CaptainCook4D: A Dataset for Understanding Errors in Procedural Activities

Rohith Peddi [+] [*] Shivvrat Arya [+] Bharath Challa [+] Likhitha Pallapothula [+]

Akshay Vyas [+] Bhavya Gouripeddi [+] Qifan Zhang [+] Jikai Wang [+]

Vasundhara Komaragiri [+] Eric Ragan [×] Nicholas Ruozzi [+] Yu Xiang [+] Vibhav Gogate [+]

Abstract

Following step-by-step procedures is an essential component of various activities carried out by individuals in their daily lives. These procedures serve as a guiding framework that helps to achieve goals efficiently, whether it is assembling furniture or preparing a recipe. However, the complexity and duration of procedural activities inherently increase the likelihood of making errors. Understanding such procedural activities from a sequence of frames is a challenging task that demands an accurate interpretation of visual information and the ability to reason about the structure of the activity. To this end, we collect a new egocentric 4D dataset **CaptainCook4D** comprising 384 recordings (94.5 hours) of people performing recipes in real kitchen environments. This dataset consists of two distinct types of activities: one in which participants adhere to the provided recipe instructions and another in which they deviate and induce errors. We provide 5.3K step annotations and 10K fine-grained action annotations and benchmark the dataset for the following tasks: error recognition, multi-step localization and procedure learning[2].

1 Introduction

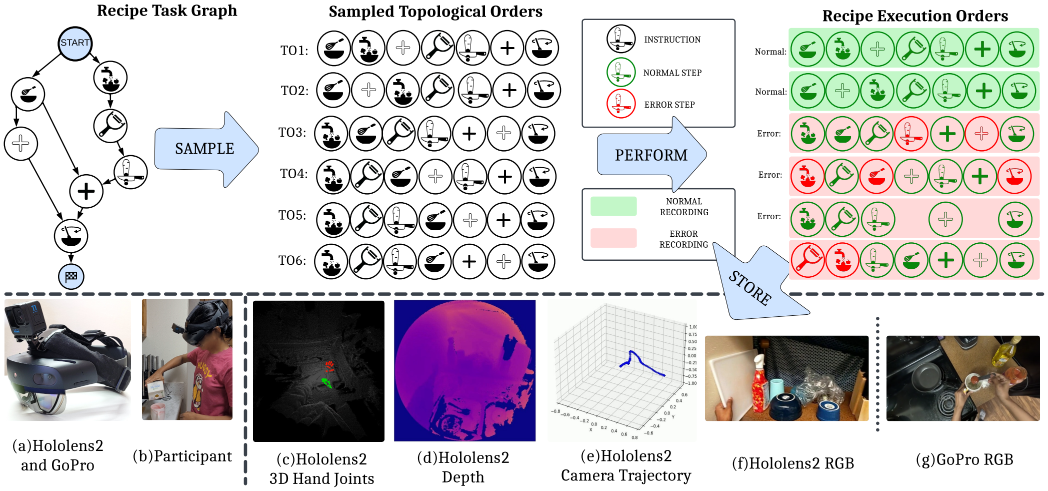
Have you ever excitedly prepared your favourite meal after a long day, only to be disappointed upon realizing you missed a key ingredient? Such scenarios are common because performing long step-by-step procedures increases the likelihood of making errors. While some errors are harmless and can be corrected with little consequence, others can have detrimental consequences, particularly those that occur during medical procedures or complex chemical experiments. Therefore, there is a pressing need to build AI systems that can guide users in performing procedural activities [7].

A key problem we need to solve in order to build such AI systems is **procedural activity understanding**, a challenging and multifaceted task that demands **interpreting what is happening** —specifically, determining whether the person is following the procedure correctly or making an error, **anticipating what will happen**, and **planning the course of action** to accomplish the goal. For effective interpretation, the system must be capable of recognizing and categorizing actions while assessing the current state of the environment. To anticipate what might happen next, it should be able to forecast actions right from the start of the interaction or even before it begins. Additionally, planning a course of action necessitates understanding the potential consequences of these actions. Numerous datasets have been developed to improve our understanding of procedural activities. However, these datasets only include videos of individuals performing step-by-step tasks correctly without making any errors.

[*] Corresponding Author , + = UT Dallas, × = University of Florida

[2] website: https://captaincook4d.github.io/captain-cook/

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks.



(a)Hololens2 and GoPro (b)Participant (c)Hololens2 3D Hand Joints (d)Hololens2 Depth (e)Hololens2 Camera Trajectory (f)Hololens2 RGB (g)GoPro RGB

Figure 1: **Overview.** Top: We constructed task graphs for the selected recipes. These graphs facilitated sampling topological orders (cooking steps) that participants followed to perform. During the execution of these steps, participants induced errors that were both **intentional** and **unintentional** in nature. Bottom Left: We present the sensors employed for data collection. Bottom Right: We describe the details of the modalities of the data collected while the participant performs the recipe.

