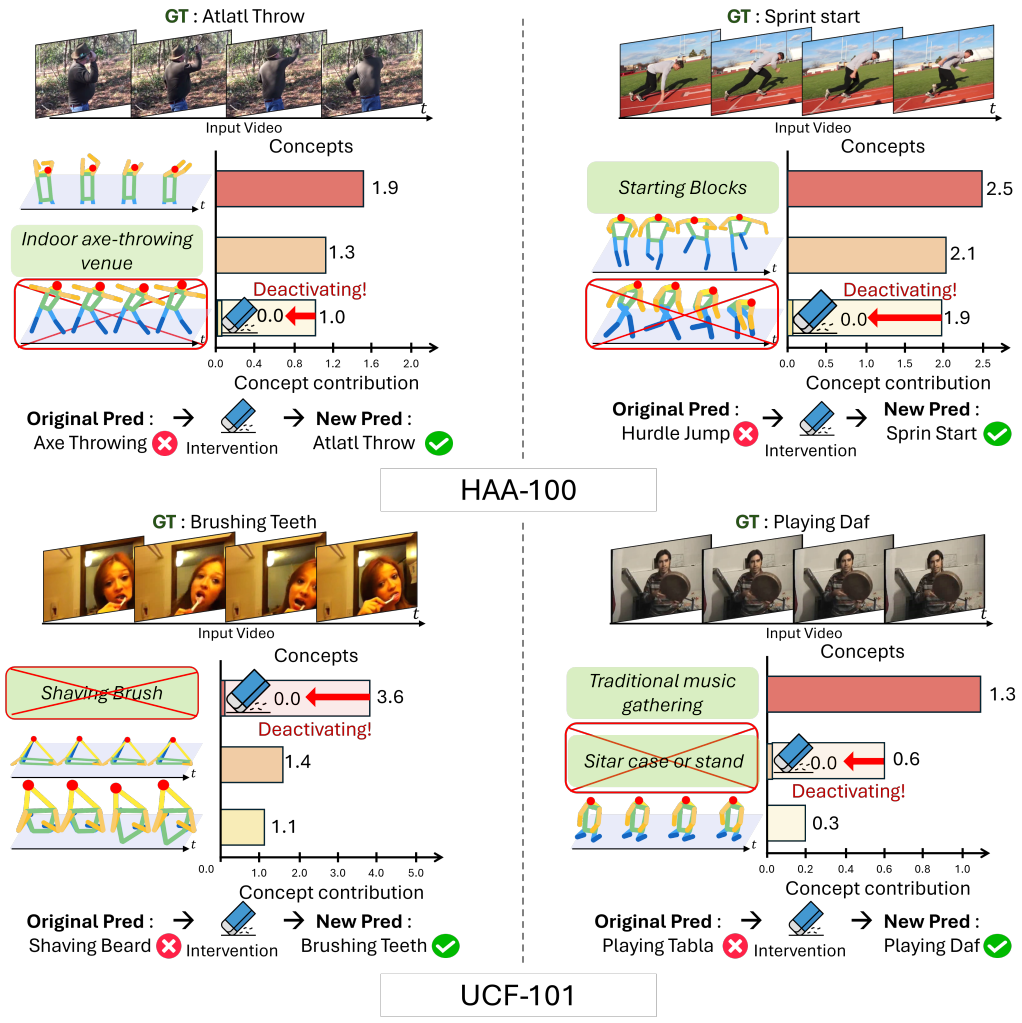


Figure 17: **Additional sample-level interventions of DANCE.**

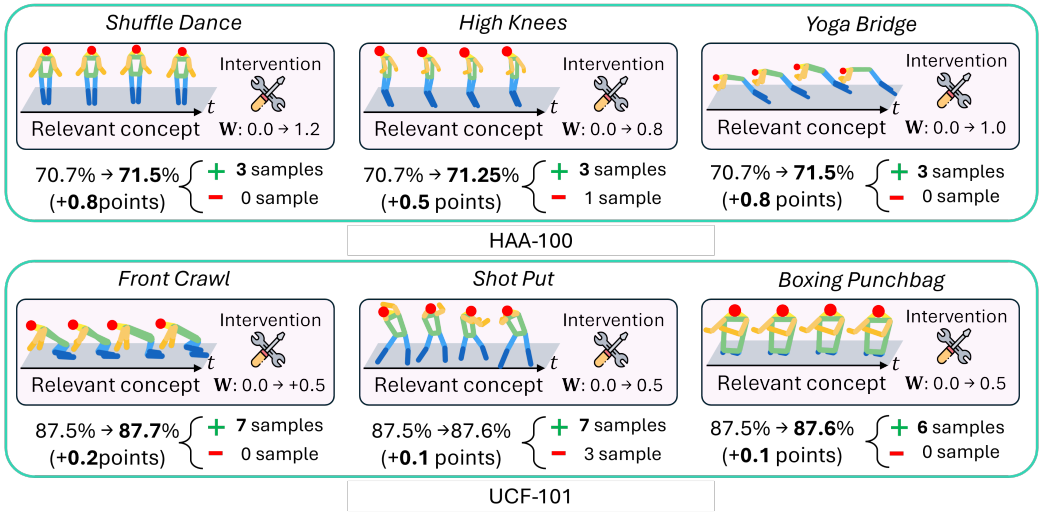


Figure 18: **Additional class-level interventions of DANCE.**

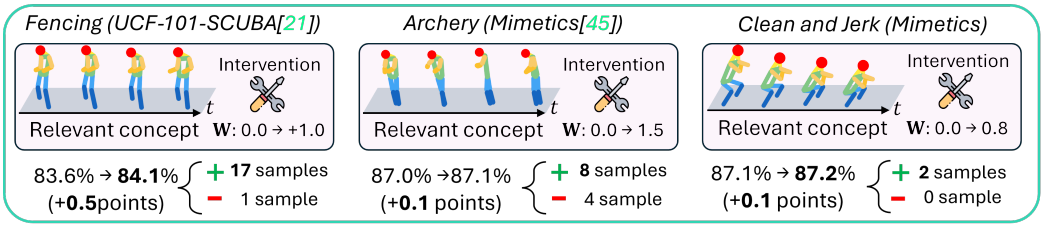


