This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Modeling Multiple Normal Action Representations for Error Detection in Procedural Tasks

Wei-Jin Huang[1,*] Yuan-Ming Li[1,*] Zhi-Wei Xia[1] Yu-Ming Tang[1] Kun-Yu Lin[1]
Jian-Fang Hu[1] Wei-Shi Zheng[1,2,3,4,†]
[1]School of Computer Science and Engineering, Sun Yat-sen University, China
[2]Peng Cheng Laboratory, China
[3]Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, China;
[4]Guangdong Province Key Laboratory of Information Security Technology, China
{huangwj235, liym266}@mail2.sysu.edu.cn; wszheng@ieee.org

Abstract

Error detection in procedural activities is essential for consistent and correct outcomes in AR-assisted and robotic systems. Existing methods often focus on temporal ordering errors or rely on static prototypes to represent normal actions. However, these approaches typically overlook the common scenario where multiple, distinct actions are valid following a given sequence of executed actions. This leads to two issues: (1) the model cannot effectively detect errors using static prototypes when the inference environment or action execution distribution differs from training; and (2) the model may also use the wrong prototypes to detect errors if the ongoing action label is not the same as the predicted one. To address this problem, we propose an Adaptive Multiple Normal Action Representation (AMNAR) framework. AMNAR predicts all valid next actions and reconstructs their corresponding normal action representations, which are compared against the ongoing action to detect errors. Extensive experiments demonstrate that AMNAR achieves state-of-the-art performance, highlighting the effectiveness of AMNAR and the importance of modeling multiple valid next actions in error detection. The code is available at https://github.com/iSEE-Laboratory/AMNAR.

1. Introduction

Understanding procedural activities is an important field in video action understanding [19–23, 49, 50], as it reflects how an AI model recognizes actions [1, 37, 39, 41, 44, 45], separates steps [2, 6, 15, 17, 29, 43, 46], and plans movements [10, 12, 26, 35], similar to how humans conduct daily tasks (e.g., cooking, assembling toys, using tools, etc.). Since procedural tasks require consistent outcomes without errors, being able to detect natural mistakes is a crucial ability for next-generation AR assisted and robotic systems [5, 11, 36, 38, 47, 48].

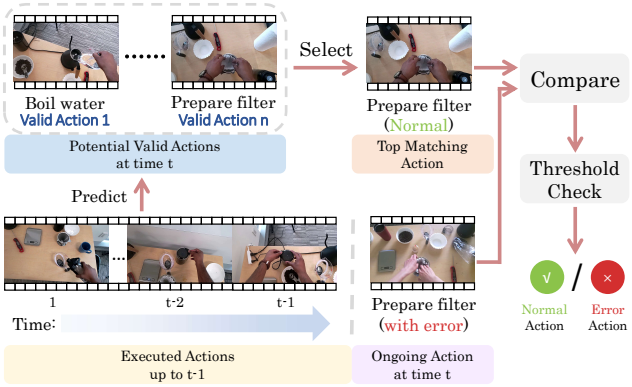

Figure 1. **Illustration of error detection using multiple valid next actions at time t.** After "Grinding Coffee Bean" at time $t-1$, valid next actions include "Boil Water" and "Prepare Filter." The best matching action is selected and compared with the ongoing action. If their distance exceeds the threshold, the action is marked as an error; otherwise, it is marked as normal.

Achieving reliable error detection requires a foundational understanding of what a normal action should be, allowing comparison to the ongoing action. So, for error detection using AI, a key challenge is constructing a "normal action representation" of the current action. Approximating a "normal action representation," some current methods [4, 7, 33] model transition relationships from correctly executed action sequences, thereby predicting the correct current action label based on prior actions. However, while these approaches can detect label-level errors by comparing predicted and actual action labels, they often fail to capture the full representation of a normal action, leading them to

*: Equal contributions. †: Corresponding author.

overlook cases where the correct action label is executed but deviates from the expected normal behavior. For instance, pouring water into a filter but spilling some outside shares the same label as the correct action, yet introduces an error (spillage) that current methods often miss.

Recently, a contrastive learning method[16] tackles this limitation by leveraging prototype-based representations. This method learns a series of prototypes during training to represent the normal execution of each action class. During inference, it detects errors by comparing the current action's representation to the closest prototype in the same class. While effective in some scenarios, these prototypes are static after the training stage and cannot adaptively change with action executions, which struggles to detect errors effectively when the action distribution varies from that of the training samples (e.g., diverse execution styles, tools with varied appearances, or distinct environments).

To detect errors with varying action distributions, we argue that an effective error detection model should be capable of dynamically generating a normal action representation for the current action, conditioned on previously executed actions. Such adaptability allows the model to account for variations in action execution that static prototypes cannot handle. From this perspective, a straightforward implementation might attempt to predict a normal action representation of the current action from the past action sequence. However, this naive approach introduces a new challenge: after a sequence of executed actions, multiple valid next actions may logically follow, depending on user preferences or contextual factors, rather than adhering to a strictly defined sequence. For instance, in the process of making coffee, after grinding the coffee beans, the next step might involve boiling water, preparing the filter, or selecting a cup—each of which is a valid action depending on the task context, as illustrated in Fig. 1. This range of valid next actions following a given sequence of actions makes it challenging for a single normal action representation to encompass all possible correct action representations.

To overcome these challenges, we propose a novel Adaptive Multiple Normal Action Representation (AMNAR) framework for error detection that predicts and models multiple valid next actions, dynamically creating multiple adaptable normal action representations that accurately capture the diversity of procedural task execution. Specifically, our approach first employs an action segmentation model to provide an initial executed action sequence, which serves as input for the Potential Action Prediction Block (PAPB). Subsequently, PAPB predicts all valid actions based on executed actions leveraging the task graph and dynamic programming. Next, our Representations Reconstruction Block (RRB) reconstructs multiple normal action representations for each valid action. As shown in Fig. 2, these modules enable AMNAR to represent all valid actions adaptively in error detection. Finally, the Representation Matching Block (RMB) selects the most likely normal action representation of current execution step and assesses its conformity with the ongoing action. By comparing this conformity to a predefined threshold, we determine whether the current action has an error.

We evaluate our error detection framework on three datasets: EgoPER [16], HoloAssist [38], and CaptainCook4D [28]. Our method improves Error Detection Accuracy (EDA) by **7.4%** and Area Under the Curve (AUC) by **6.5%** on EgoPER, **3.7%** and **1.3%** on HoloAssist, and **2.5%** and **5.3%** on CaptainCook4D, proving its robustness across tasks. Furthermore, ablation studies on our framework validate the contributions of each component, revealing several insights.