But, for AI systems to effectively identify errors in procedural activities, it is essential to have datasets that include both normal and error videos along with corresponding error annotations (descriptions).

Contributions. We introduce an egocentric[3] 4D dataset designed to enhance AI systems' understanding of procedural activities and improve their ability to recognize and anticipate errors.

- Our dataset features participants performing recipes in real kitchen environments (Fig. 1). It includes two distinct types of activities: one where the participants follow the given recipe guidelines and another where they deviate (intentionally or unintentionally), making errors.
- We provide annotations for (a) Start and end times for each step of the recipe, (b) Start and end times for each fine-grained action/interaction for 20% of the collected data, and (c) Detailed descriptions of the errors made by participants, which allowed us to compile a comprehensive overview of different error categories along with their brief explanations.
- We provide baselines for the following procedure understanding tasks: (a) Error Recognition (supervised and zero-shot), (b) Multi-Step Localization, and (c) Procedure Learning.

2 Related Work

Understanding procedural activities with errors has witnessed significant traction recently and spurred the development of new datasets (see Table 1) that aid in developing novel approaches to recognize errors. Our dataset sets itself apart from others[4] by four distinctive features: (1) **Domain:** While others address errors during assembly and disassembly, we focus on cooking activities[5]. (2) **Environment:** Unlike lab environments, we collected our dataset in real kitchen environments. (3) **Multimodal capabilities**, and (4) **Error diversity**. In this section, we briefly highlight the relevance of our dataset to the various tasks of interest and provide a comprehensive review of related work in Appendix A.

Temporal Action Localization (TAL) in videos aims to identify and classify temporal boundaries of action instances in long video sequences. TAL methods can be categorized into two primary approaches: two-stage and single-stage. Two-stage methods operate in a sequential manner by initially generating action proposals and subsequently classifying them. In contrast, single-stage

[3] An egocentric view despite ego motions helps minimize occlusions more effectively than exo-centric videos.

[4] To the best of our knowledge, we are the first to categorize and provide brief descriptions for the error types.

[5] Cooking activities are inherently complex and encompass several types of diverse cascading and non-cascading errors that can compound and often alter the state of the environment with no point of return.

Table 1: **Ours vs Current Procedural Datasets (with/without errors)**: Our dataset not only advances the study of procedural tasks found in existing literature but also enables a systematic analysis of errors in procedures.

Procedural	Errors	Dataset Name	Domain / Environment	Ego	Depth	Recorded	Error Labels	Errors Nature	Hours	Year
×	×	Epic-Kitchens [3]	Cooking / Real	✓	×	✓	-	-	100	2018
		Ego4D [41]	Daily-Life / Real	✓	✓	✓	-	-	3000	2023
✓	×	50 Salads [36]	Cooking / Real	×	✓	✓	-	-	4.5	2013
		Breakfast [25]	Cooking / Real	×	×	✓	-	-	77	2014
		MPII Cooking 2 [33]	Cooking / Real	×	×	✓	-	-	27	2015
		YouCook2[43]	Cooking / Real	×	×	×	-	-	176	2017
		EGTEA Gaze+ [26]	Cooking / Real	✓	×	✓	-	-	29	2020
		EgoProceL [1]	Assembly / Real, Lab	✓	×	✓	-	-	62	2022
✓	✓	EgoTV [32]	Cooking / Simulated	✓	×	-	✓	Intentional	168	2023
✓	✓	Assembly101 [10]	Toy Assembly / Lab	✓	×	✓	Partial*	Unintentional	53	2022
		CSV [28]	Chemistry / Lab	×	×	✓	×	Intentional	11.1	2022
		HoloAssist [40]	Assembly*/ Lab	✓	✓	✓	✓	Unintentional	166	2023
		IndustReal [34]	Toy Assembly / Lab	✓	✓	✓	✓	Int. and Unint.	5.8	2024
		ATA [17]	Toy Assembly / Lab	✓	×	✓	✓	Intentional	24.8	2024
✓	✓	**CaptainCook4D** (Ours)	Cooking / Real	✓	✓	✓	✓	Int. and Unint.	94.5	2024

methods streamline the process by simultaneously performing action localization and classification, thus integrating both tasks into a single step. Datasets such as ActivityNet [9], Ava [21], Thumos14 [24], Epic-Kitchens [4], and Ego4D [20], helped develop advanced methods for TAL. Supervised Multi-Step Localization (MSL) task, while similar to TAL, specifically targets procedural datasets.

Remark. Our dataset, featuring both normal and erroneous actions, offers a unique perspective and helps evaluate the robustness of the MSL(TAL) methods in handling actions with deviations (errors).

