LEARNING JOINT VIDEO AND TEXT REPRESENTATIONS AND TAXONOMY OF ACTIONS IN VIDEOS

Anonymous authors
Paper under double-blind review

ABSTRACT

The video of a complex action can be decomposed into a temporally ordered sequence of simpler sub-actions, each of which can, in turn, be recursively decomposed to form a video tree. Given a video whose only annotation is a set of names of possible complex action contained in it, this paper aims to simultaneously segment the video into clips of the complex actions, learn a video tree of sub-actions for each segment, and assign linguistic names to its nodes. Each video tree has a co-existing text tree, formed by the text names of the video tree nodes. We refer to both tress together as the *definition tree*. In the absence of annotations, we generate text trees using LLMs. To construct video trees, we recursively divide estimated action segments in video into ordered temporal sub-actions. For both trees we compute representation features bottom-up, starting from the leaf actions. Action–sub-action attention (ASA) lets neighboring sub-actions exchange information before they are combined into a parent action in both trees. As a second objective, we use taxonomic supervision to draw semantically related actions close to each other to form a taxonomy of the complex actions in the input video. Compared with Ti-FAD, our video features better preserve the temporal order of sub-actions for unseen actions (+16.7%), yield better taxonomic grouping (+18.2%), and improve zero-shot localization mAP by 9.0-10.8% on THUMOS14 and 2.3-3.9% on ActivityNet.

1 INTRODUCTION

Understanding complex actions in videos is central to applications such as robotic surgery, sports coaching, dance choreography, and equipment assembly. Given a video whose only annotation is a set of possible complex action names that the video may contain, this paper aims to simultaneously (i) segment the video into clips of complex actions from the set, (ii) learn a video tree of sub-actions for each segment, (iii) assign linguistic names, that is, words or phrases, to the nodes of each tree, and (iv) build a taxonomy of the given actions, in which actions with similar names are recursively grouped under a coarser action. The models to do this are trained on a class of videos of complex actions. The test actions may not have been seen during training, but they may share sub-actions with the actions in the training videos.

Video of a complex action can be decomposed into a temporally ordered sequence of simpler, shorter, child sub-actions, each of which can, in turn, be decomposed further. This forms the complex action's *video tree*, where the root node represents the entire complex action, at the most semantic level, the interior nodes are sub-action nodes, and the leaf nodes represent the shortest sub-actions, at the least semantic level; nodes at each level being an ordered sequence of the parent's sub-actions. Since actions and sub-actions are all actions in their own right, differing only in their relative positions in the tree, we refer to both as actions except when their parent-child relationships need to be made explicit. We learn video tree representations of actions in the given video that capture both the sub-actions shared across classes and their composition into different complex actions, and evaluate this learned representations through zero-shot action recognition and localization in unseen test videos.

The video tree of each action also has a corresponding *text tree*. For example, the video tree for *long jump* contains videos of ordered sub-actions named *run-up*, *leap* and *landing*. Linguistic conventions often give semantically related actions similar names, although their video trees can differ substantially because of the differences in the compositions of the sub-action videos. For

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

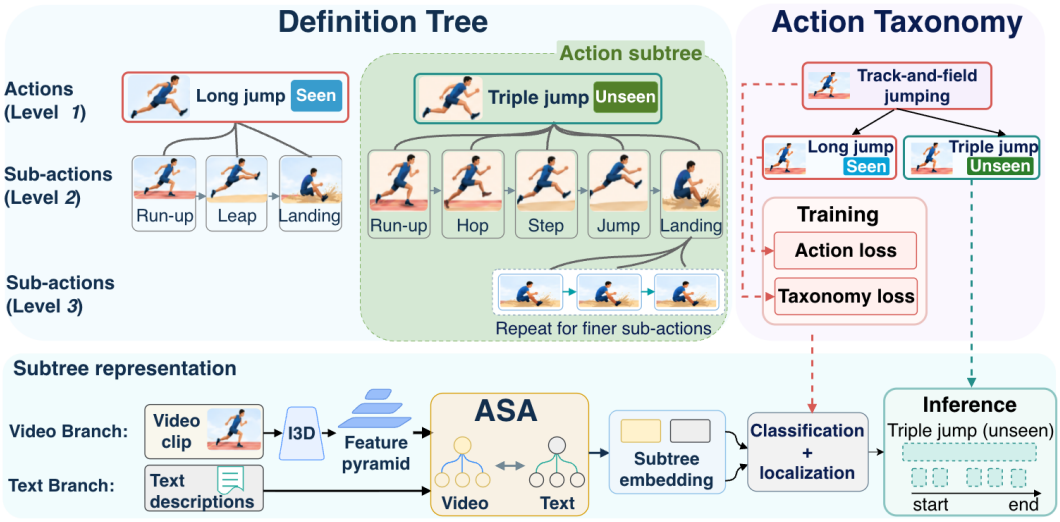

Figure 1: Overview of learning a joint video–text representation and Taxonomy. Top: a video tree recursively decomposes the video of an action into temporally ordered sub-actions, text tree mirrors the video tree but with text names of the action nodes, and the taxonomy groups action classes based on their semantics. The green region illustrates a subtree of the action triple-jump. Bottom: Action–Sub-action Attention (ASA) and cross-modal (video and text) attention constructs subtree embeddings for action classification and localization. Training uses action and taxonomy supervision. Inference involves detection, recognition and segmentation of a previously unseen action.

example, *long jump* and *triple jump* share the word "jump" but differ in their sequence of take-offs. Consequently, the accepted name of a new complex action, formed by combining its sub-action names may not be easy to translate into the corresponding new video tree. In our jump action example, the new action name may be *triple jump*, which differs by a single word from the known action names *long jump* and *high jump* but their corresponding videos differ substantially: *triple jump* involves a hop onto the same foot, a step onto the other foot, and a final jump, giving it a different temporal structure from the single leap in *long jump*. The two actions also share sub-actions such as run-up and landing, allowing the model to use what it learns about these shared parts when representing a new action. For brevity, we will refer to the video and text tree components of an action representation together by *definition tree*. Besides their definition trees, we also address a different inter-action relationship, captured in the linguistic, semantic naming taxonomy. In our jump example, *long jump* and *triple jump* are both kinds of *jumping*. Such relations are organized in an action taxonomy, which groups actions into coarser categories based on linguistic criteria, rather than based on the exact sub-actions. The taxonomy and the definition trees meet at the level of the action classes: each class is a leaf of the taxonomy and the root of its own definition tree. As a second objective of our paper, we extend learning of definition tree representation, to also ensure that semantically related actions such as *long jump* and *triple jump* are close to each other. We achieve this using taxonomic supervision: in addition to recognizing each action, the model is trained to recognize the taxonomy category it belongs to, which draws the representations of related actions together. We use existing dataset taxonomies when available and manually define the class groupings for THUMOS14 and ActivityNet (Appendix A.4).

Given a set of untrimmed, unannotated test videos and the names of candidate actions that they may contain, our method obtains video tree representations of the complex actions in these videos using models trained on videos of seen action classes. Since labels of these actions are available during training but their sub-actions are neither labeled nor temporally segmented, we first generate text trees of the candidate actions using LLMs. We recursively divide the estimated action intervals in video into ordered temporal sub-actions to construct the video tree. For both trees we compute the representation features bottom-up, starting from the leaf actions. We introduce action–sub-action attention (ASA), which lets neighboring sub-actions exchange information before we combine them with their parent action, in both the video and the text trees. The video representation parts of the definition trees are used for two recognition tasks during training: recognizing the action, such as *long jump*, from its

action label, and recognizing its taxonomy category, such as *jumping*, through taxonomy supervision. The first task keeps related actions distinguishable by their definition trees, while the second draws them together under their shared category. To evaluate our learned representations, we test them on videos of new action classes, having new linguistic names whose class labels were not encountered during training. We perform zero-shot recognition of these new actions in a test video and localize each action instance while obtaining its video representation. Learning such representations using normal supervised training would require annotating the training videos with definition trees, which would be expensive and prone to annotation errors and inter-annotator inconsistencies. This is a major reason why video representations in existing methods describe actions mostly at a single level (Carreira & Zisserman, 2017; Bertasius et al., 2021). The existing hierarchical datasets (Shao et al., 2020; Xu et al., 2022) use either taxonomic or action-sub-action criteria for defining the hierarchy, not both as is the case in our method in this paper. None of existing methods composes an action hierarchy from its ordered sub-actions recursively, at any depth, without sub-action annotations: Ju et al. (2023) describes each action as a whole; Ghoddoosian et al. (2022) and HierVL (Ashutosh et al., 2023) use two fixed, human-annotated levels; and PDA (Zhu et al., 2026a) uses a fixed number, three, of sub-actions per action. Compared with Ti-FAD (Lee et al., 2024), which is used as the baseline by most methods having SOTA performance, our video features better preserve temporal order of sub-actions for unseen actions (Kendall rank correlation, +16.7%), yields better taxonomic grouping of unseen actions (silhouette coefficient, +18.2%), and gives better zero-shot localization performance, improving mAP by 9.0–10.8% on THUMOS14 (Jiang et al., 2014) and 2.3–3.9% on ActivityNet v1.3 (Heilbron et al., 2015). Other similar favorable results of our method compared to other SOTA methods are given in Sec. 4.