Figure 19: **Additional cross domain class-level interventions of DANCE.**

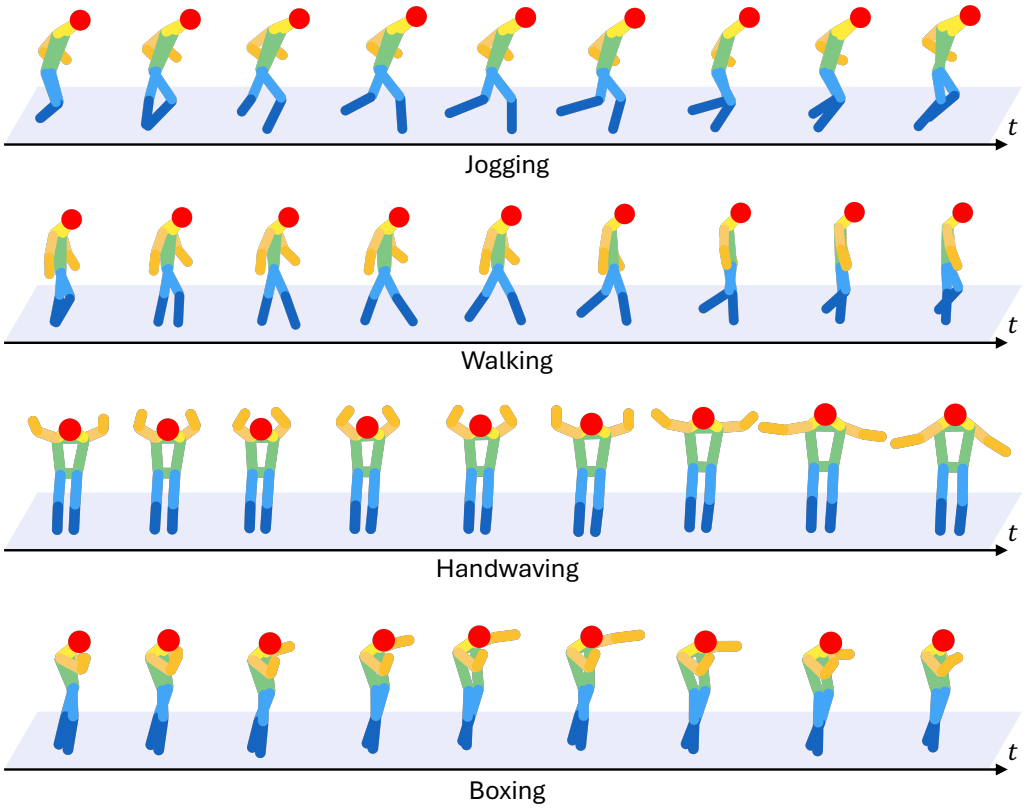


Figure 20: **Visualization of Motion Dynamics Concepts on KTH dataset.** The class label of the source video is shown below each concept for reference.