Error Recognition in videos aims to identify errors (deviations from procedure text) in procedural activities. It was introduced as mistake detection by Assembly-101 [10] where a multi-class classification problem was formulated to classify the given clip corresponding to a procedure as correct, mistake or correction. Anomaly detection, while closely related to error recognition, differentiates itself by using static cameras and backgrounds to identify abnormal behaviour. Recently, [15] proposed an online error recognition method. Using Vision-and-Language Models (VLMs), they predict future actions and compare these predictions with actual observations to recognize errors online[6].

Remark. Unlike assembly, cooking involves continuous changes in the shape and color of the ingredients thus making our dataset valuable for developing error recognition methods transferable to procedural activities in the medical sector or that involve performing complex chemical experiments.

Procedure Learning in videos aims to identify the key steps in long video sequences and determine their logical order to complete a procedural activity. Datasets such as CrossTask [44], COIN [37], EgoProceL [1], Egtea [11], Meccano [30], Epic-Tents [23], Cmu-Mmac [5], helped develop advanced methods for supervised, weakly supervised and self-supervised procedure learning task variants.

Remark. Videos in our dataset are characterized by a longer average step length, which presents a challenge for algorithms previously designed for the existing egocentric procedure learning datasets.

3 Data Collection

Sensors. We utilized a Hololens2 device and a GoPro Hero 11 camera mounted on the user's head (see Fig. 1) to capture the activity data. We built a custom tool using hl2ss [6] to capture data from the depth sensor, front RGB camera, microphone and the depth sensor of the hololens2 device. We additionally captured the processed head and hand tracking information provided by the HoloLens2.

Recipes. We curated a selection of 24 cooking recipes sourced from WikiHow (refer to Appendix C), specifically focusing on recipes with a preparation time of 30 minutes or less. These recipes encompassed a wide range of culinary traditions, showcasing the diversity of cooking styles in various cuisines. Our primary objective was to identify and capture potential errors that could arise from using various authentic cooking instruments in preparing recipes sampled from different cuisines.

[6] At the time of writing, the code for [15] was not released.

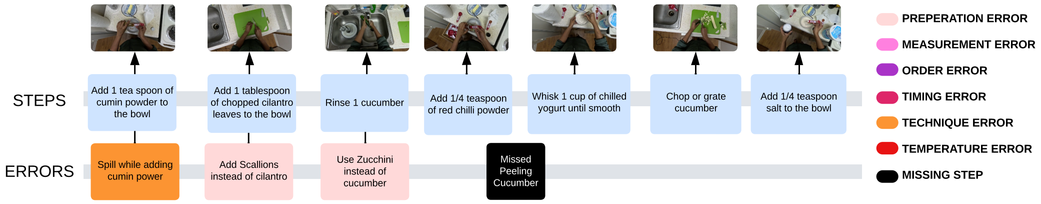

STEPS
ERRORS
Add 1 tea spoon of cumin powder to the bowl
Add 1 tablespoon of chopped cilantro leaves to the bowl
Rinse 1 cucumber
Add 1/4 teaspoon of red chilli powder
Whisk 1 cup of chilled yogurt until smooth
Chop or grate cucumber
Add 1/4 teaspoon salt to the bowl
Spill while adding cumin power
Add Scallions instead of cilantro
Use Zucchini instead of cucumber
Missed Peeling Cucumber
PREPERATION ERROR
MEASUREMENT ERROR
ORDER ERROR
TIMING ERROR
TECHNIQUE ERROR
TEMPERATURE ERROR
MISSING STEP

Figure 2: **Snapshots** of steps and recorded errors while preparing the recipe *Cucumber Raita*. Three of the four errors were intentional, but the participant missed the *Peeling* step **unintentionally**.

Task Graphs. visually represents the sequential steps required to complete a recipe. Each node in the task graph (for a recipe) corresponds to a step in a recipe, and a directed edge between a node x and a node y in the graph indicates that x must be performed before y. Thus, a task graph is a directed acyclic graph, with a topological order representing a valid recipe completion. To construct task graphs for selected recipes, we identified all the critical steps involved and determined their inter-dependencies, thus establishing a topological order of tasks (see website for final task graphs).

3.1 Protocol

Our dataset was compiled by eight participants[7] in 10 different kitchens. Each participant was provided a tablet-based recording interface accessible through a web browser, a GoPro and a Hololens2. Participants were instructed to adjust their GoPro cameras to capture footage in 4K resolution at 30 fps to ensure high-quality video. The HoloLens2 device was programmed to stream RGB frames at 360p resolution and 30 fps. It also streamed depth frames in Articulated Hand Tracking mode, referred to as *depth_ahat* mode. Besides visual data, the device also streamed three streams of IMU (Inertial Measurement Unit) sensor data and spatial data, capturing both head and hand poses[8].