Our contributions are as follows: (1) a hierarchical video representation based on definition trees, in which each action is recursively composed of a sequence of temporally ordered sub-actions; (2) construction of these tree representations in videos without sub-action annotations, by partitioning estimated action intervals according to LLM-generated text trees; (3) action–sub-action attention (ASA), which lets neighboring sub-actions exchange information before combining them into a common parent action, in both video and text; (4) supervision by manually constructed taxonomy to group related actions together; (5) improved unseen-action localization under strict zero-shot protocols on THUMOS14 and ActivityNet, and under coarse supervision on FineGym and FineDiving.

2 Related Work

Hierarchical and Compositional Action Representations. MViT (Fan et al., 2021) builds visual feature hierarchies across resolutions. Semantic hierarchies relate complex actions to their sub-actions: Ghoddoosian et al. (2022) recognize the overall task and restrict segmentation to action sequences associated with it. HierVL (Ashutosh et al., 2023) aligns clips and whole videos with descriptions at both levels. LogicCAR (Ye et al., 2025) uses logical constraints on verb-object combinations and their semantic relationships to recognize unseen combinations. For low-shot localization, Ju et al. (2023) use detailed action descriptions or video-conditioned text prompts. PDA (Zhu et al., 2026a) describes three temporal phases and the overall action, filters visual features for each phase, and combines the matching scores with learned weights. Our decomposition has multiple temporal levels, with child counts chosen separately at each level. ASA recursively composes child representations into their parent in both video and text, using nested partitions of estimated action intervals in the video branch. Taxonomy supervision uses coarser labels from a separate action taxonomy.

Open-Vocabulary Temporal Action Localization. Zero-shot TAL localizes classes without labeled training examples. STALE (Nag et al., 2022) jointly learns localization and classification through vision–language prompting. Ti-FAD (Lee et al., 2024) combines text-video cross-attention with foreground-aware modeling. DFAlign (Zhu et al., 2026b) uses text-conditioned diffusion to suppress background and refine text prompts, while TF-CADE (Lee et al., 2026) weights foreground features and refines location scores through text–video similarity. Other methods change the training data or adaptation procedure. STOV-TAL (Hyun et al., 2025) retrains a class-agnostic localizer using temporal pseudo-labels from unlabeled videos. FreeZAD (Han et al., 2026) scores candidate segments with a pretrained vision-language model without training on seen-action annotations and offers a test-time adaptation variant. Bi-TSA (Tan et al., 2026) adapts to unlabeled test videos by predicting neighboring segment features and evaluates seen and unseen classes jointly. We build on Ti-FAD, adding subtree composition and taxonomy supervision to learn the hierarchical representation.

3 METHOD

Our goal is to learn hierarchical video representations that combine each action with its ordered sub-actions, recursively, without sub-action annotations. We first define the definition tree, its text and video forms (branches), and how it differs from the action taxonomy in Figure 1 (Section 3.1). An LLM generates the text tree, and we construct its video counterpart by recursively dividing estimated action intervals into ordered parts. ASA then composes each action with its sub-actions into video and text subtree embeddings (Section 3.2). We train these embeddings with action classification and localization losses and with taxonomy supervision, which draws related actions together (Section 3.3). At inference, the attribute similarity prior (ASP) adjusts unseen-action confidence using the typical lengths of seen actions with similar attributes (Section 3.4). We build on Ti-FAD's video encoder and cross-modal attention (Lee et al., 2024).

3.1 ACTION HIERARCHY, ACTION TAXONOMY, AND PROBLEM SETUP

Action hierarchy. We represent an action by a temporal composition tree, also called its *definition tree*. The root represents the complete action, and each non-leaf node is composed of an ordered sequence of sub-actions $(s_1(c), \ldots, s_K(c))$, where c denotes the action class represented by the node and K is its number of immediate children. Each sub-action can be decomposed in the same way. We choose K separately for each dataset and at different definition tree levels, and ask the LLM to describe the action at that granularity (Section 3.2.2). We index the levels of the temporal tree by l, starting at $l = 1$ for the action root. The tree has up to D levels of sub-actions below it, indexed by $l = 2, \ldots, D + 1$. A *subtree* contains a node, its temporal descendants, and their connecting edges. An action instance occupies a timespan containing the ordered timespans of its sub-actions. Each node has a label, a text description, and attributes $R_c \subseteq R$ describing properties such as whether the action uses both hands. Here, R is the attribute vocabulary and $r \in R$ is an attribute. The shaded triple-jump subtree in Figure 1 contains the action node and its ordered sub-actions. The expanded landing node illustrates how a sub-action can be decomposed further.

Action taxonomy. A separate taxonomy groups related action classes into coarser categories. Its edges mean "is a kind of," not "consists of": triple jump is a kind of jumping, but triple jump is not a temporal part of jumping. In the definition tree, by contrast, hop is a temporal part of triple jump. The taxonomy has H levels indexed by h, with level 1 the coarsest. The evaluation protocol selects a taxonomy level h_d and asks the model to predict the classes at that level and their timespans. We call h_d the *detection level*. For a class c at this level, $\pi(c)$ is its immediate taxonomy parent and $\pi^{(h)}(c)$ is the class reached by following parent links to level $h < h_d$. We call supervision from these coarser classes *taxonomy supervision*. In the upper-right panel of Figure 1, long jump and triple jump belong to the same track-and-field jumping category, which can also belong to an even coarser category of horizontal jumps. These links express class membership. We use an existing dataset taxonomy when available, and otherwise manually define the class groupings (Appendix A.4).

Representation problem. Our goal is to learn hierarchical video representations that encode how actions are composed of ordered sub-actions and support recognition and localization beyond the training classes. Let $\mathcal{C}$ denote the set of action classes and V denote an untrimmed video containing instances of classes $c \in \mathcal{C}$. Let $\mathcal{T}_c^{\text{text}}$ denote the LLM-generated text definition tree of action class c. To represent actions within V, the model estimates the start and end times of a possible action at each temporal location n. It constructs a video tree $\mathcal{T}_n^{\text{video}}$ for this interval: the root represents the entire interval, and its descendants represent its ordered temporal parts, recursively. The model, with learned parameters θ, takes V and the action classes' text trees $\mathcal{T}_c^{\text{text}}$ as input and computes:

$$\left(V, \{\mathcal{T}_c^{\text{text}}\}_{c\in\mathcal{C}}\right) \xrightarrow{\theta} \left(\{\widetilde{\mathbf{f}}_n\}_n, \{\widehat{\mathbf{p}}(c)\}_{c\in\mathcal{C}}\right), \tag{1}$$

where $\widetilde{\mathbf{f}}_n$ is the *video subtree embedding* that combines features across $\mathcal{T}_n^{\text{video}}$, and $\widehat{\mathbf{p}}(c)$ is the *text subtree embedding* of $\mathcal{T}_c^{\text{text}}$. The collection $\{\widetilde{\mathbf{f}}_n\}_n$ forms the video representation. We learn these representations from action labels and boundaries, with additional classification supervision from coarser taxonomy labels. Sub-action labels and boundaries are not required.

Application: zero-shot TAL. We use zero-shot TAL to evaluate whether the representation supports recognition and localization beyond the training classes. Each action class at the selected taxonomy level h_d is the root of its own definition tree, and the coarser taxonomy levels $h < h_d$ provide

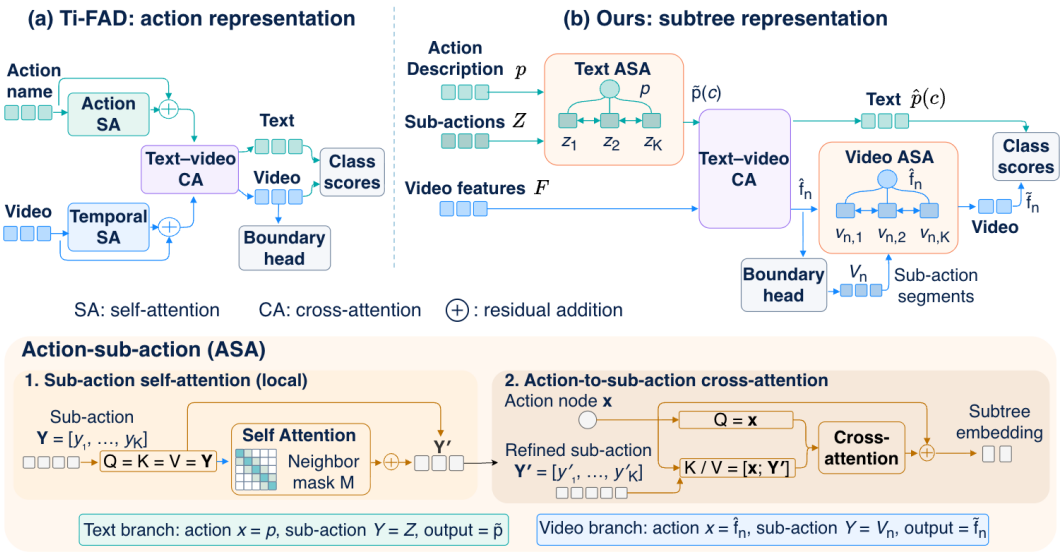

Figure 2: Subtree representation compared with Ti-FAD. (a) Ti-FAD connects action-name embeddings and video features through cross-modal attention. (b) Our text subtree embeddings guide video features through cross-modal attention, followed by Video ASA. Bottom: ASA combines neighbor-masked sub-action self-attention with action-to-sub-action cross-attention, using residual connections in both steps. The resulting video and text embeddings are used for classification.

taxonomy supervision. We split these action classes into disjoint sets $\mathcal{C}_{\text{seen}}$ and $\mathcal{C}_{\text{unseen}}$. Training uses action class labels from $\mathcal{C}_{\text{seen}}$, while evaluation targets $\mathcal{C}_{\text{unseen}}$, whose action class labels are excluded from training. Dataset-specific supervision is detailed in Section 4.1 and Appendix A.6. At inference, the trained model takes an untrimmed test video V and the unseen classes' text definition trees as inputs to Eq. 1. It compares the video subtree embedding of each estimated action interval with the text subtree embeddings to assign an action class and returns $\{(b_i, e_i, c_i, q_i)\}_{i=1}^{N_{\text{seg}}}$, $c_i \in \mathcal{C}_{\text{unseen}}$, where $b_i < e_i$ are start and end times, c_i the class, q_i the confidence, and N_{seg} the detection count.

