AXG-Reasoner: Error Detection and Explanation in Long Task Videos with Vision–Language Models

Shih-Po Lee
Northeastern University
lee.shih@northeastern.edu

Ehsan Elhamifar
Northeastern University
e.elhamifar@northeastern.edu

Abstract

Virtual task assistants must recognize and explain users' mistakes to provide effective and corrective guidance. In this paper, we address the problem of error reasoning in long task videos, which is to detect and explain errors. Although recent Vision–Language Models (VLMs) demonstrate strong capabilities in visual question answering, they struggle to attend to the sparse spatiotemporal cues associated with errors in long task videos. We introduce an error reasoning framework, AXG-Reasoner, that leverages a frozen VLM in conjunction with a proposed Action eXecution Graph (AXG) and a temporal action segmentation (TAS) model, obtained and learned from normal (error-free) videos. To enable VLMs to attend to the sparse spatiotemporal cues associated with errors, we decompose each action segment of the video, obtained by TAS, into a sequence of fine-grained subactions by aligning it with the AXG. For each subaction segment, we query the VLM using a small number of keyframes and enhanced prompts to detect and explain errors, enabling data efficiency. To avoid costly manual subaction annotations, we develop a method to automatically construct AXG from training videos using foundation models. Extensive experiments on EgoPER and CaptainCook4D show that our method consistently improves over VLM baselines in error explanation by effectively identifying spatiotemporal cues and achieves state-of-the-art performance in error detection. [1]

1. Introduction

People continually acquire new skills to enrich their daily lives and enhance their professional activities. To support this learning process, AI task assistants have emerged to analyze users' behaviors over time and provide adaptive, context-aware guidance. This has motivated research on action recognition [3, 4, 15, 16, 28, 30, 38, 56], temporal action segmentation [14, 19, 23, 26, 27, 32, 34, 35, 47, 49, 50, 52, 54, 61, 62, 69], video grounding [2, 9, 11, 29, 36, 40, 55, 66, 70], progress prediction [8, 63], and learning from instructional videos [5, 12, 13, 22, 39, 41, 42, 48, 51, 53, 65]. A key required capability of AI task assistants is the ability to detect and explain user errors, enabling users to understand and correct their mistakes effectively. This has motivated recent research on error understanding, which has primarily focused on error detection [7, 17, 18, 21, 25, 37, 43, 58] and error recognition [24]. However, the problem of providing error explainability remains largely unsolved.

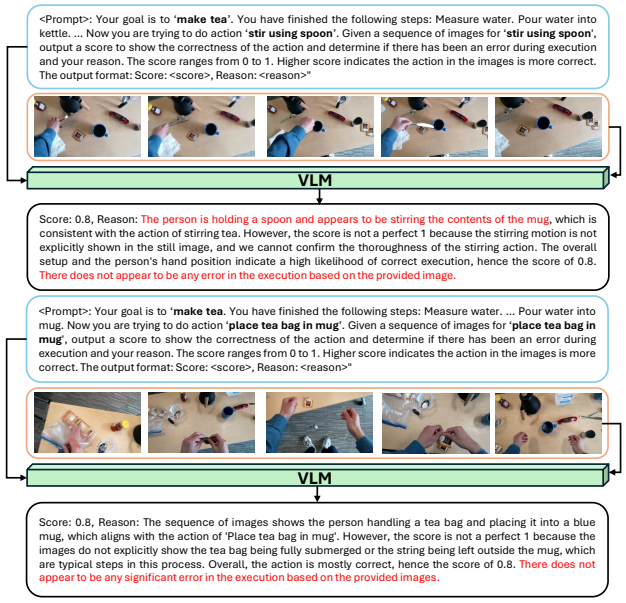

Figure 1. VLMs can fail to recognize and explain errors in long task videos. In the top row, the VLM (Qwen2.5-VL-32B) fails to recognize *spatially* subtle error (using knife instead of spoon). In the bottom row, the VLM fails to recognize a temporally subtle, short error (dropping tea bag).

In this paper, we address the problem of **error reasoning** — the task of **detecting errors** in long task videos and **explaining why** they have been detected as errors. This capability offers two key advantages. First, it facilitates task learning by helping users clearly understand the mistakes they make during the process. Second, it reveals

[1] code: https://github.com/robert80203/AXG-Reasoner

the model's rationale when identifying potential errors, enabling more accurate improvements and promoting interpretable and trustworthy behavior. Despite recent advances in Vision–Language Models (VLMs) for addressing Visual Question Answering (VQA) in videos [1, 6, 33, 45, 57, 68], significant challenges remain, particularly for error reasoning in long task videos.

First, errors in task videos are often **spatially and temporally subtle** compared to their corresponding correct actions, making them challenging for VLMs to detect and interpret. For example, using an incorrect tool or object in part of a long video may involve only a *minor spatial difference* between two small objects (e.g., spoon versus knife), posing a challenge for error detection and explanation via VLMs (the top row in Fig. 1). Similarly, when most of an action execution is correct but *a short temporal segment* contains an error (e.g., picking up a tea bag but dropping it in the middle of the clip, picking up another one, and transferring it to the mug), understanding the error becomes difficult for VLMs (the bottom row in Fig. 1). Consequently, VLMs exhibit performance degradation in error detection and explanation for long videos, as they fail to focus on the *sparse spatial and temporal cues associated with errors*, in contrast to the abundant cues available for correct actions. Therefore, our first objective is to **enhance the ability of pretrained VLMs to focus on sparse spatiotemporal cues indicative of errors**.

Second, due to limited computational resources and the need to maintain efficiency, VLMs can process only a **small subset of frames at inference time** from the large number of frames present in long task videos. Selecting *keyframes* that effectively capture the essential content of each action can therefore help alleviate computational bottlenecks and mitigate the aforementioned loss of focus on temporal cues. Prior work on long-video reasoning has explored various keyframe selection strategies [31, 59], such as hierarchical tree structures for identifying keyframes corresponding to both coarse- and fine-grained activities [59]. However, these approaches fail to specify the precise actions that must be correctly performed in the selected keyframes, which is crucial for guiding VLMs in detecting and explaining errors. Thus, our second objective is to **improve data efficiency by selecting most informative keyframes and invoking VLMs only on them**.

Third, general action descriptions in VLM prompts **lack the precise contextual details necessary for error reasoning**. For example, the general action "place tea bag in mug" may involve several subactions, such as "grab a tea bag," "open the package," "remove the tea bag from the package," and "place the tea bag in the mug," each of which can lead to different types of errors. Prompting VLMs with high-level actions often yields unpredictable and imprecise outputs, as the models must first decompose the action into fine-grained subactions and then infer possible errors based on their prior knowledge. Therefore, our third objective is to **design precise prompts based on fine-grained action decomposition** to more effectively leverage VLMs' reasoning capabilities for detecting and explaining errors.