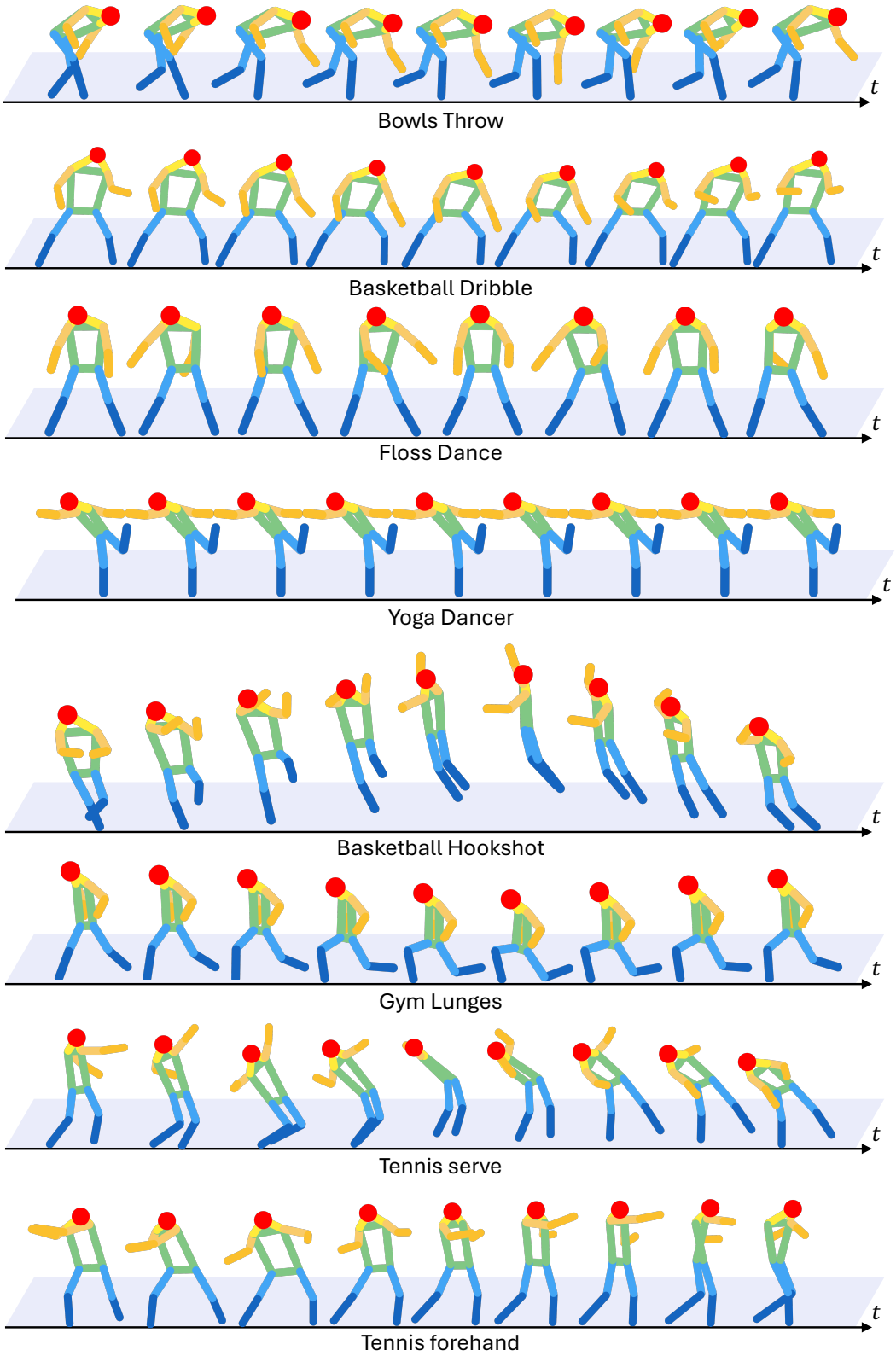


Figure 21: **Visualization of Motion Dynamics Concepts on HAA-100 dataset.** The class label of the source video is shown below each concept for reference.