The main contributions of our work are as follows:

1. We introduce a novel approach that dynamically generates multiple normal action representations for the current action, conditioned on previously executed actions, addressing the challenge of adaptively representing all normal action representations of the current action.
2. We develop a new framework utilizing task graphs and dynamic programming to predict multiple valid next actions. This enables precise and context-aware error detection by comparing the current action with adaptively generated multiple normal action representations.
3. Comprehensive experiments on the EgoPER, HoloAssist and CaptainCook4D datasets validate the effectiveness of each component in our framework, achieving state-of-the-art performance and demonstrating robustness and flexibility in handling diverse procedural task errors.

2. Related Work

Error Detection [4, 7, 8, 11, 13, 14, 16, 18, 25, 27, 28, 32–34, 38] is a task to detect when errors occur in procedural activities. Starting from EPIC-Tent[34], many researchers have begun to focus on the Error Detection task. Ding et al.[4] attempt to identify both order-related errors in toy assembly and part-relative placement errors by solving the Error Detection problem using the constructed Graphs method. Subsequently, several works [7, 33] focus on detecting order-related errors. Specifically, Flaborea et al.[7] focus on utilizing large language models to identify order-related errors, while Seminara et al.[33] propose using differentiable task graph matrices for the same purpose. Recently, EgoPED [16] expands this field by introducing errors beyond action order (*e.g.*, omission, addition or modification of steps) and proposes a new contrastive step prototype learning framework. Our work shares the same problem formulation with EgoPED [16]. Differently, we address the challenge of multiple valid next actions that can follow any given step in procedural tasks. Our approach adaptively models all normal action representations of these

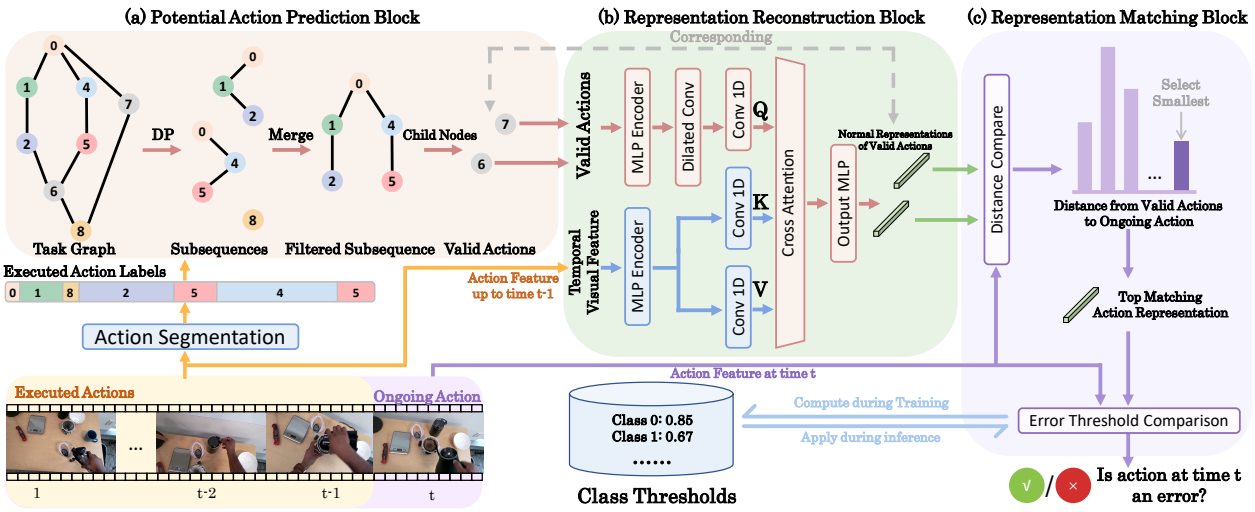

Figure 2. **Overview of the Adaptive Multiple Normal Action Representation (AMNAR) framework.** The process begins with an Action Segmentation module identifying executed actions from video input. **(a) Potential Action Prediction Block** predicts valid next actions using a task graph from executed action labels. **(b) Representation Reconstruction Block** generates normal action representations for these valid actions, leveraging temporal visual features. **(c) Representations Matching Block** compares the ongoing action's feature at time t with the generated representations to detect errors, indicated by a checkmark (✓) for normal actions or a cross (✗) for errors.

valid actions, enabling robust error detection through adaptively generated normal action representations for each possible action.

Video Anomaly Detection (VAD) focuses on spotting unusual events (*e.g.*, accidents or suspicious behaviors) in surveillance footage. VAD identifies deviations from normal activities in videos, which could indicate dangerous or unexpected situations such as falls or unauthorized entries into restricted areas. One of the main branches in this field is the reconstruction-based VAD [9, 24, 30, 31, 42]. In this branch, the models are trained to reconstruct normal frames or accurate sequences, and detect the anomaly by measuring the reconstruction error between the reconstructed and original frames (or sequences). Unlike VAD, which primarily detects deviations based on low-level visual or statistical anomalies, our approach assesses whether ongoing actions align with the predicted normal action representations of all valid next actions. Additionally, while VAD typically reconstructs a single normal scene, our method addresses the challenge of error detection in complex procedural tasks by simultaneously modeling multiple normal action representations for multiple valid next actions.

3. Method

To adaptively reconstruct all normal action representations of valid actions, which are compared with ongoing actual action to detect errors, we propose an Adaptive Multiple Normal Action Representation (AMNAR) framework that explicitly models diverse normal action representations of all valid next actions based on executed action sequence.

We will first introduce problem formulation in Sec. 3.1. After that, we provide an overview of the proposed method AMNAR in Sec. 3.2, and introduce the detailed designs in Secs. 3.3 to 3.7.

3.1. Problem Formulation

Following previous work[16], we train our model with normal videos and their corresponding action labels for each procedural task execution step. Each video, denoted as $V = \{f_i\}_{i=1}^{N}$, is paired with frame-wise action labels $Y = \{y_i\}_{i=1}^{N}$, where N indicates the number of frames, and each y_i maps to one of S predefined action classes or to a background class, expressed as $y_i \in \{1, 2, \ldots, S, S+1\}$. During inference, the objective is to detect error actions, denoted as $E = \{e_j\}_{j=1}^{M}$, where M is the number of errors.