Paper Contributions. We propose a framework to address the aforementioned challenges of *error reasoning* in long task videos using frozen VLMs. We first segment long task videos into action segments using a TAS model learned from only normal (error-free) training videos. To mitigate VLMs' limitations in attending to subtle spatial and temporal cues associated with errors in long video clips, we automatically decompose each action segment into a sequence of subactions and leverage the strengths of VLMs in understanding short clips to reason about subaction correctness. We make the inference data-efficient by calling VLMs to reason about keyframes of the small number of subactions. We incorporate the subactions themselves into VLM prompts to enhance error reasoning.

To avoid costly manual subaction annotations, we introduce an **Action eXecution Graph (AXG)** built automatically from training videos using foundation models. The AXG captures subactions and encodes feasible execution paths. It is also **training-free** (requiring no parameter optimization) and can be seamlessly integrated with any off-the-shelf TAS model and VLM.

During inference, we perform graph-to-video alignment to segment each action clip produced by TAS into subactions, apply VLM-based reasoning on their keyframes, and aggregate the results for error detection and explanation. Extensive experiments on the EgoPER and CaptainCook4D datasets show that our method surpasses state-of-the-art approaches in error detection and explanation, while maintaining high data efficiency.

2. Related Work

Error Understanding for Procedural Tasks. Recent studies have explored various directions in understanding errors within procedural tasks, including online/offline error detection [17, 21, 25, 37], error recognition [24], and error explanation [43]. Specifically, EgoPED [25] and AMNAR [21] focus on offline error detection, primarily targeting execution errors by utilizing action feature prototypes. In contrast, PREGO [17] and DTGL [37] perform online detection for procedural errors by leveraging action recognition models with LLMs and the pre-conditions based on the learned task graph, respectively. Beyond detection, GTG2Vid [24] jointly detects errors and classifies their types to provide precise feedback to users. Moreover, MistSense [43] trains a framework that integrates an error detection model with a large language model (LLM) to both identify errors and explain their causes, enhancing user understanding and experience. In contrast, we address error reasoning in long task

videos by leveraging VLMs with our training-free AXG, enabling generalizable and interpretable reasoning.

Long Video Understanding via VLMs/LLMs. Recent developments of large language models (LLMs) and vision-language models (VLMs) has advanced research in long video understanding [31, 33, 59, 60, 64]. Due to limited computational resources and the need for flexibility, several recent works have explored training-free frameworks that leverage VLMs for different vision tasks over long videos. For instance, LAVAD [64] tackles video anomaly detection by employing VLMs for frame-level captioning and LLMs for temporal aggregation and anomaly score estimation. Other training-free approaches focus on identifying frames that provide precise visual cues for effective video understanding. For example, VIDEOTREE [59] constructs a hierarchical tree to model video activities at multiple granularities, using Adaptive Breadth Expansion to extract key information and Relevance-guided Depth Expansion to capture finer visual details. BOLT [31] investigates different frame selection strategies and demonstrates that inverse transform sampling improves performance by increasing the sampling probability of frames with higher relevance to the given query. Our method jointly considers keyframe selection and prompt generation to effectively address error reasoning in long task videos.

3. Proposed Method

3.1. Problem Setting and Overview

Problem Setting. Assume we have a video that consists of multiple actions from a given task, and that some of its frames may contain user errors during task execution. Let $\mathbf{X} = \big((\mathbf{I}_1, \mathbf{x}_1), (\mathbf{I}_2, \mathbf{x}_2), \ldots, (\mathbf{I}_T, \mathbf{x}_T)\big)$ denote the sequence of video frames and their corresponding pre-extracted features. We represent the segmentation of the video into distinct actions as $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$, where N is the number of action segments and each segment $\mathbf{v}_n = (a_n, t_n^s, t_n^e)$ consists of the action class a_n and its start and end frame indices t_n^s and t_n^e, respectively. Here, $a_n \in \{0, \ldots, A\}$, where A is the number of actions and class 0 indicates background (task-irrelevant actions). For **error reasoning**, our goal is *twofold*: (1) **Error detection**: predict segment-wise errors $\mathbf{Y} = (y_1, y_2, \ldots, y_N)$, where $y_n \in \{0, 1\}$ and $y_n = 1$ indicates the presence of an error in segment n; and (2) **Error explanation**: generate textual justifications $\mathbf{Z} = (z_1, z_2, \ldots, z_N)$, where each z_n is a free-form description explaining why (or why not) the corresponding segment is erroneous.

For training, we assume we have only normal (error-free) videos of the task with ground-truth framewise action labels. As discussed in the introduction, each coarse-level action can be decomposed into fine-grained subactions, e.g., "put tea bag in mug" may consist of "grab a tea bag," "open the package," "remove the tea bag from the package," and "place the tea bag in the mug". We do not assume training videos come with ground-truth subactions, which is very costly to gather.

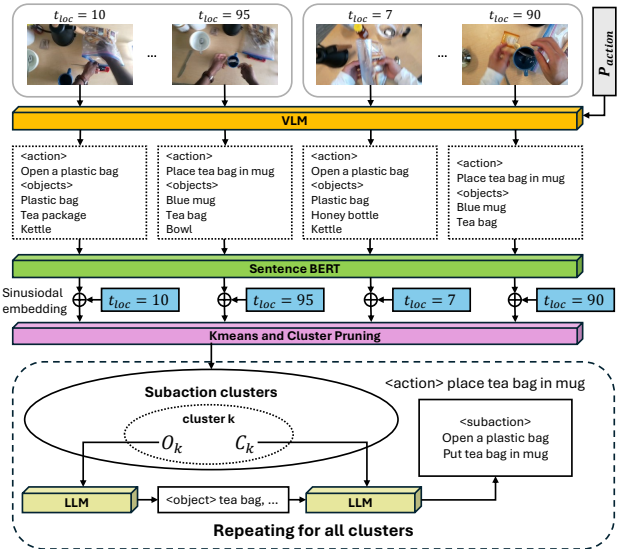

Figure 2. Our subaction generation pipeline consists of 1) action and object description generation via a VLM, 2) clustering with temporal-aware text embeddings and cluster pruning, 3) subaction generation via a LLM.

Framework Overview. We first train a temporal action segmentation (TAS) model ϕ on the available training videos using any existing architecture [24, 25, 34]. A key observation is that frames containing errors are typically classified as one of the action classes (rather than background) in the TAS output (see Figure 5). This is because errors are often spatially and temporally subtle relative to their corresponding correct actions. Our goal is to leverage frozen VLMs to efficiently analyze each action segment for error reasoning: (1) detecting if a segment contains an error and (2) generating a corresponding textual explanation.