3.2 Hierarchical Video Representation

The bottom panel of Figure 1 shows the video and text inputs used to construct subtree embeddings. Figure 2 details this computation, comparing Ti-FAD's action representation in panel (a) with our subtree representation in panel (b). Action–sub-action attention (ASA) combines an action embedding with its ordered sub-action embeddings in each modality. Text ASA precedes cross-modal attention, and Video ASA refines the resulting video features. The final text and video embeddings represent the action subtree and are used for classification.

3.2.1 Action–Sub-action Attention

As shown in the lower panel of Figure 2, ASA combines a parent embedding $\mathbf{x} \in \mathbb{R}^{1\times d}$ with its ordered child embeddings $\mathbf{Y} = [\mathbf{y}_1; \ldots; \mathbf{y}_K] \in \mathbb{R}^{K\times d}$, where d is the embedding dimension and a semicolon denotes stacking by rows. Both steps use multi-head attention (MHA) (Vaswani et al., 2017). We write $\mathrm{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}; M)$ for MHA with queries $\mathbf{Q}$, keys $\mathbf{K}$, values $\mathbf{V}$, and an optional additive mask M on the attention logits. ASA uses two MHA modules with separate learned parameters. $\mathrm{MHA}_{\text{sub}}$ performs *sub-action self-attention*: each child attends to itself and its immediate temporal neighbors. $\mathrm{MHA}_{\text{act}}$ performs *action-to-sub-action cross-attention*: the parent attends to itself and the refined children, without a mask:

$$\mathbf{Y}' = \mathbf{Y} + \mathrm{MHA}_{\text{sub}}(\mathbf{Y}, \mathbf{Y}, \mathbf{Y}; M), \qquad \mathrm{ASA}(\mathbf{x}, \mathbf{Y}) = \mathbf{x} + \mathrm{MHA}_{\text{act}}(\mathbf{x}, [\mathbf{x}; \mathbf{Y}'], [\mathbf{x}; \mathbf{Y}']) \,. \quad (2)$$

The $K \times K$ mask M restricts each child to itself and its immediate neighbors: $M_{kk'} = 0$ if $|k - k'| \le 1$, and $M_{kk'} = -\infty$ otherwise. Appendix A.1 gives the per-head form.

3.2.2 Subtree Embedding

Text branch. For class c, we describe the action node's label, taxonomy parent classes, and attributes in a sentence $\tau_0(c)$. We describe its ordered sub-actions in K sentences $\tau_1(c), \ldots, \tau_K(c)$, where

$\tau_k(c)$ describes $s_k(c)$ and $k > 0$. These sentences form the *hierarchical prompt* (HP). A frozen CLIP (Radford et al., 2021) text encoder ψ and a learned projection W_{text} map them to the action embedding $\mathbf{p}(c)$ and sub-action embeddings $\mathbf{z}_k(c)$:

$$\mathbf{p}(c) = \psi(\tau_0(c))\, W_{\text{text}}, \qquad \mathbf{z}_k(c) = \psi(\tau_k(c))\, W_{\text{text}}, \quad k = 1, \ldots, K. \tag{3}$$

We stack the sub-action embeddings in temporal order as $\mathbf{Z}(c) = [\mathbf{z}_1(c); \ldots; \mathbf{z}_K(c)] \in \mathbb{R}^{K \times d}$, with $\mathbf{p}(c), \mathbf{z}_k(c) \in \mathbb{R}^{1 \times d}$. Text ASA in Figure 2(b) combines $\mathbf{p}(c)$ with $\mathbf{Z}(c)$ to produce the text subtree embedding $\widetilde{\mathbf{p}}(c) = \text{ASA}(\mathbf{p}(c), \mathbf{Z}(c))$. When a sub-action has sub-actions of its own, it is first composed with them in the same way, and the result replaces $\mathbf{z}_k(c)$. The same encoding and attention operations apply to seen and unseen action classes.

Video branch. The video branch computes features in three stages. First, a video backbone and a convolutional transformer produce a temporal feature pyramid $\mathbf{F}$ (Zhang et al., 2022), with one feature per temporal location. Second, Ti-FAD's text–video cross-attention updates these features using the text subtree embeddings $\widetilde{\mathbf{P}}$ from the text branch, producing the updated features $\widehat{\mathbf{F}}$. We describe this step under *Connection through cross-modal attention* below. We write $\widehat{\mathbf{f}}_n \in \mathbb{R}^{1 \times d}$ for the updated feature at location n. Third, Video ASA combines the action feature with features of its ordered temporal sub-actions to form a video subtree embedding. The feature $\widehat{\mathbf{f}}_n$ serves as the parent embedding. The boundary head estimates the start and end times $b_n < e_n$ of an action at location n. The interval $[b_n, e_n]$ forms the root of $\mathcal{T}_n^{\text{video}}$. We divide it into K equal-length temporal chunks to obtain $\mathbf{v}_{n,k} \in \mathbb{R}^{1 \times d}$. Stacking the children in temporal order gives $\mathbf{V}_n = [\mathbf{v}_{n,1}; \ldots; \mathbf{v}_{n,K}] \in \mathbb{R}^{K \times d}$, and Video ASA in Figure 2(b) produces the video subtree embedding $\widetilde{\mathbf{f}}_n = \text{ASA}(\widehat{\mathbf{f}}_n, \mathbf{V}_n)$. The estimated action interval thus provides the child embeddings without annotated sub-action boundaries. For deeper subtrees, each part is split again into K' equal-length parts and composed in the same way, before the K resulting embeddings are composed with $\widehat{\mathbf{f}}_n$. Detailed procedure on how K and K' are chosen is in Appendix A.6.

Connection through cross-modal attention. Ti-FAD's bidirectional cross-modal attention (CA) connects both branches (Figure 2(b)). It maps $\widetilde{\mathbf{P}}$ and $\mathbf{F}$ to updated text embeddings $\widehat{\mathbf{P}}$ and video features $\widehat{\mathbf{F}}$. The updated video features feed Video ASA.

Subtree embedding and prediction. The final text embedding $\widehat{\mathbf{p}}(c)$ and video embedding $\widetilde{\mathbf{f}}_n$ represent the action subtree in their respective modalities. Their scaled cosine similarity gives the score for class c at location n:

$$u_{n,c} = \kappa \cos\big(\widetilde{\mathbf{f}}_n, \widehat{\mathbf{p}}(c)\big) + \beta, \tag{4}$$

with a learned scale κ and bias β. We obtain the score of a coarser taxonomy class g at level $h < h_d$ by aggregating scores of the detection classes assigned to it. Let $\mathcal{C}^{(h)}(g) = \{c : \pi^{(h)}(c) = g\}$ denote these member classes among the candidate actions. The taxonomy score is

$$u_{n,g}^{(h)} = \log \sum_{c \in \mathcal{C}^{(h)}(g)} \exp(u_{n,c}) \tag{5}$$

The log-sum-exp acts as a smooth maximum, giving a taxonomy parent class a high score when one or more of its member classes receive high scores. This aggregation follows taxonomy membership, independently of the ordered sub-actions combined by ASA.

3.3 Training with Taxonomy Supervision

The Training block in Figure 1 combines action and taxonomy supervision. Action supervision includes classification and temporal localization at the selected detection level. Taxonomy supervision adds a classification loss for the coarser taxonomy categories. Their scores are aggregated from action scores (Eq. 5), so the taxonomy loss also updates the representation used for action prediction. Temporal boundaries are supervised by the action detection loss. Neither source of supervision requires labels or boundaries for the temporal sub-actions.