3.2. Method Overview

Given that multiple valid next actions may follow a given executed action sequence, we need to create accurate representations for each possible valid action to detect small deviations in how actions are executed, even if the overall action type is correct. Combining all valid actions with video context, our method reconstructs all normal action representations for each possible current action. The top matching representation to the actual action is then selected and used to detect errors. To do this, we propose an Adaptive Multiple Normal Action Representation (AMNAR) framework, as illustrated in Fig. 2.

AMNAR initially extracts visual features from videos using a visual feature extractor. The features are then fed into an Action Segmentation Model, which outputs labeled action segments with start and end frames. Subsequently, the **Potential Action Prediction Block** predicts all valid next actions based on the task graph and executed actions. The **Representation Reconstruction Block** then generates

representations for these predicted actions. Lastly, the **Representation Matching Block** assesses any deviations between these representations and the ongoing action features to identify possible errors in the ongoing action.

3.3. Action Sequences and Features Execution

We represent procedural task actions using a pre-trained feature extractor (e.g., I3D [1]) to obtain initial visual features, which are processed by the Action Segmentation Model (ASM). The ASM identifies action segments $A = \{a_k\}_{k=1}^{H} = \{(y_k, st_k, ed_k)\}_{k=1}^{H}$, where H is the total number of actions, y_k is the label for the k-th segment, and st_k and ed_k denote the start and end frames. The ASM also generates a refined feature set $F = \{f_i\}_{i=1}^{N}$, with each f_i as a frame-level feature vector.

The executed action sequence up to time t, denoted $s_t = \{y_k\}_{k=1}^{t-1}$, captures prior actions. To account for varying segment lengths, we compute an action feature f_t^{action} by averaging frame-level features within each segment:

$$f_t^{\text{action}} = \text{average}(\{f_i\}_{i=st_t}^{ed_t}), \tag{1}$$

where the average is taken over frames from st_t to ed_t.

3.4. Potential Action Prediction Block

In procedural tasks, multiple valid actions may follow a given execution sequence. To address this, our Potential Action Prediction Block (PAPB) identifies all valid next steps using a predefined task graph G that encodes task-specific action sequences. PAPB maps the current action sequence $s_t = \{y_1, y_2, \ldots, y_{t-1}\}$, where each y_i represents an executed action label, onto G to determine the set of logically valid next actions.

To handle potential inaccuracies of action segmentation label, such as mislabeled or omitted actions, PAPB employs dynamic programming (DP) to compute s_t^*, the filtered subsequence of s_t that aligns with G, as illustrated in part (a) of Fig. 2 and Fig. 3. For example, given an executed sequence $s_t = [0, 1, 8, 2, 5, 4, 5]$. Using DP, PAPB identifies the longest common subsequences (lcs) that form non-branching paths in G. It maintains two arrays: dp$[i]$, tracking the length of the longest non-branching subsequence ending at index i, and subseq$[i]$, storing the corresponding subsequence. Updates occur as follows:

$$\text{dp}[i] = \max(\text{dp}[i], \text{dp}[j] + 1), \tag{2}$$

$$\text{subseq}[i] = \begin{cases} \text{subseq}[j] \cup \{y_i\}, & \text{if dp}[j] + 1 > \text{dp}[i], \\ \text{subseq}[i] \cup (\text{subseq}[j] \cup \{y_i\}), & \text{if dp}[j] + 1 = \text{dp}[i], \end{cases} \tag{3}$$

where a non-branching subsequence is a continuous path in G without splits (e.g., [0, 1, 2] in the task graph).

Next, PAPB merges these subsequences into a unified subgraph, as shown in part (a) of Fig. 2 and Fig. 3. Since node 0 is shared between lcs_1 and lcs_2, they are merged into $s_t^* = [0, 1, 2, 4, 5]$, capturing all relevant executed actions. Finally, PAPB identifies the valid next actions C_t by extracting the child nodes of s_t^* in G:

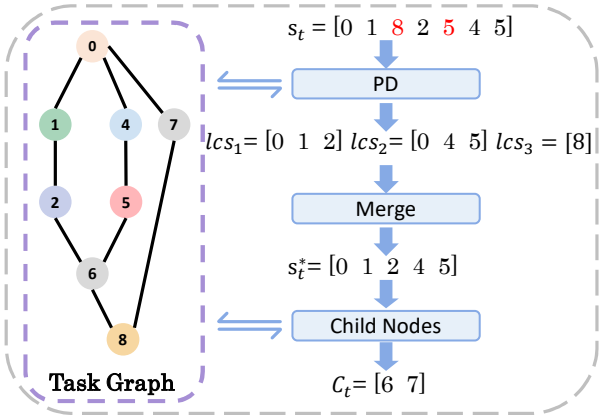

Figure 3. **Overview of the Potential Action Prediction Block.** Using Dynamic Programming (DP), this module identifies all longest common subsequences (lcs) from the executed action sequence s_t via the task graph G. These lcs are interconnected into a unified subgraph, forming the filtered sequence s_t^*. Reachable child nodes from G are then extracted as valid next actions C_t.

$$C_t = (\bigcup_{a \in s_t^*} A[a]) \setminus s_t^*, \tag{4}$$

where $\bigcup_{a \in s_t^*} A[a]$ aggregates all successors of nodes in s_t^* from the adjacency list A of task graph G, and $\setminus s_t^*$ excludes already executed actions. For instance, in Fig. 3, the child nodes of s_t^* include nodes 6 and 7, $C_t = [6, 7]$. This process ensures C_t robustly represents all potential valid next actions, enhancing error detection in complex procedural workflows. More details can be found in supplementary.

3.5. Representation Reconstruction Block

We propose the Representation Reconstruction Block (RRB) to generate normal action representations for valid next actions, as shown in part (b) of Fig. 2. Starting with frame-wise features $F_{1:ed_{t-1}} = \{f_i\}_{i=1}^{ed_{t-1}}$ up to time t, we apply dilated convolution to capture long-range dependencies, yielding $F_{1:ed_{t-1}}^{\text{conv}}$.