To help VLMs focus on the sparse spatial and temporal cues associated with errors, while also improving data efficiency by reducing the number of VLM queries, we divide each action segment into a sequence of subaction segments and select keyframes from each subaction as input to the VLM. This enables the VLM to reason over short, informative clips rather than long, redundant action sequences, providing both stronger reasoning and faster inference.

To automatically obtain subaction segments, we construct an **A**ction e**X**ecution **G**raph (**AXG**) from the training videos using foundation models. In brief, this involves captioning the frames of each action using a VLM, performing temporally aware clustering of the resulting text embeddings, and extracting subactions using an LLM. The nodes of the AXG represent subactions, while the edges encode predecessor–successor relationships, defining feasible sub-

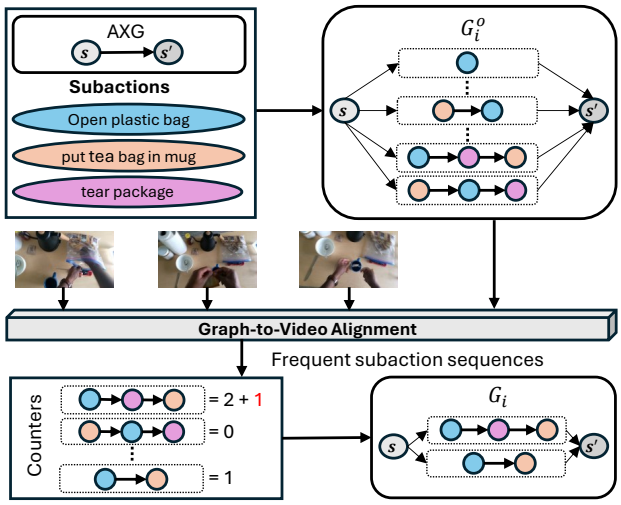

Figure 3. The illustration of our AXG construction process. We constrcut AXG by enumerating all possible subaction sequences and retain the paths that commonly observed in training videos.

Algorithm 1 Inference Procedure for Error Reasoning

Inputs: $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N), \mathbf{v}_n = (a_n, t_n^s, t_n^e)$, $\mathbf{X} = \big((\mathbf{I}_1, \mathbf{x}_1), (\mathbf{I}_2, \mathbf{x}_2), \ldots, (\mathbf{I}_T, \mathbf{x}_T)\big)$, and $G_1, \ldots, G_A$
Outputs: $\mathbf{Z} = (z_1, z_2, \ldots, z_N)$, $\mathbf{Y} = (y_1, y_2, \ldots, y_N)$
for $n = 1$ to N **do**
  Segments $\leftarrow$ G2V$(G_{a_n}, (\mathbf{x}_{t_n^s}, \ldots, \mathbf{x}_{t_n^e}))$ ▷ TSS
  for Segment in Segments **do**
    Frames $\leftarrow$ Sampling_Frames(segment)
    if Assigned segment **then**
      score, reason $\leftarrow \mathcal{F}$(Frames, P^{c})
    else
      score $\leftarrow 0$
      reason $\leftarrow \mathcal{F}$(Frames, P^{r})
    end if
    $y_n \leftarrow 1$, if score $\leqslant 0.5$
    $z_n \leftarrow z_n \oplus$ reason
  end for
end for
return $\mathbf{Y}, \mathbf{Z}$

action sequences. By aligning this graph with a given action clip (graph-to-video alignment), we segment the clip into subactions. Finally, we enrich VLM prompts with the identified subactions to improve the quality of error reasoning.

3.2. Learning Subactions from Training Videos

Our goal is to automatically extract the subactions involved in performing each action. To do so, we take *ground-truth segments of an action* across training videos, sample one every β frame and use a **VLM** with the prompt P_{action} (e.g., "describe the action the person is doing and the name for every object the person is interacting with") to produce **action descriptions and object names** for those. We use Sentence BERT [46] to embed descriptions of actions and objects for the selected frames, see Figure 2.

We apply Kmeans **clustering on temporally-aware textual embeddings** to produce K temporally contiguous and coherent subaction clusters. More specifically, we embed the time-stamp $t_{loc} = \text{round}(\frac{t-t^s}{t^e-t^s} \times 100)$ of each text embedding using a sinusoidal embedding and combine it with textual embedding by summation, where t, t^s, t^e denote the timestamps of the frame associate to the text, and the start and end of the associated action segment, respectively. This allows distinguishing similar actions that have different context (e.g., holding a kettle before moving it close to mug vs holding it after pouring water from it to the mug), thereby facilitating more accurate subaction localization across varying temporal contexts. Some subaction clusters, however, may be *invalid* (e.g., irrelevant to the action) due to noisy or inconsistent action descriptions generated by VLMs. A key characteristic of such clusters is having a small size. Therefore, to obtain **valid clusters**, we discard clusters whose size is less than T_k/K, where T_k denotes the total number of action descriptions in cluster k.[2]

Next, for objects in each valid cluster (associated with a subaction), we use an LLM to obtain the **most common objects** in that subaction (we use the prompt "output at most five object names that commonly occur in O_k", where O_k consists of all object names identified by VLM in cluster k). We then use these common objects to better **summarize the action descriptions** in the same cluster using an LLM. More specifically, we use an LLM with the prompt "output an action that most commonly occurs in the descriptions: C_k by focusing on the object names in O_k", where C_k denotes all VLM-based action descriptions in cluster k. Repeating this process across all valid clusters and removing repetitions of the same subaction summary, we produce final textual list of subactions and their features by performing average over pre-extracted features corresponding to the frames in each cluster associated with an action.

3.3. Action eXecution Graph (AXG)

We build the AXG of each action, which encodes **feasible sequences of subactions** to perform it, see Figure 3. It is a directed acyclic graph $G = (U, E)$, where each node $u \in U$ denotes a subaction in an action and each edge $(u_j, u_i) \in E$ indicates that the subaction u_i can start only after subaction u_j is done. In the AXG, each path from the source node s to the sink node s' represents a valid subaction sequence to execute the action. To construct the AXG for action i, denoted by G_i, we first construct an **initial graph** G_i^o by using the set of subactions obtained from our clustering approach (discussed in the previous subsection) and *enumerating all possible subaction sequences as paths* in this graph. We

[2] In our experiments, this choice worked well. Smaller thresholds increased the number of both good and noisy (bad) clusters.