Normal Recordings. A recording is classified as a **normal recording** when it is captured as the participant accurately follows the procedure described in the recipe. Participants are presented with one of the pre-constructed topological orders of the selected recipe[9], as determined by the task graphs. The participants then follow and perform each step from the topological order sequentially.

Error Recordings. A recording is classified as an **error recording** when it is captured while the individual deviates from the recipe's procedure, thereby inducing errors. Following the terminology used in scientific disciplines such as neuroscience [2] and chemistry, we will refer to deviations from procedures as *errors*[10]. Following [2, 14, 16], we classified common errors performed during a cooking activity into the following categories: (1) Preparation Error, (2) Measurement Error, (3) Technique Error, (4) Timing Error, (5) Temperature Error, (6) Missing Steps, and (7) Ordering Errors.

Error Induction. We developed three strategies[11] for participants to choose from, each tailored to perform the recipe in a specific environment. After choosing the strategy, participants were given detailed instructions on how to perform the recipes. We list the strategies presented to the participants (1) **Impromptu**: Participants were asked to induce errors while performing the recipe. Following the completion of each recording, participants used a web-based interface to update the errors they performed during each step. Due to the complex nature of cooking activities and the lack of experience of the participants in cooking, many errors induced in this strategy were **unintentional** (Figure 2 presents one such example). (2) **Disordered Steps**: Participants were given pre-prepared error scripts with missing steps and ordering errors. (3) **Induct Error**: Participants used a web-based interface to create an error script for each selected recipe recording. The modified recipe steps were displayed on a tablet, enabling participants to perform according to their scripted errors. In Fig. 4, we present both general statistics of the dataset and specific statistics of normal and error recordings.

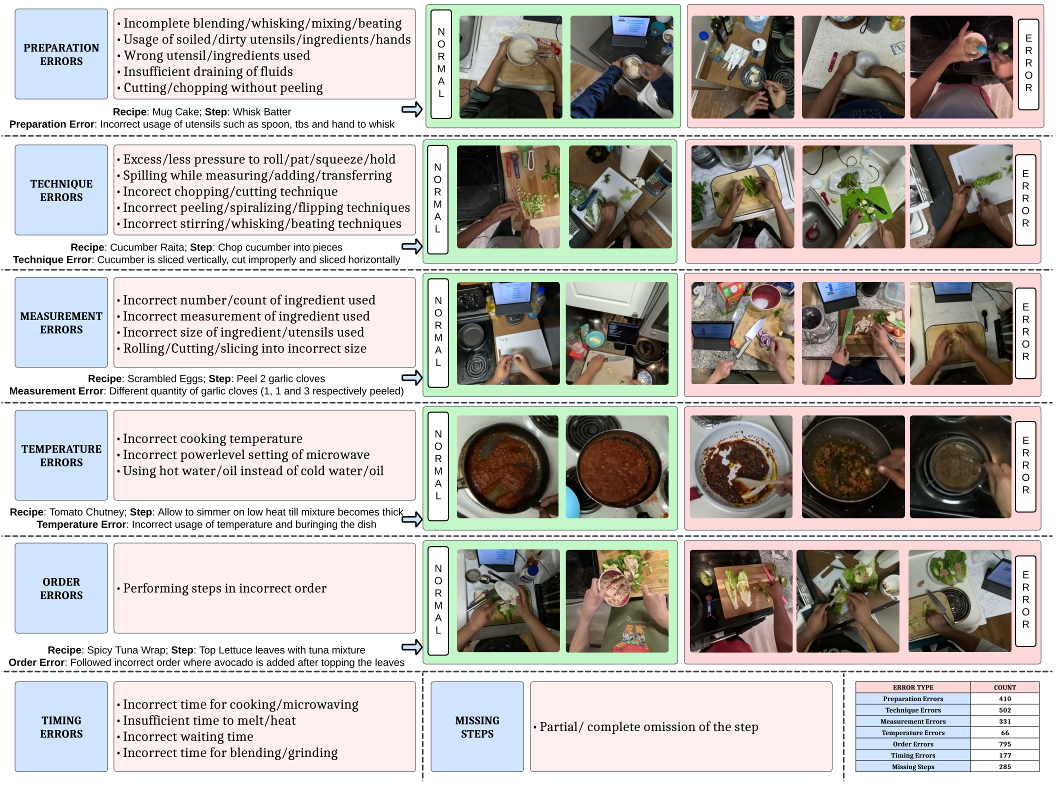
[7] During filming, participants were instructed to be alone in the kitchen environment and remove any items that could potentially identify them, such as personal portraits, mirrors, and smartwatches with portraits.

[8] The University of Florida IRB approved our protocol

[9] Utilizing the recording interface, each participant chose a recipe from the selected 24 WikiHow recipes.

[10] The term *errors* is synonymous with what the AI community typically calls *mistakes* (cf. [10]).