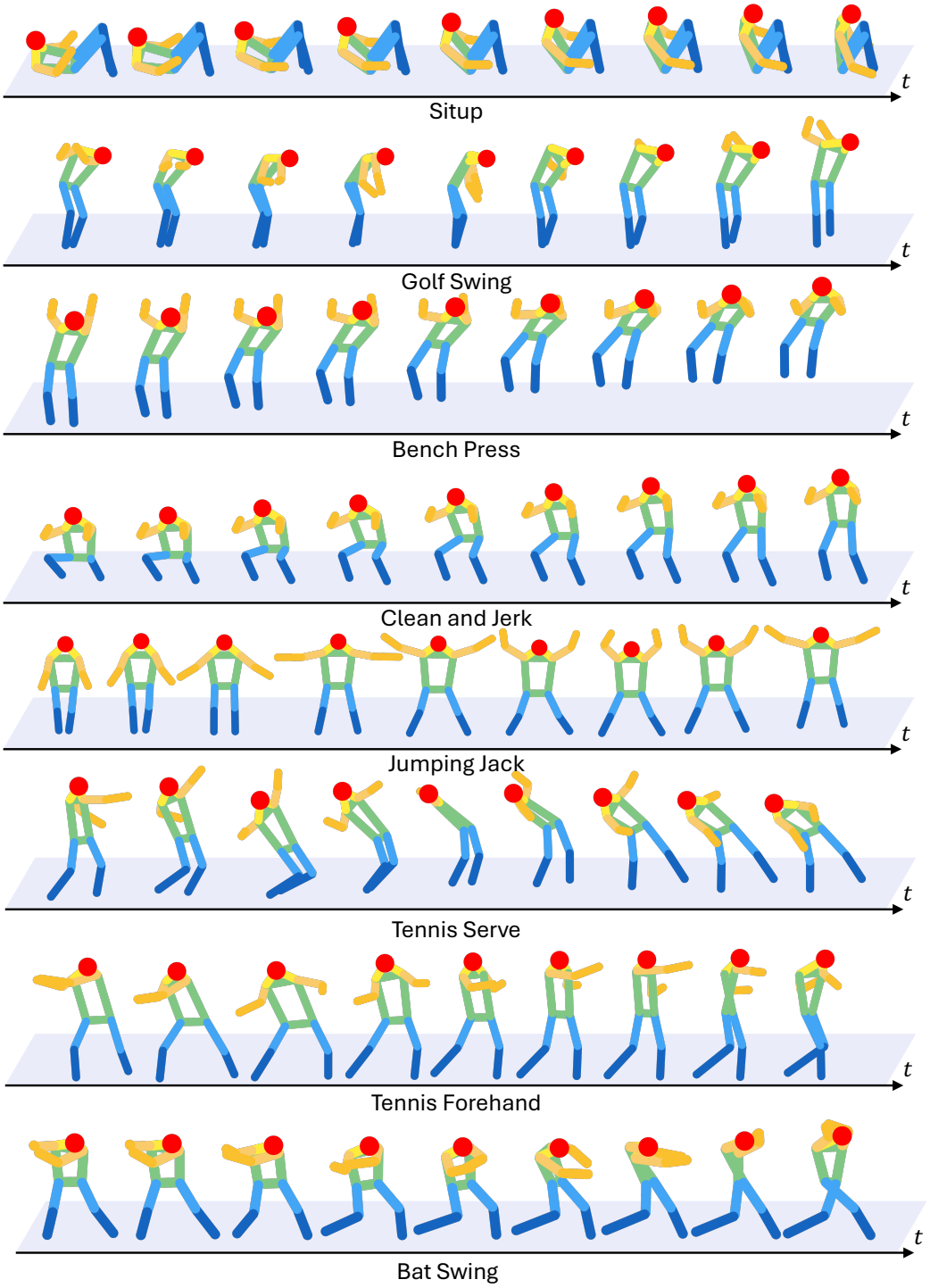


Figure 22: **Visualization of Motion Dynamics Concepts on Penn Action dataset.** The class label of the source video is shown below each concept for reference.