Method	VLM	EgoPER					
		Quesadilla	Oatmeal	Pinwheel	Coffee	Tea	All
Naive	Qwen2.5-VL	4.0	4.0	4.0	0.0	6.0	3.6
VTREE [59]	Qwen2.5-VL	5.0	6.0	4.0	1.0	9.0	5.0
AXG	Qwen2.5-VL	**22.0**	**17.0**	**18.0**	**17.0**	**13.0**	**17.4**
Naive	InternVL	22.0	19.0	18.0	**22.0**	18.0	19.8
AXG	InternVL	**31.0**	**22.0**	**27.0**	20.0	**23.0**	**24.6**
		Cook4D					
		Hot Chocolate	Sandwich	Burritos	Ramen	Raita	All
Naive	Qwen2.5-VL	8.0	4.0	4.0	2.0	3.0	4.0
VTREE [59]	Qwen2.5-VL	3.0	3.0	6.0	2.0	1.0	3.0
AXG	Qwen2.5-VL	**18.0**	**21.0**	**20.0**	**18.0**	**19.0**	**19.2**
Naive	InternVL	19.0	16.0	**24.0**	**19.0**	12.0	18.0
AXG	InternVL	**23.0**	**21.0**	19.0	18.0	**25.0**	**21.2**

Table 1. Error explanation performance (%) on GT action segments.

then obtain the AXG G_i using graph-to-video alignment by pruning G_i^o via retaining the subaction sequences that are commonly observed in the training videos of action i.

More specifically, we perform **graph-to-video alignment (G2V)** to associate frames within each ground-truth segment of action i in the training videos with their corresponding subactions in G_i^o. In practice, we use [10] as the G2V method, which allows dropping frames when no subactions are matched. G2V also provides the optimal path that best aligns with the sequence of frames. We record the occurrence count of each subaction sequence, and construct G_i by retaining only sequences whose occurrence counts are greater than or equal to $\left\lfloor \frac{N_i}{M_i} \right\rfloor$, where N_i denotes the total number of training action i segments and M_i is the number of paths of G_i^o that occur at least once in the training videos. This ensures that only *consistently observed subaction sequences* are preserved, yielding a robust and representative AXG structure that captures the common procedural flow while discarding outlier or noisy variations.

An alternative yet naive approach to build AXG is to aggregate subaction sequences from training videos by assigning subactions to frames according to their corresponding subaction clusters. However, the obtained subaction segmentations often contain noise, as they may include repeated or interleaved subactions that deviate from the ordering typically observed in the action.

3.4. Error Reasoning Inference

At test time, we *leverage VLMs with our actionwise-AXGs (we have one AXG per action) to perform error detection and explanation* in long task videos. Inference consists of two stages: (1) Temporal Subaction Segmentation (TSS) and (2) Reasoning with VLMs, see Algorithm 1.

Temporal Subaction Segmentation (TSS). We use a temporal action segmentation model ϕ to segment the test video into actions and obtain action segments $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$, with $\mathbf{v}_n = (a_n, t_n^s, t_n^e)$. For a non-background action segment $\mathbf{v}_n$ with action $a_n = i$, we perform TSS using graph-to-video (G2V) alignment [10] using the AXG G_i. We ignore background segments, since they are task-irrelevant actions by definition and most error frames are classified as action segments (see the analysis in the experimental section). We perform TSS for all action segments and obtain their corresponding subaction segmentations. In the output of G2V, each subaction segment is either *assigned* to a subaction or *dropped* (not matched with any subaction, hence considered as error).

Method	TAS	VLM	EgoPER			Cook4D		
			N.	E.	F1@.5	N.	E.	F1@.5
EgoPED [25]	ACTF	N/A	24.0	21.1	22.5	13.4	7.5	10.5
GTG2Vid [24]	GTG2Vid	N/A	42.8	23.2	33.0	18.0	14.3	16.2
Naive	ACTF	Qwen2.5-VL	10.6	24.0	17.3	**20.4**	21.0	20.7
AXG	ACTF	Qwen2.5-VL	**27.4**	**24.4**	**25.9**	18.1	**28.0**	**21.1**
Naive	GTG2Vid	Qwen2.5-VL	30.6	10.2	20.4	13.3	17.4	15.3
AXG	GTG2Vid	Qwen2.5-VL	**36.4**	**30.4**	**33.4**	**19.0**	**38.2**	**28.5**
Naive	ACTF	InternVL3.5	**31.0**	10.2	20.6	18.0	15.7	16.9
AXG	ACTF	InternVL3.5	30.5	**20.7**	**25.6**	**18.7**	**28.8**	**23.8**
Naive	GTG2Vid	InternVL3.5	30.1	9.1	19.6	15.9	21.4	18.7
AXG	GTG2Vid	InternVL3.5	**38.3**	**25.2**	**31.8**	**19.0**	**39.1**	**29.0**

Table 2. Average Error Detection results of different methods over all tasks in each dataset for EgoPER and CaptainCook4D.

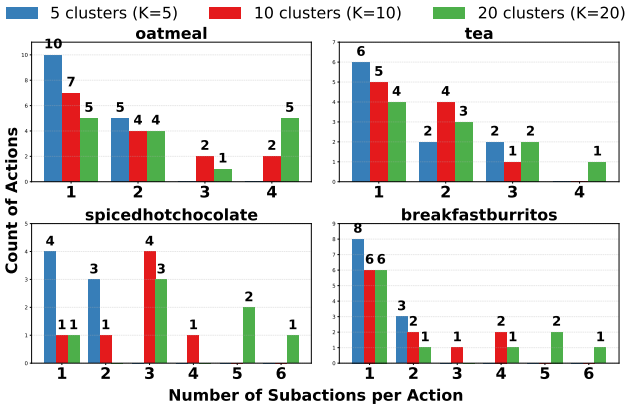

Figure 4. Distribution of the number of subactions per action. Each bar shows how many actions contain a given number of subactions.

Reasoning with VLMs. We use a VLM $\mathcal{F}$ to perform error reasoning on subaction segments. We uniformly sample α frames from each subaction segment as keyframes for $\mathcal{F}$. We design **two types of prompts for $\mathcal{F}$ to handle assigned and dropped segments**, respectively. A segment assigned to a subaction may contain subtle errors related spatial or temporal deviations (e.g., using spoon versus knife) as they are visually similar to the associated subaction. To handle them, we use $\mathcal{F}$ to obtain the correctness score and textual justification. We use the *subaction and action to create an enhanced prompt* P^c: "You are performing subaction

Method	VLM	EgoPER			Cook4D		
		N.	E.	F1	N.	E.	F1
Naive	Qwen2.5-VL	**87.1**	26.3	56.7	**53.0**	29.9	41.5
VTREE	Qwen2.5-VL	85.6	22.2	53.9	52.8	34.7	43.7
AXG	Qwen2.5-VL	80.1	**47.0**	**63.6**	24.7	**65.5**	**45.1**
Naive	InternVL	**88.2**	13.7	50.9	**67.5**	22.2	44.9
AXG	InternVL	79.2	**44.1**	**61.6**	33.8	**58.0**	**45.9**

Table 3. Error detection performance of methods on GT action segments for EgoPER and CaptainCook4D.