[11] The practice of using scripted videos for activity understanding [35] has inspired us to develop the strategies.


PREPARATION ERRORS
• Incomplete blending/whisking/mixing/beating
• Usage of soiled/dirty utensils/ingredients/hands
• Wrong utensil/ingredients used
• Insufficient draining of fluids
• Cutting/chopping without peeling
Recipe: Mug Cake; Step: Whisk Batter
Preparation Error: Incorrect usage of utensils such as spoon, tbs and hand to whisk
NORMAL
ERROR
TECHNIQUE ERRORS
• Excess/less pressure to roll/pat/squeeze/hold
• Spilling while measuring/adding/transferring
• Incorect chopping/cutting technique
• Incorrect peeling/spiralizing/flipping techniques
• Incorrect stirring/whisking/beating techniques
Recipe: Cucumber Raita; Step: Chop cucumber into pieces
Technique Error: Cucumber is sliced vertically, cut improperly and sliced horizontally
NORMAL
ERROR
MEASUREMENT ERRORS
• Incorrect number/count of ingredient used
• Incorrect measurement of ingredient used
• Incorrect size of ingredient/utensils used
• Rolling/Cutting/slicing into incorrect size
Recipe: Scrambled Eggs; Step: Peel 2 garlic cloves
Measurement Error: Different quantity of garlic cloves (1, 1 and 3 respectively peeled)
NORMAL
ERROR
TEMPERATURE ERRORS
• Incorrect cooking temperature
• Incorrect powerlevel setting of microwave
• Using hot water/oil instead of cold water/oil
Recipe: Tomato Chutney; Step: Allow to simmer on low heat till mixture becomes thick
Temperature Error: Incorrect usage of temperature and buringing the dish
NORMAL
ERROR
ORDER ERRORS
• Performing steps in incorrect order
Recipe: Spicy Tuna Wrap; Step: Top Lettuce leaves with tuna mixture
Order Error: Followed incorrect order where avocado is added after topping the leaves
NORMAL
ERROR
TIMING ERRORS
• Incorrect time for cooking/microwaving
• Insufficient time to melt/heat
• Incorrect waiting time
• Incorrect time for blending/grinding
MISSING STEPS
• Partial/ complete omission of the step

ERROR TYPE	COUNT
Preparation Errors	410
Technique Errors	502
Measurement Errors	331
Temperature Errors	66
Order Errors	795
Timing Errors	177
Missing Steps	285

Figure 3: **Error Categories.** Left: We present a categorization of participant-induced errors derived from the annotated error descriptions of the recordings. Right: We display frames captured from various recordings, highlighting correct and erroneous executions. Bottom Right: We present statistics on the error categories in the dataset derived from the compiled annotations of all recordings.

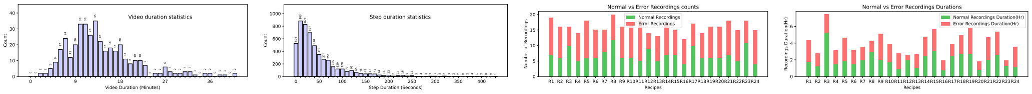


Figure 4: **Statistics.** Top: We present video and step duration statistics to the left & right respectively. Bottom: We present the total count and the durations of normal and error recordings for each recipe.

3.2 Data Annotation

We implemented a dual-layer review process to guarantee high-quality annotations. Each video was first annotated by the person who recorded it and then reviewed for accuracy by a second reviewer. Thus ensuring that all errors were correctly captured in the annotations corresponding to each step. Our annotations are structured to provide detailed insights into the recorded actions, facilitating both coarse-grained and fine-grained action analyses. Specifically, we offer the following annotations: (1) **Coarse-Grained Actions:** We mark the start and end times of each step in a recording of the recipe. (2) **Fine-Grained Actions:** For 20% of our data, we provide fine-grained action annotations to support semi/weakly supervised learning techniques for action recognition. (3) **Error Descriptions:** For each step, if an error occurs during its execution, we link its step annotation with the specific category of the error and a description of the error, thus enabling a comprehensive understanding.

Coarse-Grained Action/Step Annotations. We designed an interface for annotating steps in Label Studio[12]. Annotators are presented with this interface to mark each step's start and end times. Our coarse-grained actions/steps are significantly longer than a single fine-grained action and encompass

[12] https://labelstud.io/

multiple such fine-grained actions to perform the described step successfully. For example, to accomplish the step *{Chop a tomato}*, we include the following in the annotation (1) **Pre-conditional actions:** *{opening refrigerator, grabbing a polythene bag of tomatoes, taking a tomato, placing the tomato on cutting board, close fridge}* (2) **Post-conditional actions:** *{placing down the knife, grabbing the polythene bag of tomatoes, opening refrigerator and placing the bag in the refrigerator}*.