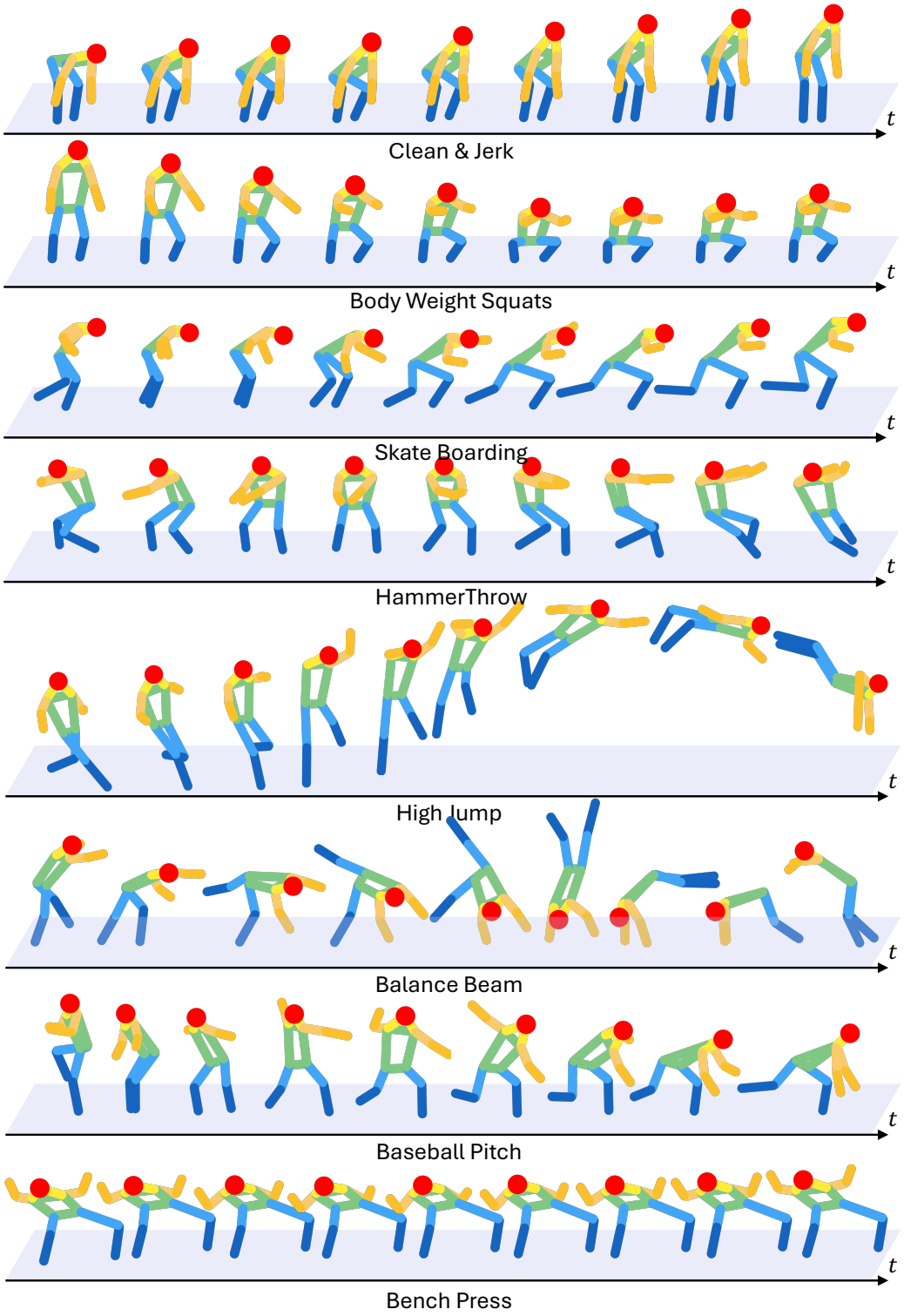


Figure 23: **Visualization of Motion Dynamics Concepts on UCF-101 dataset.** The class label of the source video is shown below each concept for reference.

References

[1] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 5

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, 2021. 2

[3] Shyamal Buch, Cristóbal Eyzaguirre, Adrien Gaidon, Jiajun Wu, Li Fei-Fei, and Juan Carlos Niebles. Revisiting the" video" in video-language understanding. In *CVPR*, 2022. 19

[4] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020. 5

[5] Jihoon Chung, Cheng-hsin Wuu, Hsuan-ru Yang, Yu-Wing Tai, and Chi-Keung Tang. Haa500: Human-centric atomic action dataset with curated videos. In *ICCV*, 2021. 5, 7, 10

[6] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, 2020. 5

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 8, 14

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 5

[9] 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*, 2017. 5, 12

[10] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In *ICCV*, 2017. 17

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 8

[12] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 5

[13] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 3, 10, 11, 12, 14, 15, 18

[14] H. Idrees, A. R. Zamir, Y. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah. The thumos challenge on action recognition for videos “in the wild”. *CVIU*, 155:1–23, 2017. 7, 9, 10, 11, 12, 14, 18

[15] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[16] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *ICML*, 2020. 2, 9, 10, 11, 12, 14

[17] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982. 12