Method	VLM	EgoPER	Cook4D
Naive	Qwen2.5-VL	3.4	5.4
VTREE	Qwen2.5-VL	4.8	2.8
AXG	Qwen2.5-VL	**16.8**	**26.6**
Naive	InternVL	18.4	19.4
AXG	InternVL	**25.0**	**21.4**

Table 4. Error explanation (%) for actions with one subaction on GT segments.

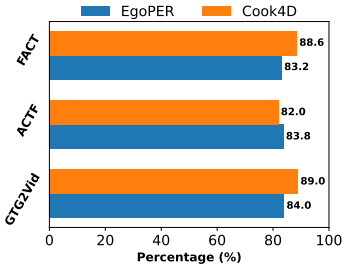

Figure 5. The percentages (%) of error frames classified as actions by different TAS models.

$<$ *subaction* $>$ of action $<$ *action* $>$. Given a sequence of images, output a score that measures the correctness of the subaction being performed and provide your reason, where the score ranges from 0 to 1".

On the other hand, we consider segments dropped by G2V as errors, since they do not correspond to any valid subaction. We use $\mathcal{F}$ to *produce the textual description describing the error* and with the correctness score set to 0. We use the prompt P^{r}: "You are performing action $<$ *action* $>$ and you have made a mistake. Given a sequence of images, describe the mistake being made.". See our supplementary materials for the details of the prompts.

Finally, we **predict action segment v_n as erroneous** if the correctness score of any subaction within v_n is less than or equal to 0.5 and generate the justification z_n by combining all descriptions of subaction segments within v_n.

4. Experiments

4.1. Experimental Setup

Dataset. We evaluate our proposed method on EgoPER [25] and CaptainCook4D [44] for error reasoning, including error detection and explanation. EgoPER consists of 5 procedural tasks with 386 egocentric videos, 5 types of errors, and the corresponding error descriptions, which we use as ground-truth error explanation text. CaptainCook4D contains egocentric procedural tasks videos among 24 tasks with various types of errors and their descriptions as well. For EgoPER, we use all 5 tasks for evaluation with the same training and testing split in [25]. For CaptainCook4D (Cook4D), we follow [24] to select task *Spiced Hot Chocolate* (*Hot Chocolate*), *Microwave Egg Sandwich* (*Sandwich*), *Breakfast Burritos* (*Burritos*), *Ramen*, and *Cucumber Raita* (*Raita*) for evaluation as they have sufficient normal and erroneous videos for both training and evaluation. For each dataset, we report the average score over all its asks, referred to as All.

Evaluation Metrics. For error explanation evaluation, we use an LLM to evaluate free-form descriptions $\mathbf{Z}$ based on ground-truth (GT) action segments. We use the prompt "output a similarity score based on the semantics between GT and predicted descriptions, where the score is from 0 to 1". For error detection evaluation, we follow GTG2Vid [24] to compute the segment-wise F1 score using an overlap threshold γ to determine whether a predicted segment is a correct match. To address data imbalance, we report two F1 scores, N. and E., which treat normal and error segments as the positive class, respectively. Their average is reported as F1@γ. Also, we report F1 score with GT action segments to show the upper-bound performance of error detection.

Baselines. For error explanation, we compare our methods with a keyframe selection method, VIDEOTREE [59] (VTREE), and a naive baseline (Naive) that uniformly sample α frames from action segment as keyframe for $\mathcal{F}$ with prompt "You are performing action $<$ *action* $>$. Given a sequence of images, output a score that measures the correctness of the action being performed and provide your reason, where the score ranges from 0 to 1". We use two VLMs as $\mathcal{F}$ for error reasoning: Qwen2.5-VL-32B-Instruct [45] (Qwen2.5-VL) and InternVL3_5-14B [57] (InternVL3.5). For error detection, we report the performance of Qwen2.5VL, InternVL3.5, and other two non-VLM error detection baselines GTG2Vid [24] and EgoPED [25]. Since GTG2Vid also performs TAS, we use ActionFormer (ACTF) [67] and GTG2Vid [24] as TAS model ϕ.

Implementation Details. We use Timesformer [3] pretrained on Ego4D [20] to extract frame features from videos as in [24]. We set the number of sampling frames α to 8 and 3 for Qwen2.5-VL and InternVL3.5, respectively across all datasets. We set the number of clusters $K = 5$ to obtain a decent performance across different datasets for both error detection and explanation. To avoid noisy reasoning on oversegmented subaction segments, in practice, we ignore the subaction segments whose duration is less than 2 seconds. We set the sampling frame rate $\beta = 6$. We use Qwen2.5VL as the VLM for generating action descriptions and object names from frames. We use Qwen2.5-32B-Instruct (Qwen2.5) as the LLM for object/subaction generation and error explanation evaluation. We train different TAS models on EgoPER and CaptainCook4D, respectively. We do the training and evaluation on a NVIDIA H100 GPU.

K	Oatmeal		Tea		Hot Chocolate		Burritos	
	S	F1	S	F1	S	F1	S	F1
5	20.0	**65.1**	17.0	**77.1**	22.0	39.0	27.0	39.7
10	18.0	63.2	18.0	74.4	19.0	**42.2**	29.0	**41.4**
20	**22.0**	61.3	**19.0**	74.8	**26.0**	40.2	**30.0**	39.9

Table 5. Error detection (F1) and explanation (similarity score, S) performance of AXG across different values of K.

4.2. Experimental Results

Error explanation. Table 1 shows the error explanation performance of different methods on GT action segments for EgoPER and CaptainCook4D. For All in EgoPER, our method outperforms VTREE and Naive (Qwen2.5-VL) by achieving 17.4% and 24.6%, compared to their gains of 5.0% and 3.6%. For All in CaptainCook4D, our method achieves improvements of 19.2% and 21.1%, exceeding the 4.0% and 18.0% obtained by Naive (Qwen2.5-VL and InternVL3.5).

Our experiments yield three main findings. First, AXG consistently improves the error explanation performance of both Qwen2.5-VL and InternVL3.5, demonstrating its effectiveness in keyframe selection and enriched prompt construction. Second, VTREE struggles to identify keyframes in task videos, as it primarily relies on LLM-based clustering without explicitly modeling procedural action structure. Third, InternVL3.5 reliably outperforms Qwen2.5-VL in error explanation across all settings.