Action supervision. We use Ti-FAD's detection objective:

$$\mathcal{L}_{\text{det}} = \alpha_{\text{cls}}\, \mathcal{L}_{\text{cls}} + \alpha_{\text{reg}}\, \mathcal{L}_{\text{reg}} + \alpha_{\text{fg}}\, \mathcal{L}_{\text{fg}} + \alpha_{\text{ctr}}\, \mathcal{L}_{\text{ctr}}, \tag{6}$$

where $\mathcal{L}_{\text{cls}}$ supervises action classification from the scores $u_{n,c}$ of Eq. 4; $\mathcal{L}_{\text{reg}}$ supervises regression of the segment boundaries b_n and e_n; $\mathcal{L}_{\text{fg}}$ supervises foreground prediction, that is, whether location

n lies inside an action or in the background; and $\mathcal{L}_{\text{ctr}}$ supervises centerness, that is, how close location n is to the center of its action segment. The coefficients $\alpha_{\text{cls}}, \alpha_{\text{reg}}, \alpha_{\text{fg}}, \alpha_{\text{ctr}}$ weight these terms.

Taxonomy supervision (TS). A training segment labeled *long jump* also belongs to the taxonomy category *track-and-field jumping*. The taxonomy loss trains the representation to recognize this category, which long jump shares with other jumping classes. On THUMOS14 and ActivityNet, training aggregates scores over seen member classes only and unseen classes are excluded, so $\mathcal{C}^{(h)}_{\text{seen}}(g) = \{c \in \mathcal{C}_{\text{seen}} : \pi^{(h)}(c) = g\}$ where $\pi^{(h)}(c)$ is the taxonomy category containing action class c at level h. In the coarse-supervision setting on FineGym and FineDiving, we instead aggregate over $\mathcal{C}^{(h)}(g)$, including both seen and unseen member classes. Their scores contribute jointly to the taxonomy prediction, so unseen classes receive gradients through their coarser categories, without direct action-class targets (Appendix A.6). At each supervised taxonomy level $h < h_d$, we compare the category scores $u^{(h)}_{n,g}$ with the taxonomy labels using sigmoid focal loss (FL) (Lin et al., 2017):

$$\mathcal{L}^{(h)}_{\text{tax}} = \frac{1}{N_{\text{pos}}} \sum_{n} \sum_{g} \text{FL}\big(u^{(h)}_{n,g},\, y^{(h)}_{n,g}\big), \tag{7}$$

where $y^{(h)}_{n,g} = 1$ if the instance annotated at location n belongs to taxonomy category g, and 0 otherwise. N_{pos} is an exponential moving average of the number of positive locations used for boundary regression, shared with the detection loss.

Uncertainty weighting across levels. We use uncertainty weighting (Kendall et al., 2018) to learn the contribution of each supervised taxonomy level. We assign a trainable log-variance parameter to each supervised loss: ρ_{det} for the detection level h_d, and ρ_h for each coarser taxonomy level $h < h_d$. A loss with log-variance ρ receives the weight $e^{-\rho}$, so a larger ρ lowers its weight. The fixed coefficient λ_{tax} sets the overall scale of taxonomy supervision. The training objective is

$$\mathcal{L} = e^{-\rho_{\text{det}}}\, \mathcal{L}_{\text{det}} + \lambda_{\text{tax}} \sum_{h=1}^{h_d-1} e^{-\rho_h}\, \mathcal{L}^{(h)}_{\text{tax}} + \frac{1}{2}\Big(\rho_{\text{det}} + \sum_{h=1}^{h_d-1} \rho_h\Big), \tag{8}$$

where $\mathcal{L}_{\text{det}}$ is the detection loss (Eq. 6) and $\mathcal{L}^{(h)}_{\text{tax}}$ is the taxonomy loss at level h (Eq. 7). The learned weight of taxonomy level h is $\lambda_{\text{tax}} e^{-\rho_h}$. The final term keeps the weights from collapsing to zero, since it grows as the ρ values increase. Each additional taxonomy level adds one loss and one learned weight. Appendix A.2 describes the sequential training of the text and video instances of ASA.

3.4 Inference

At inference, the detector uses the action scores from Eq. 4 and the predicted boundaries to produce segments at taxonomy level h_d, as illustrated by the unseen triple-jump example in Figure 1. We then apply soft non-maximum suppression (Bodla et al., 2017).

Attribute similarity prior (ASP). Shared attributes from the LLM descriptions, such as a running approach or repeated movements, can indicate how alike an unseen class is to each seen class. At inference, we compute the *attribute-weighted adjustment* as:

$$q_{\text{ASP}} = q \exp\Big(-\frac{\gamma}{2}\eta_c(\delta)^2\Big), \qquad \eta_c(\delta) = \frac{\log \delta - \mu_c}{\widehat{\sigma}_c}, \tag{9}$$

where q and q_{ASP} are the confidence of a predicted segment $[b, e]$ of unseen class c before and after adjustment, $\delta = e - b$ is the segment length, μ_c and $\widehat{\sigma}_c$ the transferred log-duration mean and standard deviation, and $\gamma \geq 0$ the adjustment strength.

4 Experiments

We evaluate the learned video representation by its temporal ordering and taxonomy grouping (Sec. 4.2), followed by localization under strict zero-shot and coarse-supervision protocols (Sec. 4.3).

4.1 Experimental Setup

We evaluate representations on THUMOS14 (Jiang et al., 2014), and strict zero-shot localization on THUMOS14 and ActivityNet v1.3 (Heilbron et al., 2015), excluding annotations of unseen action

Table 1: Representation validation on THUMOS14 (75/25, splits 0–2) using learned video representations (App. A.5). Higher scores indicate better temporal ordering and taxonomy grouping.

Representation	Kendall Rank Correlation Coefficient		Silhouette Coefficient
	Seen ↑	Unseen ↑	Unseen ↑
Ti-FAD	0.43	0.24	0.11
PDA (adapted)	0.39	0.22	−0.05
Ours	**0.43**	**0.28**	**0.13**

instances, including their taxonomy labels, from training under the 75/25 and 50/50 protocols of Ju et al. (2022). FineGym (Shao et al., 2020) and FineDiving (Xu et al., 2022) test deeper composition under *coarse supervision*: training withholds unseen action-class labels but retains the instances' taxonomy labels and class-agnostic temporal boundaries. This setting tests whether the representation distinguishes fine-grained actions when only coarser labels and temporal boundaries are available for those classes. Appendices A.4–A.7 give splits, hierarchies, and evaluation details. We measure temporal ordering with the Kendall rank correlation coefficient (Dwibedi et al., 2019), taxonomy grouping with the silhouette coefficient (Rousseeuw, 1987), and TAL with mean average precision (mAP), reported separately for unseen and seen classes. On THUMOS14 and ActivityNet, we build on Ti-FAD with two-stream I3D features (Carreira & Zisserman, 2017), a frozen CLIP-B/16 text encoder, and its external-classifier fusion. Qwen2.5-VL-7B-Instruct (Bai et al., 2025) generates descriptions before training. Appendix A.4 gives the implementation settings. We adapt DFAlign and PDA to the same Ti-FAD base, using the same encoders, external-classifier fusion, and description generator as our method. Each method retains its description format within this shared setup. Appendix A.9 details the adaptations and seven PDA configurations.

4.2 Representation Validation

We compare our learned video representation with Ti-FAD and our PDA adaptation (Table 1, Appendix A.5). On unseen actions, our Kendall rank correlation reaches 0.28, compared with 0.24 for Ti-FAD and 0.22 for PDA, while matching Ti-FAD on seen actions (0.43), indicating that our representation preserves temporal progression beyond the training classes. Our silhouette coefficient of 0.13, compared with 0.11 for Ti-FAD and -0.05 for PDA, shows more coherent grouping of unseen actions by taxonomy category. Our representation thus captures both the progression within an action and the relationships between actions, without sub-action labels or boundaries.

4.3 Zero-Shot Temporal Action Localization

Comparison with Existing Methods. With the same encoders and detection architecture, our method outperforms Ti-FAD and our DFAlign and PDA adaptations in all four settings (Table 2). On THUMOS14, it reaches 51.00 and 45.66 mAP under the 75/25 and 50/50 splits, compared with 48.1 and 43.2 for DFAlign (strongest adapted comparator). The relative gain over Ti-FAD is larger when fewer classes are seen during training, on both THUMOS14 (10.8% at 50/50 vs. 9.0% at 75/25) and ActivityNet (3.9% vs. 2.3%). Since DFAlign and PDA already go beyond action names, these comparisons assess our method against alternatives using richer descriptions (Appendix A.9).

Ablation Study. Table 3 shows whether the gains come from the hierarchical descriptions themselves or from learning to compose them. Averaging the hierarchical prompt (HP) embeddings improves over the action name alone (48.83 to 49.73 mAP), and ASA raises mAP further to 51.00, showing that descriptions help more when the model learns how an action relates to its temporal parts. Removing TS (taxonomy supervision) lowers mAP to 49.67. ASP adds a smaller gain (50.78 to 51.00).

Recursive Composition on Hierarchical Datasets. Under coarse (taxonomy) supervision (Section 4.1), we evaluate FineGym actions ($h_d = 3$) and phrases ($h_d = 2$), with one and two sub-action levels, and FineDiving actions ($h_d = 2$), with two levels (Table 4). The full method outperforms Ti-FAD without taxonomy supervision in every setting, raising unseen-class mAP from 11.0 to 26.0 and from 7.1 to 28.7 on FineGym actions and phrases, and from 29.6 to 54.7 on FineDiving. Seen-class gains are smaller, so the improvement comes mainly from transfer to unseen classes, and it is largest for FineGym phrases, whose definition trees are deepest.