Then, we use local cross-attention to align the valid actions with the temporal context. Here, C_t serves as the query, while $F_{1:ed_{t-1}}^{\text{conv}}$ provides the keys and values. This mechanism generates a contextual feature for each valid action $y_{t,i} \in C_t$. To enhance robustness, we adopt a cluster-residual prediction approach. Specifically, for each valid action $y_{t,i}$, we predict a residual $r_{t,i}$ that refines the cluster center $c_{t,i}$ of the corresponding action class. The normal action representation $f_{t,i}^{\text{normal}}$ is then computed as:

$$r_{t,i} = \mathcal{E}_{\text{res}}(y_{t,i}, F_{1:ed_{t-1}}^{\text{conv}}), \tag{5}$$

$$f_{t,i}^{\text{normal}} = c_{t,i} + r_{t,i}, \tag{6}$$

where $\mathcal{E}_{\text{res}}$ denotes the residual prediction operation via cross-attention, and $c_{t,i}$ is the precomputed cluster center for action class $y_{t,i}$, derived from normal training samples. The resulting set of representations, denoted $f_t^{\text{normal}} = \{f_{t,i}^{\text{normal}}\}_{i=1}^{|C_t|}$, encapsulates all valid next actions:

$$f_t^{\text{normal}} = \mathcal{E}(C_t, F_{1:ed_{t-1}}), \tag{7}$$

where $\mathcal{E}$ integrates all operations within this block. This approach leverages the cluster center as a stable baseline while adapting to context-specific variations through the residual.

3.6. Normal Action Representation Alignment and Conformity Assessment

To assess action conformity and detect potential errors, we propose the Representation Matching Block (RMB), as illustrated in part (c) of Fig. 2. This component measures the deviation between normal action representations f_t^{normal} and ongoing action features f_t^{action}. Specifically, RMB evaluates the alignment between the ongoing action feature f_t^{action} and each potential normal action representation feature $f_{t,i}^{\text{normal}}$ for valid next actions. We calculate the Euclidean distance $d_{t,i}$ as follows:

$$d_{t,i} = \|f_t^{\text{action}} - f_{t,i}^{\text{normal}}\|_2. \tag{8}$$

To identify the best alignment, we select the smallest of these distances (*i.e.*, $d_t^{\text{min}} = \min_i d_{t,i}$) as the top matching of ongoing action to any normal action representation.

Error Detection Criterion. To determine if an ongoing action deviates from all potential normal action representations, we apply a threshold $\theta(y_t)$ specific to each action class y_t, based on the distribution of alignment distances in normal samples. An action is flagged as an error if:

$$\text{Error} = \begin{cases} 1, & \text{if } d_t^{\text{min}} > \theta(y_t), \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

If d_t^{min} exceeds $\theta(y_t)$, this indicates that the ongoing action at time t does not conform to any normal action representation, thereby identifying it as an error. If it remains within the threshold, the ongoing action is considered to conform to an expected, normal action.

3.7. Training Strategy and Objective Function

In this section, we outline the model training process and our objective function.

During training, segments predicted by the Action Segmentation Model may include inaccuracies, such as incomplete or incorrect boundaries. To filter these samples, we calculate an overlap ratio R_{overlap} between each predicted segment S_{pred} and its closest ground truth segment S_{GT}. The overlap ratio R_{overlap} is defined as:

$$R_{\text{overlap}} = \frac{|S_{\text{pred}} \cap S_{\text{GT}}|}{|S_{\text{pred}}|}. \tag{10}$$

We filter the segment based on whether R_{overlap} meets a predefined threshold τ:

$$S_{\text{filtered}} = \begin{cases} 0, & \text{if } R_{\text{overlap}} < \tau \quad \text{(exclude)}, \\ 1, & \text{if } R_{\text{overlap}} \geq \tau \quad \text{(retain)}. \end{cases} \tag{11}$$

For retained segments, the assigned label may still need refinement. We assign each action segment a representative label y_t^* by selecting the most frequently occurring ground truth label within the frames of the segment. Let

$$L_t = \{y_i^{\text{GT}} \mid i \in \text{segment}\}, \tag{12}$$

where y_i^{GT} is the ground truth label for the i-th frame within the segment. Then, we define y_t^* as:

$$y_t^* = \arg\max_y \text{count}(y, L_t), \tag{13}$$

where $\arg\max$ identifies the label y that appears most frequently in L_t.

Since each training sample video corresponds to a single target action at time t in C_t, we omit the Potential Action Prediction Block (PAPB) and set the query size of the Representation Reconstruction Block (RRB) to one during training. Using features from Sec. 3.3, we obtain frame-wise features $F_{1:ed_{t-1}} = \{f_i\}_{i=1}^{ed_{t-1}}$ up to time t, as well as the action feature f_t^{action}. Based on C_t and $F_{1:ed_{t-1}}$, our method then predicts an normal action representation vector f_t^{normal} for the action at time t:

$$f_t^{\text{normal}} = \mathcal{E}(C_t, F_{1:ed_{t-1}}), \tag{14}$$

where $\mathcal{E}$ represents the operations within the Representation Reconstruction Block (Sec. 3.3).

Since C_t contains only one action during training, f_t^{normal} includes only one expected representation, $f_{t,1}^{\text{normal}}$. Our optimization objective then minimizes the distance between this normal action representation and the actual action feature f_t^{action}, as shown below:

$$\mathcal{L}_{\text{normal}} = \left\|f_{t,1}^{\text{normal}} - f_t^{\text{action}}\right\|^2. \tag{15}$$

4. Experiments

In this section, we first introduce our experimental setup in Sec. 4.1, followed by the evaluation metrics used to assess the performance of our method in Sec. 4.2. Finally, we conduct ablation studies in Sec. 4.3 to demonstrate the effectiveness of our proposed AMNAR framework.

4.1. Experimental Setup

Datasets. We conduct experiments on the EgoPER [16], HoloAssist [38] and CaptainCook4D[28] datasets. **The EgoPER dataset** is an egocentric video dataset with five cooking tasks (*i.e.*, Quesadilla, Qatmeal, Pinwheel, Coffee and Tea). It includes 386 videos (28 hours) with both normal and erroneous executions, together with frame-level action and error annotations. **The HoloAssist dataset** features 166 hours of video from 350 instructor-performer pairs completing real-world tasks (e.g., furniture assembly, device operation). **The CaptainCook4D dataset** is an egocentric dataset of cooking activities, comprising 384 recordings of 24 recipes. It captures both normal and error executions, with step annotations and seven error types.