Fine-Grained Action Annotations. Inspired by the pause-and-talk [3], we have developed a web-based tool for fine-grained action annotations using Whisper [29] (for speech-to-text translation).

Error Category Annotations. Following each recording, participants were also asked to categorize errors performed in each step based on a set of guidelines. Specifically, we ask participants to broadly classify an error as a (1) *Preparation Error* when they use soiled/wrong ingredients or use different tools, (2) *Measurement Error* when they use wrongly measured ingredients, (3) *Timing Error* when they perform a step in shorter or longer duration than what is prescribed (e.g. Microwave for Microwave instead of 30 seconds) (4) *Temperature Error* when they set higher/lower power levels in the microwave or on a stove than what is prescribed (5) *Missing Step* when they omit to perform a step (6) *Technique Error* when they perform the required action incorrectly, leading to a wrong outcome than expected. (7) *Order Error* when they execute steps out of the required sequence. We compile and present the categorization of errors, their descriptions and visual illustrations in Fig. 3.

4 Experiments

Our experiments are designed to address the following questions: (Q1) What is the efficacy of transfer learning in recognizing errors? (Q2) How effective are current Vision Language Models (VLMs) in zero-shot error recognition? (Q3) How do state-of-the-art Multi-Step Localization methods perform on our dataset, particularly in terms of robustness to technique errors? (Q4) How do current self-supervised procedure learning methods in literature perform when applied to our dataset[13]?

Features. To answer the above questions we trained[14] our proposed baseline models on features obtained using pre-trained models such as 3D-ResNet [22], SlowFast [13], X3D [12], VideoMAE [38], Imagebind [18] and Omnivore [19] which were originally trained for video recognition tasks. Specifically, we split each video into 1-second sub-segments and extracted features to train models.

4.1 Error Recognition

This section answers questions (Q1) and (Q2); specifically, we address **Q1** by formulating the error recognition task as a *supervised binary classification* problem. We proposed three architectural variants (Fig. 5) as our baseline models and trained them using video/multimodal features. To address **Q2**, we employed a *prompt-and-predict* paradigm to recognize errors in activity recordings. Specifically, we formulated the problem as a Video Question Answering task (Fig. 6), crafted targeted question prompts using task graphs and error annotations(Fig.3); supplied these engineered prompts along with the videos as input to a VLM and evaluated its performance in zero-shot error recognition.

Supervised Error Recognition (SupervisedER). We utilized the features extracted using pre-trained models to train variants of our baseline binary classification models and evaluated trained models using the standard metrics such as accuracy, precision, recall, F1 and AUC (see Table 2). Specifically, we trained our models to classify each step of a video into one of two classes {*error(1)*, *normal(0)*}. We constructed two data splits, step and recording splits, for training error recognition models. For the step split, we first compiled a dataset of video segments corresponding to all steps of all recipes in the dataset. Then divided it into train, validation, and test subsets. For the recordings split, we compiled all the recordings of all recipes in the dataset and divided the dataset into train, validation, and test subsets. Using error annotations (Figure 3), we first generated class labels for all video segments corresponding to the recipe steps. Then, we assigned the class label corresponding to the step to all 1-second sub-segments within the step and trained baseline models. During inference, we assigned the majority class label of the sub-segments corresponding to a step as the label to that step.

We proposed three architectural variants as baselines:$\{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}$ (see Fig. 5). In $\mathcal{V}_1$, we used the extracted features and constructed labels as described above and trained a Multi-Layer Perceptron

[13] Characterized by longer step durations, refer App. C for a comparison across procedural activity datasets

[14] We present the details about hyperparameters used for training the proposed baseline models in Appendix B.

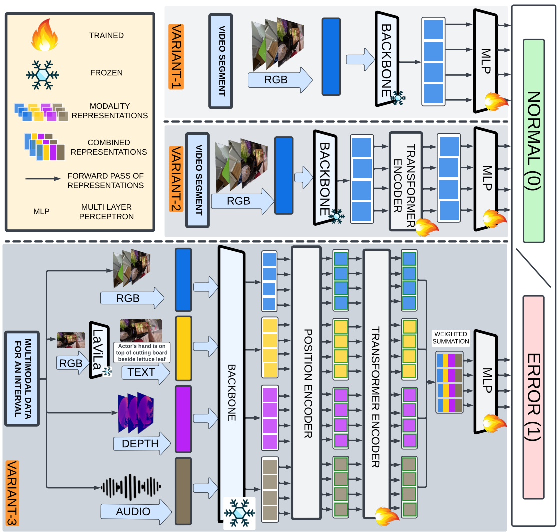


Figure 5: **SupervisedER** architectures of 3 baselines.