Table 2: Unseen-action mAP (%). Columns show seen/unseen class proportions. Blocks separate published results, adaptations to a common Ti-FAD base, and other evaluation settings. Adaptations share our encoders, fusion, and description generator (Appendix A.9).

Method	Reference	THUMOS14 75/25	THUMOS14 50/50	ActivityNet 75/25	ActivityNet 50/50
Published reference results					
EffPrompt	Ju et al. ECCV'22	23.3	21.9	23.1	19.6
STALE	Nag et al. ECCV'22	23.8	22.2	24.9	20.5
DeTAL	Li et al. TPAMI'24	25.3	24.1	25.8	22.4
mProTEA	Raza et al. TCSVT'24	27.9	26.1	27.6	25.6
ZEETAD	Phan et al. WACV'24	43.2	30.2	32.5	24.9
OVFormer†	Gupta et al. BMVC'24	35.3	30.5	29.5	27.2
STOV-TAL‡	Hyun et al. WACV'25	34.5	31.5	30.5	28.0
Controlled comparison on a common Ti-FAD base					
DFAlign (adapted)	Zhu et al. SIGIR'26	48.1	43.2	34.4	31.8
PDA (adapted)	Zhu et al. CVPR'26	46.4	40.4	33.8	32.0
Ti-FAD (baseline)	Lee et al. NeurIPS'24	46.8	41.2	34.7	32.0
Ours	–	**51.00**	**45.66**	**35.51**	**33.25**
Relative gain over Ti-FAD		+9.0%	+10.8%	+2.3%	+3.9%
Different evaluation settings					
TF-CADE (no external classifier)	Lee et al. CVPR'26	34.5	21.1	17.2	10.5
Bi-TSA (generalized zero-shot)	Tan et al. TCSVT'26	29.8	22.1	27.1	26.4
T3AL (training-free)	Liberatori et al. CVPR'24	9.2	10.4	15.4	14.3
LLT (training-free)	Jiang et al. CVPR'26	12.8	12.6	17.1	15.7
FreeZAD (training-free)	Han et al. TMM'26	10.0	9.9	18.3	18.7

Table 3: THUMOS14 component ablations: unseen mAP (%) over ten 75/25 splits and tIoU 0.3–0.7, with external-classifier fusion (Appendix A.4).

HP	ASA	TS	ASP	mAP
✓	✓	✓	✓	**51.00**
✓	✓	✓	×	50.78
✓	✓	×	✓	49.67
✓	×	✓	✓	49.73
×	×	✓	✓	48.83

Table 4: Coarse-supervision mAP (%) versus Ti-FAD without TS (Section 4.1). FineGym evaluates actions ($h_d = 3$) and phrases ($h_d = 2$), and FineDiving evaluates actions. Δ%: relative gain. Table 7 holds TS constant.

Dataset	h_d	Method	Unseen	Δ%	Seen	Δ%
FineGym	3	Ti-FAD	11.0	+136.36%	55.5	+5.59%
		Ours	**26.0**		**58.6**	
	2	Ti-FAD	7.1	+304.23%	61.4	+0.49%
		Ours	**28.7**		**61.7**	
FineDiving	2	Ti-FAD	29.6	+84.80%	63.3	+18.33%
		Ours	**54.7**		**74.9**	

5 Conclusions and Limitations

We presented hierarchical video representations that capture shared sub-actions and their composition into different complex actions. LLM-generated text trees guide the construction of corresponding video trees without requiring explicit sub-action annotations. ASA lets neighboring sub-actions exchange information before combining them with their parent in both trees. Taxonomy supervision also draws related actions together under shared categories. Learned from seen actions, these representations better preserve temporal order and taxonomy grouping for unseen actions. They improve recognition and localization under strict zero-shot protocols on THUMOS14 and ActivityNet, with further localization gains under coarse supervision on FineGym and FineDiving.

Limitations and Future Work: (1) Our current design divides each action interval into equal-length sub-actions, which simplifies alignment but does not hold when sub-actions have uneven spans; ASA partially compensates for this, since each sub-action attends to its neighbors across misplaced boundaries. (2) The LLM-generated text trees are not verified by humans, so their quality depends on the language model. (3) Our evaluation is for up to two sub-action levels and assesses the learned representations rather than individual sub-action segments or names. Learning more precise sub-action boundaries from video, comparing generated trees with human decompositions, and evaluating deeper trees and sub-action predictions directly are candidates for future work.

AI USE STATEMENT

We used generative AI tools for grammar correction, writing polishing, exhaustive finding of related work, and assistance with implementing selected functions. The research ideas and method designs were developed by the authors, and the final manuscript was drafted by the authors. We manually reviewed and tested all AI-generated code to check for implementation correctness. We also used a language model to generate the action descriptions used by our method, as described in Section 4.1. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We will release the codebase upon acceptance. Section 3 describes the hierarchy, attention modules, training losses, and inference procedure. Section 4.1 specifies the datasets, evaluation protocols, and metrics. Appendices A.1–A.4 provide the attention equations, training schedule, hyperparameters, ablation settings, and duration-transfer calculations. Appendices A.5 and A.6 detail the representation evaluation and the FineGym hierarchy, class splits, and implementation.

REFERENCES

Kumar Ashutosh, Rohit Girdhar, Lorenzo Torresani, and Kristen Grauman. HierVL: Learning hierarchical video-language embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23066–23078, 2023. 3

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. 8

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 813–824. PMLR, 2021. 3

Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S. Davis. Soft-NMS – improving object detection with one line of code. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 7

João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the Kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6299–6308, 2017. 3, 8

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal cycle-consistency learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1801–1810, 2019. 8, 15

Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6824–6835, 2021. 3

Reza Ghoddoosian, Saif Sayed, and Vassilis Athitsos. Hierarchical modeling for task recognition and action segmentation in weakly-labeled instructional videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022. 3

Akshita Gupta, Aditya Arora, Sanath Narayan, Salman Khan, Fahad Shahbaz Khan, and Graham W. Taylor. Open-vocabulary temporal action localization using multimodal guidance. In *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA, 2024. 9, 18

Chaolei Han, Hongsong Wang, Jidong Kuang, Lei Zhang, and Jie Gui. Training-free zero-shot temporal action detection with vision-language models. *IEEE Transactions on Multimedia*, 2026. doi: 10.1109/TMM.2026.3701531. 3, 9

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 961–970, 2015. 3, 7

Jeongseok Hyun, Su Ho Han, Hyolim Kang, Joon-Young Lee, and Seon Joo Kim. Exploring scalability of self-training for open-vocabulary temporal action localization. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 9388–9397, 2025. 3, 9

Han Jiang, Haoyu Tang, Xiaoxuan Mu, Chen Li, and Jihua Zhu. Memory matters: Boosting training-free zero-shot temporal action localization with a learnable lookup table. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9762–9772, 2026. 9

Yu-Gang Jiang, Jingen Liu, Amir Roshan Zamir, George Toderici, Ivan Laptev, Mubarak Shah, and Rahul Sukthankar. THUMOS challenge: Action recognition with a large number of classes. Challenge website, 2014. 3, 7

Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *Computer Vision – ECCV 2022*, pp. 105–124, 2022. 8, 9

Chen Ju, Zeqian Li, Peisen Zhao, Ya Zhang, Xiaopeng Zhang, Qi Tian, Yanfeng Wang, and Weidi Xie. Multi-modal prompting for low-shot temporal action localization. *arXiv preprint arXiv:2303.11732*, 2023. 3

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7482–7491, 2018. 7

Yearang Lee, Ho-Joong Kim, and Seong-Whan Lee. Text-infused attention and foreground-aware modeling for zero-shot temporal action detection. In *Advances in Neural Information Processing Systems*, volume 37, pp. 9864–9884, 2024. doi: 10.52202/079017-0316. 3, 4, 9

Yearang Lee, Ho-Joong Kim, and Seong-Whan Lee. TF-CADE: Foreground-concentrated text-video alignment for zero-shot temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2843–2852, 2026. 3, 9

Zhiheng Li, Yujie Zhong, Ran Song, Tianjiao Li, Lin Ma, and Wei Zhang. DeTAL: Open-vocabulary temporal action localization with decoupled networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):7728–7741, 2024. doi: 10.1109/TPAMI.2024.3395778. 9

Benedetta Liberatori, Alessandro Conti, Paolo Rota, Yiming Wang, and Elisa Ricci. Test-time zero-shot temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18720–18729, 2024. 9

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017. 7

Sauradip Nag, Xiatian Zhu, Yi-Zhe Song, and Tao Xiang. Zero-shot temporal action detection via vision-language prompting. In *Computer Vision – ECCV 2022*, pp. 681–697, 2022. doi: 10.1007/978-3-031-20062-5_39. 3, 9

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision, 2023. 18

Thinh Phan, Khoa Vo, Duy Le, Gianfranco Doretto, Donald Adjeroh, and Ngan Le. ZEETAD: Adapting pretrained vision-language model for zero-shot end-to-end temporal action detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 7046–7055, 2024. 9