Table 1. **Comparison with existing methods on the EgoPER dataset for each task and the average over all tasks.**

Methods	Quesadilla		Oatmeal		Pinwheel		Coffee		Tea		All	
	EDA	AUC	EDA	AUC	EDA	AUC	EDA	AUC	EDA	AUC	EDA	AUC
Random	19.9	50.0	11.8	50.0	15.7	50.0	8.20	50.0	17.0	50.0	14.5	50.0
HF^2-VAD [24]	34.5	62.6	25.4	62.3	29.1	52.7	10.0	59.6	36.6	62.1	27.1	59.9
HF^2-VAD + SSPCAB [30]	30.4	60.9	25.3	61.9	33.9	51.7	10.0	60.1	35.4	63.2	27.0	59.6
S3R [40]	52.6	51.8	47.8	61.6	50.5	52.4	16.3	51.0	47.8	57.9	43.0	54.9
EgoPED[16]	**62.7**	65.6	51.4	65.1	59.6	55.6	55.3	58.3	56.0	**66.0**	57.0	62.0
AMNAR (Ours)	61.4	**71.9**	**65.0**	**75.4**	**65.0**	**65.4**	**73.5**	**67.8**	**57.0**	61.9	**64.4**	**68.5**

Table 2. **Comparison on HoloAssist dataset.** Lacking action segmentation annotations, we train with fine-grained noun and verb labels. "Noun" and "Verb" denote ASM training with only noun or verb annotations, respectively; "All" averages both results.

Methods	Noun		Verb		All	
	EDA	AUC	EDA	AUC	EDA	AUC
Random	48.3	52.4	49.8	48.7	49.0	50.6
EgoPED[16]	65.2	56.1	67.2	54.3	66.2	55.2
AMNAR (Ours)	**67.2**	**56.8**	**72.6**	**56.2**	**69.9**	**56.5**

Table 3. **Comparison on the CaptainCook4D dataset.**

Methods	Precision	EDA	AUC
Random	49.9	49.7	51.2
EgoPED [16]	56.5	69.8	54.9
AMNAR (Ours)	**66.8**	**72.3**	**60.2**

Evaluation Metrics. Following prior work [16], we evaluate models with **Error Detection Accuracy (EDA)** and **Area Under the Curve (AUC)**. **EDA** measures the accuracy in identifying both erroneous and normal segments, reflecting the model's overall accuracy in error detection at the segment level. **AUC** evaluates the ability of the model to distinguish between errors and non-errors by comparing true and false positive rates across varying thresholds.

Implementation Details. Following EgoPED[16], we adopt I3D [1] for video feature extraction and ActionFormer [46] for action segmentation, with the segmentation model pretrained before joint training with other components. For EgoPER, we use provided task graphs, while for HoloAssist and CaptainCook4D, we construct task graphs from training sequences. In the Representation Reconstruction Block (Sec. 3.5), actions in C_t are represented by their class cluster centers. The error detection threshold $\theta(y_t)$ (Sec. 3.6) is set at the 0.85 quantile of the distance distribution from normal training instances, with an overlap threshold $\tau = 0.6$. More details are in the supplementary.

4.2. Comparisons with SoTA Methods

We compare our AMNAR with SoTA error detection approaches [16, 24, 30, 40] on the EgoPER[16], HoloAssist[38] and CaptainCook4D[28] datasets.

As shown in Tab. 1, our results on the EgoPER dataset outperform all methods, with AMNAR achieving an average **7.4%** improvement in EDA and **6.5%** in AUC over EgoPED [16]. For the HoloAssist dataset (Tab. 2), AMNAR improves EDA by **3.7%** and AUC by **1.3%**. On the CaptainCook4D dataset (Tab. 3), AMNAR surpasses EgoPED with improvements of **10.3%** in Precision, **2.5%** in EDA, and **5.3%** in AUC, demonstrating robustness across diverse procedural tasks with complex action sequences.

4.3. Ablation Studies

In this section, we conduct a series of **ablation studies** on the EgoPER dataset [16] to evaluate the contributions of each component in our AMNAR framework.

Potential Action Prediction Block (PAPB). The PAPB identifies valid next actions in procedural tasks, enabling AMNAR to model multiple normal action representations post-sequence, as outlined in the method section. We evaluate its role by comparing "AMNAR" with "Random Selection", a variant using random action selection instead of PAPB's contextual prediction. As shown in Tab. 4, AMNAR with PAPB boosts EDA by 5.8% and AUC by 5.5% over the random variant, proving that task-informed prediction enhances error detection in complex workflows.

Representation Reconstruction Block (RRB). The RRB generates context-aware normal action representations for PAPB-predicted actions, as described earlier. We test its impact by comparing "AMNAR" against "w/o PAPB & RRB", which omits RRB and uses past action features with self-attention. Tab. 4 shows AMNAR with RRB improves EDA by 4.4% and AUC by 4.5% over this variant, confirming RRB's role in enhancing error detection through tailored representations.

Action Representation Types. AMNAR uses cluster-centered action representations to model the diversity of valid next actions by dynamically capturing their average distribution within each action class. Compared to text-based embeddings (e.g., BERT [3]), our ablation study (Tab. 5 (a)) shows superior performance, likely due to better alignment with the model's feature space, enhancing robustness to subtle execution errors.

Prediction Methods. We compare direct prediction, which generates full features from context, with residual-based

Table 4. **Ablation studies on the Potential Action Prediction Block (PAPB) and the Representation Reconstruction Block (RRB).** "w/o PAPB & RRB" excludes both PAPB and RRB, using only previously executed action features with a local self-attention mechanism. "Random Selection" indicates a variant where candidate actions are randomly selected instead of using PAPB to predict valid next actions.

Variants	Components		Quesadilla		Oatmeal		Pinwheel		Coffee		Tea		All	
	PAPB	RRB	EDA	AUC	EDA	AUC	EDA	AUC	EDA	AUC	EDA	AUC	EDA	AUC
w/o PAPB & RRB	✗	✗	58.7	68.7	58.9	68.4	55.5	56.6	67.6	64.7	**59.5**	61.5	60.0	64.0
Random Selection	✗	✓	56.9	67.2	60.1	72.5	50.2	56.2	71.7	61.7	54.1	57.3	58.6	63.0
AMNAR	✓	✓	**61.4**	**71.9**	**65.0**	**75.4**	**65.0**	**65.4**	**73.5**	**67.8**	57.0	**61.9**	**64.4**	**68.5**

Table 5. **Ablation studies across six configurations**: action representation types, prediction methods, video features, distance metrics, training sample selection strategies, task graph construction. The highlighted rows indicate our default implementation.