[18] Matthew Kowal, Achal Dave, Rares Ambrus, Adrien Gaidon, Konstantinos G Derpanis, and Pavel Tokmakov. Understanding video transformers via universal concept discovery. In *CVPR*, 2024. 8, 10, 11, 12

[19] Dongho Lee, Jongseo Lee, and Jinwoo Choi. Cast: Cross-attention in space and time for video action recognition. In *NeurIPS*, 2023. 6

[20] Jie Lei, Tamara L Berg, and Mohit Bansal. Revealing single frame bias for video-and-language learning. *arXiv preprint arXiv:2206.03428*, 2022. 18, 19

[21] Haoxin Li, Yuan Liu, Hanwang Zhang, and Boyang Li. Mitigating and evaluating static bias of action representations in the background and the foreground. In *ICCV*, 2023. 7, 19

[22] Shicheng Li, Lei Li, Yi Liu, Shuhuai Ren, Yuanxin Liu, Rundong Gao, Xu Sun, and Lu Hou. Vitatecs: A diagnostic dataset for temporal concept understanding of video-language models. In *ECCV*, 2024. 18, 19

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 3, 5

[24] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023. 14

[25] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 5

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5

[27] Wufei Ma, Kai Li, Zhongshi Jiang, Moustafa Meshry, Qihao Liu, Huiyu Wang, Christian Häne, and Alan Yuille. Rethinking video-text understanding: Retrieval from counterfactually augmented data. In *ECCV*, 2024. 18, 19

[28] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, volume 5, pages 281–298. University of California press, 1967. 12

[29] Jingjing Meng, Hongxing Wang, Junsong Yuan, and Yap-Peng Tan. From keyframes to key objects: Video summarization by representative object proposal selection. In *CVPR*, 2016. 2