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. 6

Asif Raza, Bang Yang, and Yuexian Zou. Zero-shot temporal action detection by learning multimodal prompts and text-enhanced actionness. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(11):11000–11012, 2024. doi: 10.1109/TCSVT.2024.3414275. 9

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. 8, 15

Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. FineGym: A hierarchical video dataset for fine-grained action understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2616–2625, 2020. 3, 8

Mingkui Tan, Yihao Qian, Yirui Wang, Runhao Zeng, Victor C. M. Leung, and Xiping Hu. Bidirectional temporal-sensitive adaptation for generalized zero-shot temporal action localization. *IEEE Transactions on Circuits and Systems for Video Technology*, 36(6):7600–7614, 2026. doi: 10.1109/TCSVT.2026.3664694. 3, 9

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. 5, 13

Jinglin Xu, Yongming Rao, Xumin Yu, Guangyi Chen, Jie Zhou, and Jiwen Lu. FineDiving: A fine-grained dataset for procedure-aware action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 8, 16

Gefan Ye, Lin Li, Kexin Li, Jun Xiao, and Long Chen. Zero-shot compositional action recognition with neural logic constraints. In *Proceedings of the ACM International Conference on Multimedia*, 2025. 3

Chen-Lin Zhang, Jianxin Wu, and Yin Li. ActionFormer: Localizing moments of actions with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 492–510, 2022. 6

Sa Zhu, Wanqian Zhang, Lin Wang, Xiaohua Chen, Chenxu Cui, Jinchao Zhang, and Bo Li. Decompose and transfer: CoT-prompting enhanced alignment for open-vocabulary temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026a. 3, 9

Sa Zhu, Wanqian Zhang, Lin Wang, Jinchao Zhang, Cong Wang, and Bo Li. Denoise and align: Diffusion-driven foreground knowledge prompting for open-vocabulary temporal action detection. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 2026b. 3, 9

A Appendix

A.1 Multi-Head Attention in ASA

The sub-action self-attention in ASA (Eq. 2) uses multi-head attention (Vaswani et al., 2017) with the neighbor mask M defined in Section 3.2.1. For head j with feature dimension d_j, the output is

$$\mathbf{H}_j = \mathrm{softmax}\left(\frac{(\mathbf{Y}W_j^{\mathrm{Q}})(\mathbf{Y}W_j^{\mathrm{K}})^{\mathsf{T}}}{\sqrt{d_j}} + M\right)\mathbf{Y}W_j^{\mathrm{V}}, \tag{10}$$

where $W_j^{\mathrm{Q}}, W_j^{\mathrm{K}}, W_j^{\mathrm{V}} \in \mathbb{R}^{d\times d_j}$ are learned projections. The softmax runs across keys in each row, and the mask assigns zero weight to non-neighboring sub-actions. Concatenating the head outputs and applying an output projection gives the $\mathrm{MHA}_{\mathrm{sub}}$ term of Eq. 2. The action-to-sub-action cross-attention $\mathrm{MHA}_{\mathrm{act}}$ follows the same per-head form, with $\mathbf{x}$ as the query and $[\mathbf{x}; \mathbf{Y}']$ as the keys and values, and no mask. The text and video branches use separate parameters for $\mathrm{MHA}_{\mathrm{sub}}$ and $\mathrm{MHA}_{\mathrm{act}}$. Within each branch, these parameters are shared across seen and unseen actions.

A.2 Training Schedule

We train the text and video instances of ASA in sequence. We first train the detector's temporal self-attention and cross-modal attention layers, text projection, and action-to-sub-action cross-attention in the text branch, using $\mathbf{Y}' = \mathbf{Y}$. We then freeze these parameters and train the text branch's sub-action self-attention. Finally, we train the ASA block in the video branch on the frozen detector. Both ASA instances are used at inference. Each new attention layer's output projection and bias are initialized to zero, so its initial residual update preserves the input embedding. Appendix A.4 gives the prompt-masking and substitution settings used during training.

A.3 Attribute-Similarity Prior: Weighting and Moment Matching

Section 3.4 transfers duration statistics from seen actions to an unseen action c by attribute overlap. For each seen action c', we measure training-segment durations in seconds and estimate their log-duration mean $\mu_{c'}$ and variance $\sigma_{c'}^2$. The transfer weight of c' is its attribute similarity to c, measured by the Jaccard index $J_{cc'}$ of their attribute sets $R_c, R_{c'}$ and normalized over the seen actions $\mathcal{C}_{\mathrm{seen}}$:

$$J_{cc'} = \frac{|R_c \cap R_{c'}|}{|R_c \cup R_{c'}|}, \qquad w_{cc'} = \frac{J_{cc'}}{\sum_{c''\in\mathcal{C}_{\mathrm{seen}}} J_{cc''}}. \tag{11}$$

We set $J_{cc'} = 0$ if both attribute sets are empty. If no seen action shares an attribute with c, or attributes are unavailable, the weights are uniform. The transferred mean and variance in log-duration space are

$$\mu_c = \sum_{c'\in\mathcal{C}_{\mathrm{seen}}} w_{cc'}\mu_{c'}, \qquad \sigma_c^2 = \sum_{c'\in\mathcal{C}_{\mathrm{seen}}} w_{cc'}\left(\sigma_{c'}^2 + \mu_{c'}^2\right) - \mu_c^2, \tag{12}$$

which match the first two moments of the attribute-weighted mixture of seen-class log-duration distributions. Attribute similarity determines the transfer weights, and the segment duration determines the score adjustment. For numerical stability, we use $\widehat{\sigma}_c = \max(\sqrt{\max(\sigma_c^2, 0)}, 0.15)$ in Eq. 9.

A.4 Experimental Implementation Details

Evaluation settings. Strict zero-shot results on THUMOS14 and ActivityNet average unseen-action mAP over ten random class splits for each seen/unseen proportion, 75/25 and 50/50. We average mAP over tIoU thresholds 0.3–0.7 in steps of 0.1 on THUMOS14, FineGym, and FineDiving, and 0.5–0.95 in steps of 0.05 on ActivityNet. All three evaluation metrics are higher-is-better. Representation validation uses THUMOS14 75/25 splits 0–2 (Appendix A.5). FineGym and FineDiving use three splits per setting (Appendices A.6 and A.7). Qwen2.5-VL-7B-Instruct receives text-only input when generating descriptions.

THUMOS14 and ActivityNet use a two-level taxonomy of activity categories and action classes, with detection at $h_d = 2$. Each action roots a temporal tree containing one level of sub-actions ($D = 1$). FineGym provides a three-level activity–phrase–action taxonomy. We use $D = 1$ for action

detection at $h_d = 3$ and $D = 2$ for phrase detection at $h_d = 2$ (Appendix A.6). FineDiving uses one taxonomy level above its action classes ($H = 2$, $h_d = 2$) and two temporal sub-action levels ($D = 2$, Appendix A.7).

Taxonomy groupings. Neither THUMOS14 nor ActivityNet provides a class hierarchy, so we define the activity level ourselves. On THUMOS14, the twenty classes form nine activities: aquatic dive (cliff diving, diving), ball throw (baseball pitch, cricket bowling), court jump strike (basketball dunk, volleyball spiking), field throw (hammer throw, javelin throw, shot put, discus throw), implement strike (billiards, cricket shot, golf swing, tennis swing), jump field (high jump, long jump, pole vault), and three single-class activities (clean and jerk, frisbee catch, soccer penalty). On ActivityNet, we group the 200 classes into 39 activities by the kind of activity performed (for example, grooming, precision target sports, or tool craft and repair). The taxonomy parent name enters each class description ("a kind of ball throw"), and the taxonomy loss uses these groupings. The groupings, the FineGym attribute tables, and all generated descriptions will be released with the code.

Prompts and attention. A learned projection maps the frozen text encoder's outputs to the detector's embedding dimension. Each ASA attention layer uses four heads. We choose the number of children at each temporal level to set the decomposition granularity, informed by dataset statistics. The LLM generates that many ordered descriptions, and the video branch uses the same number of temporal chunks. Thus, a child count specifies the resolution of the decomposition, while the descriptions specify the constituent actions and their sequence.

THUMOS14 and ActivityNet use ten sub-actions below each action root. FineGym action detection uses ten phases per action ($D = 1$). FineGym phrase detection uses three action children, each with ten phase children ($D = 2$). FineDiving action detection uses three sub-actions, each with five phase children ($D = 2$). Its depth ablation retains both Text and Video ASA but removes the phase level for $D = 1$. Within each setting, classes share these per-level counts. Their descriptions and the corresponding video features differ. Each temporal level has separate ASA parameters in each modality.

Training settings. The detection-loss weights α_{cls}, α_{reg}, α_{fg}, and α_{ctr} are 0.5, 1, 1, and 1, respectively. THUMOS14 supervises one taxonomy parent level. FineGym supervises two taxonomy parent levels at $h_d = 3$ and one at $h_d = 2$ (Appendix A.6). During training, each sub-action embedding is randomly masked in action cross-attention with probability 0.3. We also replace the full hierarchical prompt with the immediate taxonomy parent's description with probability 0.3. These operations train the model to recognize actions from incomplete descriptions or descriptions of their taxonomy parents. At inference, all prompt sentences are used.