(a) Action Representation Types

Variants	EDA	AUC
Text-Based	**65.6**	64.1
Cluster-Centered	64.4	**68.5**

(b) Prediction Methods

Variants	EDA	AUC
Direct Prediction	64.2	62.2
Cluster Center Residual	**64.4**	**68.5**

(c) Visual Features

Variants	EDA	AUC
DINOv2 Feature	**71.0**	**69.2**
I3D Feature	64.4	68.5

(d) Distance Metrics

Variants	EDA	AUC
L1 Norm	62.4	63.9
Cosine Similarity	63.2	62.9
Euclidean	**64.4**	**68.5**

(e) Training Sample Selection Strategies

Variants	EDA	AUC
Training on GT Segments	**69.3**	51.4
Training on ASM Output	65.1	65.0
Hybrid Training	64.4	**68.5**

(f) Task Graph Construction

Variants	EDA	AUC
Training Set Graph	**66.6**	66.1
Predefined Graph	64.4	**68.5**

prediction, where the model predicts deviations from a cluster center for each intention-aligned action class. Per Tab. 5 (b), the residual method boosts EDA by 0.2% and AUC by 6.3%. It enhances the learning of action-context relationships and sensitivity to subtle deviations from normal action distributions.

Distance Metrics. We compare L_1-norm, cosine similarity, and Euclidean distance for feature alignment. As shown in Tab. 5 (d), Euclidean distance outperforms others, achieving an EDA of 64.4% and AUC of 68.5%, against cosine similarity (63.2% EDA, 62.9% AUC) and L_1-norm (62.4% EDA, 63.9% AUC). Its sensitivity to magnitude and direction offers the best balance for error detection in complex procedural tasks.

Training Sample Selection Strategies. Effective training samples are key to robust error detection. We assess three approaches: (1) GT Segments, using ground-truth data for high accuracy but less resilience to ASM noise; (2) ASM Output, leveraging unfiltered ASM predictions for deployment alignment, risking label errors; and (3) Hybrid Training (Sec. 3.7), filtering ASM segments by an overlap threshold (τ) with GT, balancing accuracy and ASM consistency. Per Tab. 5 (e), Hybrid achieves the best AUC (68.5%) with a solid EDA (64.4%), optimizing robustness and precision.

Visual Features. Our AMNAR utilizes clip-level features from I3D for fair comparisons with existing works[16]. Besides, Fig.S3 (2nd row, rightest) of Supplementary shows that AMNAR can detect fine-grained errors with I3D features. Moreover, using more robust feature representations further helps error detection. To highlight this, we replace the I3D with DINOv2, and the results are in Tab. 5 (c).

Task Graph Construction. Following the previous works[4, 16, 33], AMNAR uses predefined task graphs.**Note that we can also construct task graphs from the training set.** Tab. 5 (f) shows that the impact of constructing the graph from the training set is mild. Also, experiments on HoloAssist (Tab. 2) and CaptainCook4D (Tab. 3) utilize task graphs constructed from the training set and gain significant performance improvement.

4.4. Influence of Previous Errors

In procedural tasks, errors can accumulate over time. Since our method relies on temporal context from previous actions, a natural question could be raised: *How do errors in previous actions affect the future reconstruction of subsequent normal action representations?*

To address this, we conduct an ablation study to evaluate the robustness of AMNAR when prior errors are present in the action sequence. As shown in Tab. 6, we divide the test samples into two groups: the "w/o previous errors" group, which includes the first error in a sequence along with prior normal actions, and the "w previous errors" group, which covers actions following one or more prior errors. The results indicate that multiple prior errors cause a mild reduction in error detection performance. The "w/o previous errors" group achieves a higher EDA (70.8%) compared to the "w previous errors" group (64.3%), likely due to fewer distortions in the temporal context. However, the AUC remains comparable (67.3% vs. 68.0%), suggesting that AMNAR retains reasonable robustness even when prior errors are present. This demonstrates that, while prior errors intro-

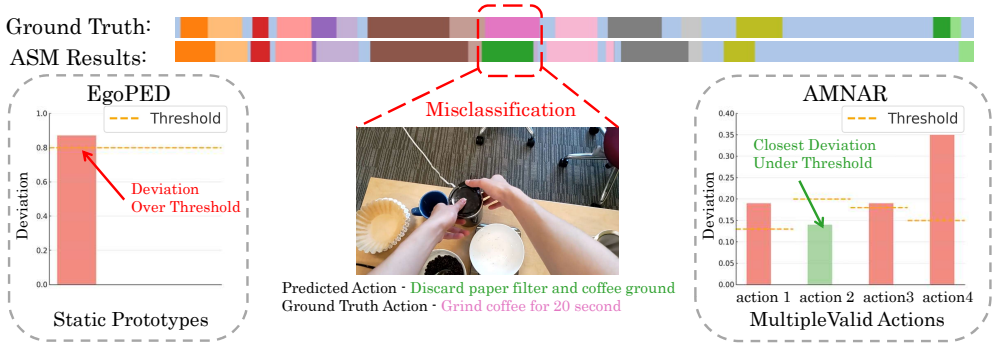

Figure 4. Error detection when the Action Segmentation Model (ASM) misclassifies an action. AMNAR correctly identifies the action as normal. In contrast, the EgoPED framework incorrectly detects a false positive.

Table 6. **Influence of previous errors.** Previous errors cause only a mild reduction in error detection performance.

Variants	EDA	AUC
w/o previous errors	**70.8**	67.3
w previous errors	64.3	**68.0**

Table 7. **EDA of non-deterministic actions in EgoPER dataset.** AMNAR consistently outperforms baselines, excelling in handling complex, non-deterministic action sequences.

Methods&Variants	Quesadilla	Oatmeal	Pinwheel	Coffee	Tea	All
EgoPED[16]	**74.0**	65.5	56.5	65.3	64.8	65.2
AMNAR (w/o PAPB & RRB)	60.2	61.4	61.3	72.5	63.4	63.8
AMNAR (Random)	57.9	63.8	61.7	71.5	57.6	62.5
AMNAR	73.8	**75.5**	**66.8**	**76.7**	**75.6**	**73.7**

duce some challenges, our adaptive representation strategy mitigates their impact effectively.