Error Detection. Table 2 compares error detection performance across different methods on EgoPER and CaptainCook4D using various TAS models. Overall, AXG consistently outperforms the naïve baseline, improving F1@0.5 by roughly 13% on both datasets when paired with GTG2Vid as TAS model and Qwen2.5-VL. Moreover, AXG achieves state-of-the-art performance, yielding gains of 33.4% and 29.0% with Qwen2.5-VL and InternVL3.5, respectively, higher than the 33.0% and 16.2% reported by GTG2Vid on EgoPER and CaptainCook4D. These results indicate that our method enhances VLMs' ability to detect errors without over-flagging normal segments.

Next, Table 3 reports error detection performance on GT action segments, reflecting the upper-bound capability of the models. Under this setting, AXG achieves 63.6% and 45.1% F1@.5, outperforming VTREE (53.9% and 43.7%) and Naive (56.7% and 41.5%) on EgoPER and CaptainCook4D with Qwen2.5-VL. AXG also improves over InternVL3.5, reaching 61.6% and 45.9%, compared with the naive baseline scores of 50.9% and 44.9%. Although AXG shows a drop in N., indicating that it over-flags normal segments as errors, it achieves superior overall error detection performance compared to the naive baseline and VTREE.

Temporal Action Segmentation on Errors. We investigate the prediction behaviors of different TAS models when they encounter errors mentioned in Section 3.4. Figure 5 presents the percentage of error frames classified as any action class rather than background class by TAS models. Across both datasets, the most error frames are categorized as actions, accounting for over 80% in all cases. This observation indicates that errors predominantly occur within action segments rather than background ones, consistent with the intuition that procedural mistakes are more similar to normal actions, justifying our claim in Section 3.4. Among the models, GTG2Vid and a recent TAS model, FACT [34], achieve slightly higher action-error ratios than EgoPED, implying that their segmentation boundaries align more closely with actual error regions.

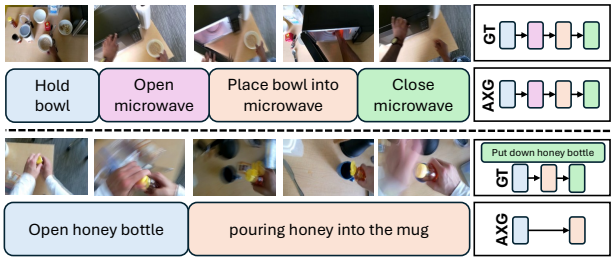

Figure 6. Visualization of AXG (left) and GT subactions (right) for the actions "put bowl in microwave" (top) and "add honey to mug" (bottom).

Detailed Analysis for Subactions in AXG. In this section, we 1) analyze how the number of subactions generated by AXG varies with the number of clusters K and how this relates to error detection and explanation performance and 2) visualize the subactions.

Figure 2 shows the distribution of subaction counts per action under different clustering settings ($K = 5, 10, 20$). Across the four tasks, most actions contain only a single subaction with $k = 5$ or 10. This trend suggests that the majority of actions are either (1) simple, (2) temporally short, or (3) dominated by one long subaction, with only a small portion requiring multiple subactions to capture finer procedural structure. Table 4 reports the error explanation performance specifically for actions whose AXGs contain only one subaction. Specifically, AXG achieves 25.0% and 21.4% higher F1 scores than the naive baseline with InternVL3.5. This demonstrates that performing TSS with a single-subaction AXG can effectively localize a refined subaction segment and erroneous frames within an action, improving error explanation.

As the number of clusters increases, we observe a rise in actions containing more subactions, reflecting the finer procedural granularity captured by larger values of K. Table 5 reports error explanation and detection performance on GT action segments across different cluster sizes for Oatmeal and Tea in EgoPER, and Hot Chocolate and Burritos in CaptainCook4D. The results show that larger K generally improves AXG's error explanation performance, sug-

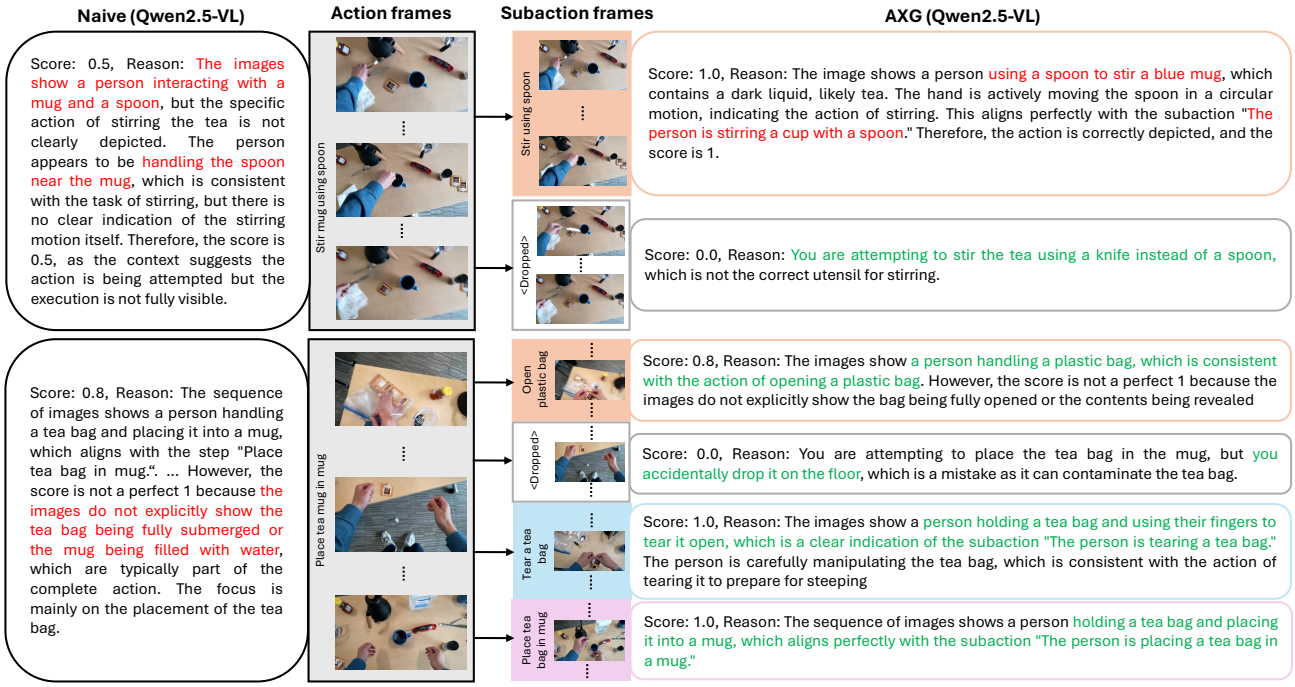

Figure 7. Qualitative results on error reasoning for action "stir mug using spoon" (top) and "place tea bag in mug" (bottom) of task Tea in EgoPER.

gesting that finer subactions offer more temporal cues for identifying errors. However, increasing the number of clusters degrades error detection performance, indicating that excessive granularity introduces noisy subactions that may cause AXG to over-flag normal segments as errors.