Ablation settings. We evaluate the component ablations on THUMOS14 under the 75/25 protocol, averaged over ten splits with external-classifier fusion. Removing ASA replaces text composition with the mean of the eleven sentence embeddings and scores each video location from its cross-modal attention output $\widehat{\mathbf{f}}_n$ without the ASA update. Removing TS retains HP and ASA, and removing ASP retains external-classifier fusion. The action-name variant uses a single action-name prompt in place of HP and ASA.

Duration rescoring. We use attribute-based transfer with $\gamma = 0.3$ on THUMOS14 and uniform transfer with $\gamma = 0.15$ on ActivityNet. Appendix A.3 gives the transfer equations.

Text and video instances of ASA. Table 3 removes both ASA instances together. Table 5 isolates the video instance by comparing Text ASA with Text and Video ASA at the same temporal-tree depth. Adding Video ASA improves unseen and seen mAP in all six settings. The largest relative unseen-class gains occur on FineGym phrases and FineDiving, where the trees contain two sub-action levels, showing the value of video composition beyond the text subtree embeddings.

Learned level weights. The uncertainty weights of Eq. 8 converge to $e^{-\rho_{\text{det}}} \approx 1.00$ for the detection loss and $e^{-\rho_h} \approx 1.07$–1.10 for the taxonomy loss on THUMOS14 (splits 0–2), and to 1.27 for the detection loss, 1.55 for the phrase level, and 1.54 for the activity level on FineGym at $h_d = 3$. These learned multipliers describe the balance between losses in each run. The taxonomy terms also include the coefficient λ_{tax} in Eq. 8.

Table 5: Text ASA versus Text and Video ASA at the same tree depth. Entries report unseen mAP, with seen mAP in parentheses. THUMOS14 and ActivityNet use ten splits per protocol. FineGym evaluates phrases ($h_d = 2$, $D = 2$), and FineDiving evaluates dives ($h_d = 2$, $D = 2$), under coarse supervision. Gains are relative percentages.

Dataset / split	Text ASA	Text + Video ASA	Relative gain
THUMOS14 75/25	50.92 (59.92)	**51.00 (60.50)**	+0.16% (+0.97%)
THUMOS14 50/50	45.40 (62.13)	**45.66 (62.59)**	+0.57% (+0.74%)
ActivityNet 75/25	35.38 (35.86)	**35.51 (35.98)**	+0.37% (+0.33%)
ActivityNet 50/50	33.11 (35.55)	**33.25 (35.77)**	+0.42% (+0.62%)
FineGym phrases	22.8 (57.9)	**28.7 (61.7)**	+25.88% (+6.56%)
FineDiving dives	48.3 (69.1)	**54.7 (74.9)**	+13.25% (+8.39%)

A.5 Representation Validation Details

We evaluate the learned video representations on THUMOS14 test videos under the 75/25 protocol, using splits 0–2. We compare our representation with those of Ti-FAD and our PDA adaptation (Section 4.2). For our model, we extract the cross-modal attention output $\widehat{\mathbf{f}}_n$ before Video ASA, giving one learned feature per temporal location. The model parameters used to extract these representations are not updated during representation evaluation. The detector's final classification scores use the video subtree embedding $\widetilde{\mathbf{f}}_n$ produced by Video ASA (Eq. 4). We report both metrics on unseen actions and additionally report Kendall rank correlation on seen actions.

Temporal ordering. We train a linear regressor on features from training videos to predict each location's normalized position within its annotated action segment. On held-out segments, we compute the Kendall rank correlation coefficient between the predicted positions and the true temporal order (Dwibedi et al., 2019). This evaluation uses action start and end times to determine relative positions within a segment.

Taxonomy grouping. At each unseen-class location, we assign the extracted feature to the immediate parent category of its ground-truth action class in the action taxonomy. We compute the silhouette coefficient (Rousseeuw, 1987) using cosine distance. Higher values indicate that features are closer within taxonomy groups than across groups. PDA's negative coefficient indicates that these taxonomy groups overlap in its feature space.

Additional temporal-ordering checks. A regressor trained on shuffled positions reaches a Kendall rank correlation coefficient of 0.05, supporting the presence of temporal-order information in all three representations. Training our representation on all twenty action classes yields 0.44 on held-out videos, close to 0.43 on the fifteen seen classes in the zero-shot setting. The improvement on unseen actions thus accompanies retained temporal-order information on seen actions.

A.6 FineGym Taxonomy, Temporal Composition, and Implementation

Taxonomy and class splits. FineGym annotates apparatus categories, element sets, and gymnastics elements, which we call activities, phrases, and actions. These form a three-level taxonomy ($H = 3$): four activities at level 1, fourteen phrases at level 2, and ninety-nine actions at level 3. The taxonomy records class membership. Temporal phases belong to the generated definition trees, not to an additional taxonomy level. At $h_d = 3$, each detection class has phrase and activity categories for taxonomy supervision and a temporal tree with ten phase sub-actions ($D = 1$). We hold out twenty of the ninety-nine action classes and train detection on the remaining seventy-nine. At $h_d = 2$, each detection class has an activity category for taxonomy supervision and a temporal tree with ordered actions and their phases ($D = 2$). We hold out four of the fourteen phrases and train detection on the remaining ten. Each setting uses three random seen/unseen class splits and we report the mean over splits.

Coarse-supervision protocol. FineGym and FineDiving retain seen and unseen instances in the training videos under coarse supervision. Action-classification loss excludes both the unseen instances and the unseen-class output columns, including negative class targets. Class-agnostic boundary and foreground supervision use all action instances. Our taxonomy loss uses their coarser category

labels. In Eq. 5, each category score aggregates all its member classes, including unseen classes. The taxonomy loss therefore updates unseen-class scores through category membership, without assigning unseen action-class targets. The temporal sub-actions below the selected detection class have no label or boundary supervision.

The Ti-FAD baseline uses the same class split, visual backbone, training schedule, and class-agnostic boundary supervision. It uses action-name text embeddings. The additional Ti-FAD+TS control in Table 7 also receives our taxonomy loss, including its learned level weights and aggregation over seen and unseen member classes. This comparison measures the remaining gain when coarse classification supervision is held constant. THUMOS14 and ActivityNet use the strict protocol, in which unseen-class annotations and unseen-class columns in taxonomy aggregation are excluded from training.

Models and training. Both systems use a CLIP ViT-B/32 image encoder trained end-to-end, followed by a convolutional transformer detector. A frozen CLIP text encoder and a learned projection produce action embeddings. Scaled cosine similarity between text embeddings and video features gives the class scores. The Ti-FAD re-implementation uses one action-name prompt per class, cross-modal attention at each feature-pyramid level, and a foreground prediction head. Our model adds HP, ASA, TS, and ASP. The Ti-FAD+TS control adds the taxonomy loss to the action-name baseline. Both systems use 32-frame windows with a stride of 16 frames.

Composition and supervision. For action detection ($h_d = 3$, $D = 1$), each action is composed from ten phase sub-actions in both branches ($K = 10$ text sentences and video chunks). Taxonomy supervision uses activity and phrase labels, with learned weights for both taxonomy losses and the detection loss. For phrase detection ($h_d = 2$, $D = 2$), the phrase roots the temporal tree. The text branch first combines each generated action sub-node with its phase descriptions and then combines the phrase with these action embeddings. The video branch divides each proposed phrase segment into $K = 3$ equal-length action chunks and each action chunk into $K' = 10$ phase chunks. ASA composes each action chunk with its phase chunks, then composes the location feature with the three refined action chunks. These temporal children come from the generated ordered decomposition, rather than treating all taxonomy members as a temporal sequence. We select K and K' from the dataset statistics: $K = 3$ covers the number of actions in 99% of the phrase instances, and $K' = 10$ matches the ten phase descriptions of each action. Each temporal depth uses separate ASA parameters in each branch. Taxonomy supervision uses only activity labels in this setting. Labels and boundaries of the action and phase sub-nodes are not used as training supervision.

Equal-length chunks as sub-action segments. The video branch divides each proposed segment into equal-length chunks to obtain sub-action features. We assess this partition using FineGym action annotations for analysis, without supplying them as sub-action training supervision: over the 8,463 phrase instances with more than one action, the equal-length chunk assigned to each action overlaps the annotated action with a mean tIoU of 0.59 (median 0.65), and 69% of the instances reach a tIoU of at least 0.5. Unannotated time between consecutive actions accounts for 22–45% of an instance. Equal-length chunks provide temporal regions for feature composition, and neighbor attention lets each region draw on its adjacent regions. Table 4 evaluates the representation built from these regions. Additional taxonomy parent levels contribute description information and weighted losses through Eq. 8, separately from this temporal partition.