4.5. Ability of Handling Multiple Valid Next Actions

We provide more discussions about our AMNAR on the ability of handling multiple valid next actions. On the one hand, on the task of "coffee", which contains the most complex action branching patterns, our AMNAR achieves significant performance improvement of EDA by 18.2% and AUC by 9.5% according to Tab. 1.

On the other hand, we conduct an evaluation on those non-deterministic actions, which are preceded by actions with multiple valid next options. Results in Tab. 7 demonstrate AMNAR achieves a top EDA of 73.7% overall and 76.7% for "coffee", outperforming EgoPED and its variants, underscoring its strength in managing complex, non-deterministic sequences. Details of frequency analysis of multiple valid next actions and definition of non-deterministic actions are in Supplementary B.4 & B.5.

4.6. Discussions on Action Segmentation Module

We further present the Action Segmentation (AS) results. Although AMNAR and EgoPED employ the same AS module for fair comparison, AMNAR consistently outperforms EgoPED, as shown in Tab. 8. This performance gap arises due to differences in feature utilization—EgoPED modifies features through clustering and contrastive learning,

Table 8. **Analysis of Action Segmentation Module. Left**: Our AMNAR outperforms EgoPED on action segmentation task. **Right**: EgoPED under-performs our AMNAR even if used with action segmentation results of AMNAR (noted it by *).

Methods	IoU	Edit	F1@0.5	Acc
EgoPED	44.6	61.3	47.5	68.5
AMNAR	**56.3**	**69.4**	**57.3**	**75.3**

Methods	Avg. EDA	Avg. AUC
EgoPED*	63.1	61.9
AMNAR	**64.4**	**68.5**

whereas AMNAR uses them as input for reconstruction. Notably, even when EgoPED leverages AMNAR's segmentation results (denoted as EgoPED* in Tab. 8), it still underperforms in Error Detection (ED). This demonstrates that AMNAR's superior ED performance stems from its innovative design, beyond mere improvements in AS.

4.7. Visualization

As shown in Fig. 4, a "coffee" task sample demonstrates resilience of AMNAR to ASM misclassification. The ground truth action, "Grind coffee for 20 seconds," is erroneously labeled by ASM as "Discard paper filter and coffee grounds." EgoPED, hindered by an incorrect prototype, misclassifies it as an error. In contrast, AMNAR employs multiple normal action representations, selecting the top matching one to accurately classify the action as normal. This underscores robustness of AMNAR in handling label ambiguity and varied action sequences. More visualization samples are in Section C of supplementary material.

5. Conclusion

In this work, we uncover a critical limitation in existing error detection approaches: their inability to effectively handle scenarios of multiple valid next actions. To address this, we develop the Adaptive Multiple Normal Action Representation (AMNAR) framework, which dynamically predicts and reconstructs representations for all valid next actions. Through comprehensive experiments—including comparative analyses, ablation studies, and evaluations of non-deterministic actions across three datasets—we confirm the effectiveness of AMNAR. We believe this adaptive, multi-representation strategy could improve error detection and contribute to advancements in broader action understanding fields.

Acknowledgement

This work was supported partially by NSFC(92470202, U21A20471), National Key Research and Development Program of China (2023YFA1008503), Guangdong NSF Project (No. 2023B1515040025). The authors thank anonymous reviewers and ACs for their constructive suggestions.

References

[1] Joao 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*, pages 6299–6308, 2017. 1, 4, 6

[2] Yu Cheng, Quanfu Fan, Sharath Pankanti, and Alok Choudhary. Temporal sequence modeling for video event detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2227–2234, 2014. 1

[3] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 6

[4] Guodong Ding, Fadime Sener, Shugao Ma, and Angela Yao. Every mistake counts in assembly. *arXiv preprint arXiv:2307.16453*, 2023. 1, 2, 7

[5] Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation. *arXiv preprint arXiv:2410.00371*, 2024. 1

[6] 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

[7] Alessandro Flaborea, Guido Maria D'Amely di Melendugno, Leonardo Plini, Luca Scofano, Edoardo De Matteis, Antonino Furnari, Giovanni Maria Farinella, and Fabio Galasso. Prego: online mistake detection in procedural egocentric videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18483–18492, 2024. 1, 2

[8] 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. 2

[9] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1705–1714, 2019. 3

[10] Dayoung Gong, Joonseok Lee, Manjin Kim, Seong Jong Ha, and Minsu Cho. Future transformer for long-term action anticipation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3052–3061, 2022. 1

[11] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19383–19400, 2024. 1, 2

[12] Hongji Guo, Nakul Agarwal, Shao-Yuan Lo, Kwonjoon Lee, and Qiang Ji. Uncertainty-aware action decoupling transformer for action anticipation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18644–18654, 2024. 1

[13] Yuto Haneji, Taichi Nishimura, Hirotaka Kameko, Keisuke Shirai, Tomoya Yoshida, Keiya Kajimura, Koki Yamamoto, Taiyu Cui, Tomohiro Nishimoto, and Shinsuke Mori. Egooops: A dataset for mistake action detection from egocentric videos with procedural texts. *arXiv preprint arXiv:2410.05343*, 2024. 2

[14] Youngkyoon Jang, Brian Sullivan, Casimir Ludwig, Iain Gilchrist, Dima Damen, and Walterio Mayol-Cuevas. Epictent: An egocentric video dataset for camping tent assembly. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 2

[15] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017. 1

[16] Shih-Po Lee, Zijia Lu, Zekun Zhang, Minh Hoai, and Ehsan Elhamifar. Error detection in egocentric procedural task videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18655–18666, 2024. 2, 3, 5, 6, 7, 8

[17] Shijie Li, Yazan Abu Farha, 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*, 45(6):6647–6658, 2020. 1

[18] Yuan-Ming Li, Wei-Jin Huang, An-Lan Wang, Ling-An Zeng, Jing-Ke Meng, and Wei-Shi Zheng. Egoexo-fitness: Towards egocentric and exocentric full-body action understanding. In *European Conference on Computer Vision*, 2024. 2

[19] Yuan-Ming Li, An-Lan Wang, Kun-Yu Lin, Yu-Ming Tang, Ling-An Zeng, Jian-Fang Hu, and Wei-Shi Zheng. Technocoach: Towards technical keypoint-aware descriptive action coaching. *arXiv preprint arXiv:2411.17130*, 2024. 1

[20] Yuan-Ming Li, Ling-An Zeng, Jing-Ke Meng, and Wei-Shi Zheng. Continual action assessment via task-consistent score-discriminative feature distribution modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

[21] Kun-Yu Lin, Jia-Run Du, Yipeng Gao, Jiaming Zhou, and Wei-Shi Zheng. Diversifying spatial-temporal perception for video domain generalization. *Advances in Neural Information Processing Systems*, 36:56012–56026, 2023.