Finally, we manually construct the ground-truth (GT) AXGs for two actions in EgoPER and provide qualitative comparisons in Figure 6, which demonstrate that our AXGs are semantically meaningful. However, GT AXGs may have annotator bias, as their structure can vary based on subjective interpretations of subactions. Overall, our AXGs closely align with those constructed by human annotators.

Qualitative Results. Figure 7 (left) shows the explanation generated by Qwen2.5-VL, while the right panel displays the subaction segments and corresponding explanations produced by AXG. The frames in the gray box at the top of the figure illustrate the error "stir mug using knife" for the correct action "stir mug using spoon", demonstrating the minor spatial difference between the two objects. The naive baseline fails to distinguish this subtle difference. Although AXG also struggles to correctly classify the action due to the underlying VLM's limited capability, it successfully drops the erroneous frames (hence we declare the segment as error) and generates an accurate explanation: "You are attempting to stir the tea using a knife instead of a spoon" (shown in green). A key observation is that prompting the model with phrases such as "You have made a mistake" significantly improves the VLM's ability to attend to the correct object and localize the source of the error.

On the other hand, the frames at the bottom of the figure shows the error "drop tea bag on floor" for the correct action "place tea bag in mug", demonstrating the error occurring shortly in the action. Naive baseline recognizes the action as correct because the action is performed eventually but provides bad explanation "the images do not explicitly show the tea bag being fully submerged or the mug being filled with water" (in red). Our method segments the action into three subactions (first, third, and fourth one) with explanations that correctly describe the associated subactions and one dropped segment (second one) with correct explanation "you accidentally drop it on the floor" for describing errors. AXG effectively localizes the short segment containing errors with good explanation.

5. Conclusions

We investigated **error reasoning** in long task videos, by proposing and incorporating into VLMs the Action eXecution Graph (AXG), a training-free method that decomposes each action into subactions and encodes their possible execution sequences. By performing temporal subaction segmentation, AXG identifies keyframes and enriches prompts with subactions to facilitate error reasoning. Experiments on two datasets demonstrate that our framework effectively detects errors when combined with various temporal action segmentation models and consistently enhances error explanation performance across different VLMs.

6. Acknowledgments

This work was funded, in part, by ARPA-H (1AY2AX000062), NSF (IIS-2115110), ONR (N000142512287) and DARPA (HR00112220001). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Government.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022. 2

[2] Kumar Ashutosh, Santhosh Kumar Ramakrishnan, Triantafyllos Afouras, and Kristen Grauman. Video-mined task graphs for keystep recognition in instructional videos. *Neural Information Processing Systems*, 2023. 1

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 1, 6

[4] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[5] Chien-Yi Chang, De-An Huang, Danfei Xu, Ehsan Adeli, Li Fei-Fei, and Juan Carlos Niebles. Procedure planning in instructional videos. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI*, 2020. 1

[6] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 2

[7] Guodong Ding, Fadime Sener, Shugao Ma, and Angela Yao. Every mistake counts in assembly. *arXiv: 2307.16453*, 2023. 1

[8] G. Donahue and E. Elhamifar. Learning to predict activity progress by self-supervised video alignment. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[9] Nikita Dvornik, Isma Hadji, Konstantinos G Derpanis, Animesh Garg, and Allan D Jepson. Drop-dtw: Aligning common signal between sequences while dropping outliers. In *NeurIPS*, 2021. 1

[10] Nikita Dvornik, Isma Hadji, Hai Pham, Dhaivat Bhatt, Brais Martinez, Afsaneh Fazly, and Allan D Jepson. Flow graph to video grounding for weakly-supervised multi-step localization. In *European Conference on Computer Vision*, pages 319–335. Springer, 2022. 5

[11] Nikita Dvornik, Isma Hadji, Ran Zhang, Konstantinos Derpanis, Animesh Garg, Richard Wildes, and Allan Jepson. Stepformer: Self-supervised step discovery and localization in instructional videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2023. 1

[12] E. Elhamifar and D. Huynh. Self-supervised multi-task procedure learning from instructional videos. *European Conference on Computer Vision*, 2020. 1

[13] E. Elhamifar and Z. Naing. Unsupervised procedure learning via joint dynamic summarization. *International Conference on Computer Vision*, 2019. 1

[14] Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3575–3584, 2019. 1

[15] C. Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[16] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 1

[17] Alessandro Flaborea, Guido Melendugno, Leonardo Pliniq, Luca Scofanoq, Edoardo Matteisq, Antonino Furnari, Giovanni Farinella, and Fabio Galasso. Prego: online mistake detection in procedural egocentric videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1, 2

[18] Reza Ghoddoosian, Isht Dwivedi, Nakul Agarwal, and Behzad Dariush. Weakly-supervised action segmentation and unseen error detection in anomalous instructional videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10128–10138, 2023. 1

[19] Uzay Gökay, Federico Spurio, Dominik R. Bach, and Juergen Gall. Skeleton motion words for unsupervised skeleton-based temporal action segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 1

[20] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh K. Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Z. Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Christian Fuegen, Abrham Gebreselasie, Cristina González, James M. Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jáchym Kolár, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran K. Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Yunyi Zhu, Pablo Arbeláez, David J. Crandall, Dima Damen, Giovanni Maria Farinella, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard A. Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4d: Around the world

in 3,000 hours of egocentric video. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18973–18990, 2021. 6

[21] Wei-Jin Huang, Yuan-Ming Li, Zhi-Wei Xia, Yu-Ming Tang, Kun-Yu Lin, Jian-Fang Hu, and Wei-Shi Zheng. Modeling multiple normal action representations for error detection in procedural tasks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2025. 1, 2

[22] Mohaiminul Islam, Tushar Nagarajan, Huiyu Wang, Fu-Jen Chu, Kris Kitani, Gedas Bertasius, and Xitong Yang. Propose, assess, search: Harnessing llms for goal-oriented planning in instructional videos. *European Conference on Computer Vision*, 2024. 1

[23] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks for action segmentation and detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[24] S. Lee and E. Elhamifar. Error recognition in procedural videos using generalized task graph. *International Conference on Computer Vision*, 2025. 1, 2, 3, 5, 6

[25] S. Lee, Z. Lu, Z. Zhang, M. Hoai, and E. Elhamifar. Error detection in egocentric procedural task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1, 2, 3, 5, 6