Split	Backbone	$\mathcal{V}_{\#}$	Modality	Acc	P	R	F1	AUC
$\mathcal{S}$	Omnivore	$\mathcal{V}_1$	V	71.09	66.07	14.86	24.26	75.7
		$\mathcal{V}_2$	V	69.96	51.56	59.84	55.39	75.7
	Slowfast	$\mathcal{V}_1$	V	33.54	31.88	90.6	47.16	63.06
		$\mathcal{V}_2$	V	68.09	47.69	24.9	32.72	67.18
	X3D	$\mathcal{V}_1$	V	68.34	48	19.28	27.51	60.19
		$\mathcal{V}_2$	V	67.83	42.86	9.64	15.74	61.5
	3DResnet	$\mathcal{V}_1$	V	63.45	42.9	52.21	47.1	66.16
		$\mathcal{V}_2$	V	61.58	41.47	56.63	47.88	64.5
	ImageBind	$\mathcal{V}_3$	V	62.78	38.46	32.13	35.01	57.03
			A	43.86	32.26	72.69	44.69	52.23
			V, A	63.28	40.76	38.96	39.84	53.87
			V, A, T	68.79	42.76	41.96	42.36	61.1
			V, A, D, T	69.4	50.75	48.96	49.84	70.41
$\mathcal{R}$	Omnivore	$\mathcal{V}_1$	V	59.76	45.31	58.09	50.91	63.03
		$\mathcal{V}_2$	V	62.3	46.55	33.61	39.04	62.27
	Slowfast	$\mathcal{V}_1$	V	60.06	40.82	24.9	30.93	56.89
		$\mathcal{V}_2$	V	57.82	41.67	43.57	42.6	59.83
	X3D	$\mathcal{V}_1$	V	54.69	39.6	49.79	44.12	54.66
		$\mathcal{V}_2$	V	54.25	40.78	60.58	48.75	56.58
	3DResnet	$\mathcal{V}_1$	V	41.88	37.65	94.19	53.79	62.42
		$\mathcal{V}_2$	V	57.82	43.56	58.92	50.09	59.22
	ImageBind	$\mathcal{V}_3$	V	37.56	36.14	96.27	52.55	54.1
			A	39.05	35.67	89.72	50.54	54.8
			V, A	54.25	40.98	62.24	49.42	55.25
			V, A, T	64.08	44.15	58.01	50.13	58.35
			V, A, D, T	63.87	49.57	64.4	56.02	65.25

Table 2: **SupervisedER** evaluation results of baselines. Variant type ($\mathcal{V}_{\#}$), Modality of features (M), Video (V), Audio (A), Depth (D), and Text (T).

(MLP) head. This approach assesses the efficacy of visual cues identified by pre-trained video recognition models in recognizing errors in sub-segments. In $\mathcal{V}_2$, we shifted our focus from sub-segment prediction and trained a transformer that processed all video sub-segments corresponding to each step. This method is designed to capitalize on the long-term temporal cues present within the video segments of recipe steps to enhance the prediction performance of the trained models. In variant $\mathcal{V}_3$, we harnessed the multimodal data of recordings and trained a unified transformer model. This approach employed an attention mechanism to integrate information from all modalities of data.

Insights. Our $\mathcal{V}_2$ models consistently outperformed $\mathcal{V}_1$ models. Incorporating additional data modalities like audio, text, and depth into our $\mathcal{V}_3$ models significantly improved their performance. Our omnivore-based $\mathcal{V}_2$ models performed similarly to our $\mathcal{V}_3$ models trained using Imagebind[15] features.

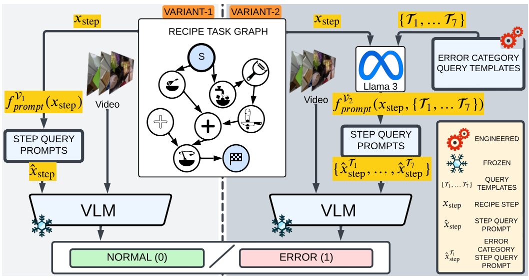

VARIANT-1
VARIANT-2
x_{step}
RECIPE TASK GRAPH
S
$f^{\mathcal{V}_1}_{prompt}(x_{\text{step}})$
Video
STEP QUERY PROMPTS
$\hat{x}_{\text{step}}$
VLM
NORMAL (0)
ERROR (1)
x_{step}
$\{\mathcal{T}_1, \ldots \mathcal{T}_7\}$
Llama 3
ERROR CATEGORY QUERY TEMPLATES
Video
$f^{\mathcal{V}_2}_{prompt}(x_{\text{step}}, \{\mathcal{T}_1, \ldots \mathcal{T}_7\})$
STEP QUERY PROMPTS
$\{\hat{x}^{\mathcal{T}_1}_{\text{step}}, \ldots, \hat{x}^{\mathcal{T}_7}_{\text{step}}\}$
VLM
ENGINEERED
FROZEN
$\{\mathcal{T}_1, \ldots \mathcal{T}_7\}$
QUERY TEMPLATES
x_{step}
RECIPE STEP
$\hat{x}_{\text{step}}$
STEP QUERY PROMPT
$\hat{x}^{\mathcal{T}_i}_{\text{step}}$
ERROR CATEGORY STEP QUERY PROMPT