[22] Kun-Yu Lin, Henghui Ding, Jiaming Zhou, Yu-Ming Tang, Yi-Xing Peng, Zhilin Zhao, Chen Change Loy, and Wei-

Shi Zheng. Rethinking clip-based video learners in cross-domain open-vocabulary action recognition. *arXiv preprint arXiv:2403.01560*, 2024.

[23] Kun-Yu Lin, Jiaming Zhou, and Wei-Shi Zheng. Human-centric transformer for domain adaptive action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 1

[24] Zhian Liu, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. A hybrid video anomaly detection framework via memory-augmented flow reconstruction and flow-guided frame prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13588–13597, 2021. 3, 6

[25] Michele Mazzamuto, Antonino Furnari, and Giovanni Maria Farinella. Eyes wide unshut: Unsupervised mistake detection in egocentric video by detecting unpredictable gaze. *arXiv preprint arXiv:2406.08379*, 2024. 2

[26] Himangi Mittal, Nakul Agarwal, Shao-Yuan Lo, and Kwonjoon Lee. Can't make an omelette without breaking some eggs: Plausible action anticipation using large video-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18580–18590, 2024. 1

[27] Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Qifan Zhang, Jikai Wang, Vasundhara Komaragiri, Nicholas Ruozzi, Eric Ragan, et al. Put on your detective hat: What's wrong in this video? 2

[28] Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Jikai Wang, Qifan Zhang, Vasundhara Komaragiri, Eric Ragan, Nicholas Ruozzi, et al. Captaincook4d: A dataset for understanding errors in procedural activities. *arXiv preprint arXiv:2312.14556*, 2023. 2, 5, 6

[29] Alexander Richard and Juergen Gall. Temporal action detection using a statistical language model. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3131–3140, 2016. 1

[30] Nicolae-Cătălin Ristea, Neelu Madan, Radu Tudor Ionescu, Kamal Nasrollahi, Fahad Shahbaz Khan, Thomas B Moeslund, and Mubarak Shah. Self-supervised predictive convolutional attentive block for anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13576–13586, 2022. 3, 6

[31] Nicolae-C Ristea, Florinel-Alin Croitoru, Radu Tudor Ionescu, Marius Popescu, Fahad Shahbaz Khan, Mubarak Shah, et al. Self-distilled masked auto-encoders are efficient video anomaly detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15984–15995, 2024. 3

[32] Tim J Schoonbeek, Tim Houben, Hans Onvlee, Fons Van der Sommen, et al. Industreal: A dataset for procedure step recognition handling execution errors in egocentric videos in an industrial-like setting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4365–4374, 2024. 2

[33] Luigi Seminara, Giovanni Maria Farinella, and Antonino Furnari. Differentiable task graph learning: Procedural activity representation and online mistake detection from egocentric videos. *arXiv preprint arXiv:2406.01486*, 2024. 1, 2, 7

[34] Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21096–21106, 2022. 2

[35] An-Lan Wang, Kun-Yu Lin, Jia-Run Du, Jingke Meng, and Wei-Shi Zheng. Event-guided procedure planning from instructional videos with text supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13565–13575, 2023. 1

[36] An-Lan Wang, Nuo Chen, Kun-Yu Lin, Yuan-Ming Li, and Wei-Shi Zheng. Task-oriented 6-dof grasp pose detection in clutters. *arXiv preprint arXiv:2502.16976*, 2025. 1

[37] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 1

[38] Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, et al. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20270–20281, 2023. 1, 2, 5, 6

[39] Zejia Weng, Xitong Yang, Ang Li, Zuxuan Wu, and Yu-Gang Jiang. Open-vclip: Transforming clip to an open-vocabulary video model via interpolated weight optimization. In *International Conference on Machine Learning*, pages 36978–36989. PMLR, 2023. 1

[40] Jhih-Ciang Wu, He-Yen Hsieh, Ding-Jie Chen, Chiou-Shann Fuh, and Tyng-Luh Liu. Self-supervised sparse representation for video anomaly detection. In *European Conference on Computer Vision*, pages 729–745. Springer, 2022. 6

[41] Angchi Xu, Ling-An Zeng, and Wei-Shi Zheng. Likert scoring with grade decoupling for long-term action assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3232–3241, 2022. 1

[42] Zhiwei Yang, Jing Liu, Zhaoyang Wu, Peng Wu, and Xiaotao Liu. Video event restoration based on keyframes for video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14592–14601, 2023. 3

[43] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. *arXiv preprint arXiv:2110.08568*, 2021. 1

[44] Ling-An Zeng and Wei-Shi Zheng. Multimodal action quality assessment. *IEEE Transactions on Image Processing*, 2024. 1

[45] Ling-An Zeng, Fa-Ting Hong, Wei-Shi Zheng, Qi-Zhi Yu, Wei Zeng, Yao-Wei Wang, and Jian-Huang Lai. Hybrid dynamic-static context-aware attention network for action assessment in long videos. In *Proceedings of the 28th ACM international conference on multimedia*, pages 2526–2534, 2020. 1

[46] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *European Conference on Computer Vision*, pages 492–510. Springer, 2022. 1, 6

[47] Dian Zheng, Xiao-Ming Wu, Shuzhou Yang, Jian Zhang, Jian-Fang Hu, and Wei-Shi Zheng. Selective hourglass mapping for universal image restoration based on diffusion model. In *CVPR*, 2024. 1

[48] Dian Zheng, Xiao-Ming Wu, Zuhao Liu, Jingke Meng, and Wei-shi Zheng. Diffuvolume: Diffusion model for volume based stereo matching. *IJCV*, 2025. 1

[49] Jiaming Zhou, Junwei Liang, Kun-Yu Lin, Jinrui Yang, and Wei-Shi Zheng. Actionhub: a large-scale action video description dataset for zero-shot action recognition. *arXiv preprint arXiv:2401.11654*, 2024. 1

[50] Jiaming Zhou, Teli Ma, Kun-Yu Lin, Zifan Wang, Ronghe Qiu, and Junwei Liang. Mitigating the human-robot domain discrepancy in visual pre-training for robotic manipulation. *arXiv preprint arXiv:2406.14235*, 2024. 1