A.7 FineDiving Hierarchy and Additional Ablations

Hierarchy and class split. We evaluate action detection on 44 FineDiving classes, with 31 seen and 13 unseen classes per split. We average results over three class splits and over tIoU thresholds 0.3–0.7 in steps of 0.1, as on FineGym. The dataset provides a two-level action–sub-action structure (Xu et al., 2022). Each detected dive is the root of a temporal tree, with three ordered sub-actions at the next level. We generate five phase descriptions for each sub-action, giving a second sub-action level. The temporal tree is dive $\rightarrow$ sub-actions $\rightarrow$ phases ($D = 2$). The one-level variant retains dive $\rightarrow$ sub-actions ($D = 1$), with both Text and Video ASA still active. A separate taxonomy places the detected action classes at $h_d = 2$, with one coarser category level for taxonomy supervision. The video branch divides each action interval into three parts and each part into five finer parts, matching the text tree's granularity. Training uses the coarse-supervision protocol described in Appendix A.6.

Temporal-tree depth. Table 6 compares one and two sub-action levels with both Text and Video ASA active. Adding the phase level improves unseen- and seen-action localization while keeping the detection classes and modalities unchanged. This isolates temporal depth, while Table 5 compares the modalities at $D = 2$.

Table 6: Effect of temporal-tree depth on FineDiving under coarse supervision. Both variants use Text and Video ASA and detect the same action classes at $h_d = 2$. The two-level variant adds five phase children beneath each of the three sub-actions.

Sub-action levels	Unseen mAP	Seen mAP
One ($D = 1$)	48.3	69.1
Two ($D = 2$)	**54.7**	**74.9**
Relative gain	+13.25%	+8.39%

Controlling taxonomy supervision. Table 4 compares the full method with Ti-FAD without taxonomy supervision. Table 7 gives Ti-FAD the same taxonomy loss as our model, including the learned level weights and aggregation over seen and unseen classes. Our model improves both unseen- and seen-class localization on FineGym phrases and FineDiving actions under this matched supervision. This comparison measures the additional benefit of the full hierarchical method over action-name embeddings when both models learn from the same coarse labels. The modality ablation varies the use of Video ASA at the same depth (Table 5), while the depth ablation keeps both modalities and varies the number of sub-action levels (Table 6).

Table 7: Matched-supervision comparison with Ti-FAD given our taxonomy loss (TS). Both models use the same coarse-supervision protocol. FineGym evaluates phrases at $h_d = 2$, and FineDiving evaluates dives at $h_d = 2$. Entries are mAP (%), and gains are relative to Ti-FAD + TS.

	FineGym phrases		FineDiving dives	
Method	Unseen	Seen	Unseen	Seen
Ti-FAD + TS	22.5	58.8	53.7	62.1
Ours	**28.7**	**61.7**	**54.7**	**74.9**
Relative gain	+27.56%	+4.93%	+1.86%	+20.61%

A.8 Class-Name Similarity Analysis

For the running example, CLIP ViT-B/16 embeddings of "a video of action long jump" and "a video of action triple jump" have cosine similarity 0.964, illustrating the ambiguity of action names without descriptions of their temporal composition.

For each class, we measure the cosine similarity between its pretrained CLIP ViT-B/16 embedding and its nearest other class embedding, using the prompt "a video of action ..." with the class labels from the dataset configurations. The mean similarity is 0.869 on ActivityNet, 0.889 on THUMOS14, and 0.957 on FineGym. The proportions above 0.9 are 27%, 35%, and 88.9% (88 of 99 FineGym classes), respectively. These similarities motivate describing actions through their temporal parts and taxonomy categories. They measure ambiguity in the class-name embeddings, rather than the effect of any individual component on localization.

A.9 Comparison Settings

Controlled comparison. We adapt DFAlign and PDA to the same Ti-FAD base as our method, using the same two-stream I3D features, frozen CLIP text encoder, external-classifier fusion, ten random class splits, and language-model generator. Each method retains its own description format. For PDA, Qwen2.5-VL-7B-Instruct generates phase descriptions in place of the original GPT-4o generator. We evaluate seven configurations of filtering, alignment, loss, and training schedule and report the best result. This comparison measures performance under the common implementation and generation setup. Our method improves over Ti-FAD and both adaptations in all four settings.

For reference, the original DFAlign paper reports 54.9/51.8 on THUMOS14 and 37.1/35.7 on ActivityNet, and PDA reports 52.1/46.9 and 37.4/34.6, respectively (75/25 and 50/50). These are the authors' published results under their implementations and description-generation settings. Table 2 labels our adapted results explicitly, and our comparative claim is restricted to the controlled setting.

Other settings. In Table 2, OVFormer (†) uses DINOv2 frame features (Oquab et al., 2023) and GPT-3.5 action descriptions (Gupta et al., 2024). STOV-TAL (‡) uses RGB input, and we report its strongest variant without additional training data. The final block lists methods evaluated under different settings. TF-CADE excludes the external classifier. Bi-TSA evaluates seen and unseen classes jointly. T3AL, LLT, and FreeZAD use pretrained models without training on seen-action annotations.

A.10 Text-Tree Inputs and Examples

Input format. The text branch receives a root description $\tau_0(c)$ and an ordered list of child descriptions $\tau_1(c), \ldots, \tau_K(c)$ (Section 3.2.2). The root describes the action class, its taxonomy category, and available properties. For THUMOS14, the root-description format is:

```
a video of a sports action:  <action name>, a kind of
<taxonomy category>, <properties>
```

Each subsequent sentence describes a temporal part and forms one child of the root. The child list is supplied in its stored order. Tables 8 and 9 reproduce complete model inputs from the THUMOS14 and FineGym description files, preserving their wording. All three examples have one sub-action level ($D = 1$) and ten children ($K = 10$). The FineGym example is an action-level tree ($h_d = 3$).

Reading the examples. Each numbered sentence is one child, so a movement such as the long-jump run-up can occupy several adjacent descriptions. The finer input granularity explains why the ten-child tree differs from the simplified run-up, leap, and landing illustration in Figure 1. The FineGym example describes approach, contact with the vaulting table, flight, and landing within a single gymnastics element. These are class-level text inputs, while the video branch obtains temporal regions from the estimated action interval.

Table 8: Stored THUMOS14 text trees for long jump and billiards. Each column contains one root description and its ten ordered children, with input wording preserved.

Long jump

Root a video of a sports action: long jump, a kind of jump field, with no equipment and with a horizontal motion

1 Athlete stands at the start line, feet shoulder-width apart, arms slightly bent, facing forward.

2 Athlete begins to run, arms pumping, legs driving forward with increasing speed.

3 Athlete accelerates down the runway, body leaning forward, knees bent, arms swinging rhythmically.

4 Athlete plants foot on the takeoff board, legs pushing off with force, body rising into the air.

5 Athlete's body is airborne, legs extended forward, arms reaching back for balance, head up.

6 Athlete's legs bend at the knee mid-air, body tucks slightly, arms adjust for optimal position.

7 Athlete's body reaches peak height, arms extended overhead, legs straight, feet slightly off the ground.

8 Athlete's legs begin to bend again, body rotates slightly, preparing for landing.

9 Athlete's body lowers, knees bending, arms swinging forward, landing feet first on the sandpit.

10 Athlete's legs absorb impact, knees bending, arms flaring out for balance, body settling into the sand.

Billiards

Root a video of a sports action: billiards, a kind of implement strike, in a bent posture

1 A player in a bent posture leans forward, cue stick held steady in right hand, aiming at a white ball on a green felt table.

2 The player's left hand grips the cue stick near the butt, fingers curled for stability, eyes focused on the target ball.

3 The cue stick is slightly angled, tip positioned just above the white ball, ready to strike.

4 The player's body shifts subtly, shoulders aligned with the cue, as the cue tip makes contact with the ball.

5 The cue stick moves forward in a smooth arc, propelling the white ball across the table surface.

6 The white ball rolls straight, leaving a faint trail of motion blur as it approaches the next target.

7 The player's bent posture remains unchanged, eyes tracking the ball's path as it nears the pocket.

8 The ball enters the pocket with a slight bounce, the cue stick still held in the player's hands.

9 The player's body straightens slightly, cue stick lowered, as the ball's trajectory is confirmed.

10 The player's hands relax, cue stick resting on the table, body upright, gaze fixed on the next ball.

Table 9: Stored FineGym action-level text tree: one vault description and ten ordered children, with input wording preserved.

FineGym vault

Root a video of a gymnastics element: round-off, flic-flac on, stretched salto backward with 2 turn off, a kind of round-off entry vault, on the vaulting table

1 The gymnast sprints down the runway toward the vaulting table, arms driving quickly as she builds maximum speed.

2 She hurdles into a round-off, hands striking the runway in turn as her legs snap together onto the springboard.

3 Facing away from the table, she rebounds off the springboard into a flic-flac, arms reaching back overhead for the table.

4 Her hands contact the vaulting table as her body passes through a brief handstand, legs whipping over in an arch.

5 She blocks hard through her shoulders, repulsing off the table and rising into post-flight with her body fully stretched.

6 She sets a stretched salto backward off the table, then immediately wraps her arms across her chest to begin twisting.

7 She completes two full twists around her long axis while somersaulting backward, body straight, legs together, and toes pointed.

8 Finishing the final twist late in flight, she opens her arms wide to slow the rotation and spots the mat.

9 Her stretched body descends toward the landing mat, chest lifting as her feet reach down for the surface.

10 She lands on both feet facing the vaulting table, knees flexing deeply to absorb the landing, arms extending forward.