[30] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. 14

[31] Liliane Momeni, Mathilde Caron, Arsha Nagrani, Andrew Zisserman, and Cordelia Schmid. Verbs in action: Improving verb understanding in video-language models. In *ICCV*, 2023. 19

[32] Tuomas Oikarinen, Subhro Das, Lam Nguyen, and Lily Weng. Label-free concept bottleneck models. In *ICLR*, 2023. 2, 3, 4, 6, 9, 10, 11, 12

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 4

[34] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 2, 17

[35] Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In *CVPR*, 2019. 3, 12, 15

[36] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *ICPR*, 2004. 5, 6, 10

[37] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 8

[38] Jaewon Son, Jaehun Park, and Kwangsu Kim. Csta: Cnn-based spatiotemporal attention for video summarization. In *CVPR*, 2024. 2, 12, 13

[39] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. A dataset of 101 human action classes from videos in the wild. *Center for Research in Computer Vision*, 2(11):1–7, 2012. 5, 6, 7, 10, 11, 12, 14, 18

[40] Alexandros Stergiou, Georgios Kapidis, Grigorios Kalliatakis, Christos Chrysoulas, Remco Veltkamp, and Ronald Poppe. Saliency tubes: Visual explanations for spatio-temporal convolutions. In *ICIP*, 2019. 8

[41] Hao Tang, Lei Ding, Songsong Wu, Bin Ren, Nicu Sebe, and Paolo Rota. Deep unsupervised key frame extraction for efficient video classification. *ACM Transactions on Multimedia Computing, Communications and Applications*, 19(3):1–17, 2023. 2

[42] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *NeurIPS*, 2022. 1, 4, 5, 6, 8, 9, 10, 12, 15, 17

[43] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *TPAMI*, 41(11):2740–2755, 2018. 5

[44] Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, et al. Internvid: A large-scale video-text dataset for multimodal understanding and generation. In *ICLR*, 2023. 2, 4, 9, 11, 16, 17, 18

[45] Philippe Weinzaepfel and Grégory Rogez. Mimetics: Towards understanding human actions out of context. *IJCV*, 129(5):1675–1690, 2021. 7, 8, 19

[46] Eric Wong, Shibani Santurkar, and Aleksander Madry. Leveraging sparse linear layers for debuggable deep networks. In *ICML*, 2021. 6

[47] Junbin Xiao, Angela Yao, Yicong Li, and Tat-Seng Chua. Can i trust your answer? visually grounded video question answering. In *CVPR*, 2024. 18, 19

[48] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. In *NeurIPS*, 2022. 3, 5

[49] Xiang Yan, Syed Zulqarnain Gilani, Hanlin Qin, Mingtao Feng, Liang Zhang, and Ajmal Mian. Deep keyframe detection in human action videos. *arXiv preprint arXiv:1804.10021*, 2018. 2

[50] Xiang Yan, Syed Zulqarnain Gilani, Mingtao Feng, Liang Zhang, Hanlin Qin, and Ajmal Mian. Self-supervised learning to detect key frames in videos. *Sensors*, 20(23):6941, 2020. 2

[51] Taojiannan Yang, Yi Zhu, Yusheng Xie, Aston Zhang, Chen Chen, and Mu Li. Aim: Adapting image models for efficient video action recognition. *arXiv preprint arXiv:2302.03024*, 2023. 6

[52] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *ICLR*, 2023. 7

[53] HongJiang Zhang, Atreyi Kankanhalli, and Stephen W Smoliar. Automatic partitioning of full-motion video. *Multimedia systems*, 1:10–28, 1993. 2, 12, 13

[54] Weiyu Zhang, Menglong Zhu, and Konstantinos G. Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *ICCV*, 2013. 5, 6, 7, 10

[55] Yue Zhao, Ishan Misra, Philipp Krähenbühl, and Rohit Girdhar. Learning video representations from large language models. In *CVPR*, 2023. 2, 17, 18

[56] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *TPAMI*, 40(6):1452–1464, 2017. 7, 14