[26] M. Li, L. Chen, Y. Duarr, Z. Hu, J. Feng, J. Zhou, and J. Lu. Bridge-prompt: Towards ordinal action understanding in instructional videos. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[27] Shi-Jie Li, Yazan AbuFarha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. Ms-tcn++: Multi-stage temporal convolutional network for action segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. 1

[28] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[29] Zeqian Li, Qirui Chen, Tengda Han, Ya Zhan, Yanfeng Wang, and Weidi Xie. Multi-sentence grounding for long-term instructional video. *European Conference on Computer Vision*, 2024. 1

[30] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1

[31] Shuming Liu, Chen Zhao, Tianqi Xu, and Bernard Ghanem. Bolt: Boost large vision-language model without training for long-form video understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 3318–3327, 2025. 2, 3

[32] Yang Liu, Jiayu Huo, Jingjing Peng, Rachel Sparks, Prokar Dasgupta, Alejandro Granados, and Sebastien Ourselin. Skit: a fast key information video transformer for online surgical phase recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21074–21084, 2023. 1

[33] Yujie Lu, Yale Song, William Wang, Lorenzo Torresani, and Tushar Nagarajan. Vited: Video temporal evidence distillation. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 8501–8511, 2025. 2, 3

[34] Z. Lu and E. Elhamifar. Fact: Frame-action cross-attention temporal modeling for efficient action segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1, 3, 7

[35] Z. Lu and E. Elhamifar. Multi-modal few-shot temporal action segmentation. *International Conference on Computer Vision*, 2025. 1

[36] Z. Lu, A. Iftekhar, G. Mittal, T. Meng, X. Wang, C. Zhao, R. Kukkala, E. Elhamifar, and M. Chen. Decafnet: Delegate and conquer for efficient temporal grounding in long videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2025. 1

[37] Antonino Furnari Luigi Seminara, Giovanni Maria Farinella. Differentiable task graph learning: Procedural activity representation and online mistake detection from egocentric videos. *Neural Information Processing Systems*, 2024. 1, 2

[38] K. Mangalam, H. Fan, Y. Li, C. Wu, B. Xiong, C. Feichtenhofer, and J. Malik. Reversible vision transformers. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[39] A. Miech, J-B. Alayrac, L. Smaira, I. Laptev, J. Sivic, and A. Zisserman. End-to-end learning of visual representations from uncurated instructional videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1

[40] Fangzhou Mu, Sicheng Mo, and Yin Li. Snag: Scalable and accurate video grounding. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[41] Tushar Nagarajan and Lorenzo Torresani. Step differences in instructional video. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[42] Kumaranage Ravindu Yasas Nagasinghe, Honglu Zhou, Malitha Gunawardhana, Martin Renqiang Min, Daniel Harari, and Muhammad Haris Khan. Why not use your textbook? knowledge-enhanced procedure planning of instructional videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[43] Constantin Patsch, Yuankai Wu, Marsil Zakour, Driton Salihu, and Eckehard Steinbach. Mistsense: Versatile online detection of procedural and execution mistakes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 1, 2

[44] Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Bhavya Gouripeddi, Jikai Wang, Qifan Zhang, Vasundhara Komaragiri, Eric Ragan, Nicholas Ruozzi, Yu Xiang, and Vibhav Gogate. CaptainCook4D: A Dataset for Understanding Errors in Procedural Activities, 2024. 6

[45] Qwen-Team. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 2, 6

[46] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods*

in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019. 4

[47] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 1

[48] Y. Shen and E. Elhamifar. Semi-weakly-supervised learning of complex actions from instructional task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1

[49] Y. Shen and E. Elhamifar. Progress-aware online action segmentation for egocentric procedural task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[50] Y. Shen and E. Elhamifar. Understanding multi-task activities from single-task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2025. 1

[51] Y. Shen, L. Wang, and E. Elhamifar. Learning to segment actions from visual and language instructions via differentiable weak sequence alignment. *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1

[52] Yaser Souri, Mohsen Fayyaz, Luca Minciullo, Gianpiero Francesca, and Juergen Gall. Fast Weakly Supervised Action Segmentation Using Mutual Consistency. *PAMI*, 2021. 1

[53] Tomavs Souvcek, Dima Damen, Michael Wray, Ivan Laptev, and Josef Sivic. Genhowto: Learning to generate actions and state transformations from instructional videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[54] Yuhao Su and Ehsan Elhamifar. Two-stage active learning for efficient temporal action segmentation. pages 161–183, 2024. 1

[55] Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1

[56] Mengmeng Wang, Jiazheng Xing, and Yong Liu. Actionclip: A new paradigm for video action recognition. *CoRR*, 2021. 1

[57] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025. 2, 6

[58] Xin Wang, Taein Kwon, Mahdi Rad1 Bowen Pan, Ishani Chakraborty, and Sean Andrist. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. *IEEE International Conference on Computer Vision*, 2023. 1

[59] Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. Videotree: Adaptive tree-based video representation for llm reasoning on long videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 3272–3283, 2025. 2, 3, 5, 6

[60] Muchao Ye, Weiyang Liu, and Pan He. Vera: Explainable video anomaly detection via verbalized learning of vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, 2025. 3

[61] S. Yeung, O. Russakovsky, N. Jin, M. Andriluka, G. Mori, and L. Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *International Journal of Computer Vision*, 2018. 1

[62] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. In *The British Machine Vision Conference (BMVC)*, 2021. 1

[63] P. Zameni, Y. Shen, and E. Elhamifar. Moscato: Predicting multiple object state change through actions. *International Conference on Computer Vision*, 2025. 1

[64] Luca Zanella, Willi Menapace, Massimiliano Mancini, Yiming Wang, and Elisa Ricci. Harnessing large language models for training-free video anomaly detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 3

[65] Ali Zare, Yulei Niu, Hammad Ayyubi, and Shih-Fu Chang. Rap: Retrieval-augmented planner for adaptive procedure planning in instructional videos. *European Conference on Computer Vision*, 2024. 1

[66] Runhao Zeng, Jiaqi Mao, Minghao Lai, Minh Hieu Phan, Yanjie Dong, Wei Wang, Qi Chen, and Xiping Hu. Ovg-hq: Online video grounding with hybrid-modal queries. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 1

[67] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. *European Conference on Computer Vision*, 2022. 6

[68] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023. 2

[69] Junbin Zhang, Pei-Hsuan Tsai, and Meng-Hsun Tsai. Semantic2graph: Graph-based multi-modal feature fusion for action segmentation in videos, 2022. 1

[70] D. Zhukov, J. B. Alayrac, R. G. Cinbis, D. Fouhey, I. Laptev, and J. Sivic. Cross-task weakly supervised learning from instructional videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1