Figure 6: **ZeroShotER** evaluation pipeline of VLMs

VLM	Variant	Acc	P	R	F1
Video-LLaVa [27]	$\mathcal{V}_1$	64.3	34.2	3.9	6.7
	$\mathcal{V}_2$	52.85	36.3	49.3	41.8
TimeChat [31]	$\mathcal{V}_1$	65.0	51.11	1.15	2.26
	$\mathcal{V}_2$	43.5	34.38	69.7	46.1

Table 3: **ZeroShotER** evaluation results.

Zero-Shot Error Recognition (ZeroShotER). We proposed two baseline variants, $\mathcal{V}_1$ and $\mathcal{V}_2$, for ZeroShotER[16] using prompt-and-predict architectures, as illustrated in Fig. 6. Both variants involve constructing query prompts for each recipe step and querying VLMs using these prompts along with video inputs. We utilized state-of-the-art[17] open-source VLMs, Video-LLaVa [27] and TimeChat [31], to recognize errors and reported the evaluation results using standard metrics, such as accuracy, precision, recall, and F1 scores. Specifically, for $\mathcal{V}_1$, we leveraged the associated task graphs of each recipe and generated a prefix question prompt for each step. These prompts were then used to query VLMs to recognize errors in videos. We employed a prompt-ensembling approach in $\mathcal{V}_2$ to recognize errors in a shift from the single-prompt strategy. Specifically, we designed prompt templates tailored

[15] We chose Imagebind to represent visual, textual, and auditory data in a unified embedding space.

[16] We also adapted anomaly detection methods for zero-shot error recognition (refer to Appendix B.)

[17] State-of-the-art at the time of writing this paper

to each error category (refer to Appendix B for examples). Using Llama3 [39], we generated a set of query prompts tailored to specific recipe steps and error categories. We combined the VLMs' predictions for error category-specific queries using an OR operation to determine the prediction for each step. This strategy of using error category-specific queries focused the VLMs' attention on specific segments of the video and enhanced the overall performance as illustrated in Table. 3.

Insights. While VLMs are adept at interpreting short video segments and answering straightforward questions (Eg. *Is there cucumber in the video?, Is the person peeling the cucumber?*), they struggle with tasks that require assimilation of information from various time intervals in long videos. This limitation becomes evident in tasks like error recognition in ego-centric videos, where questions such as *"Did the person chop or grate a completely peeled cucumber?"* require a deeper understanding.

Remark. The low scores indicate the complexity of the above tasks and call for developing sophisticated methods that semantically understand the context, meaning, and cause of various errors.

4.2 Multi Step Localization (MSL)

In supervised MSL, we aim to simultaneously identify the start and end boundaries of steps within an untrimmed long video and classify these identified video segments. This section addresses the question (Q3) by splitting it into two parts. For the first part—*How do state-of-the-art supervised MSL methods fare on our dataset?*; We utilized the features extracted from pre-trained video recognition models and trained our baseline models which employed ActionFormer [42] head on three proposed splits (indicated using the symbols $\{\mathcal{E}, \mathcal{P}, \mathcal{R}\}$ and detailed in App.A) of the original dataset. For the second part - *How robust are MSL methods to technique errors?*; We trained baseline models on three proposed splits ($\{\mathcal{E}, \mathcal{P}, \mathcal{R}\}$) of a modified dataset where the training subset exclusively contained normal recordings without errors. We then evaluated these models on two distinct test sets that exclusively comprised either normal recordings or error recordings (indicated using the symbols $\mathcal{T}_n$ and $\mathcal{T}_e$ respectively). We reported results using the standard MSL/TAL metrics such as Recall@K (R@K) and Mean Average Precision (mAP) across different temporal Intersections over Unions ($\mathcal{I}_t$). We presented our evaluation for both the tasks, namely, Multi-Step Localization (MSL) and Robust Multi-Step Localization (RobustMSL), in Tables 4 and 5 respectively. We also showcased qualitative examples of step localization for sampled normal and error recordings from 4 recipes in Figure 7.

Table 4: **MSL** evaluation results.

$\mathcal{B}$	$\mathcal{D}$	$\mathcal{I}_t = 0.1$			$\mathcal{I}_t = 0.3$			$\mathcal{I}_t = 0.5$		
		mAP	**R@1**	**R@5**	**mAP**	**R@1**	**R@5**	**mAP**	**R@1**	**R@5**
3D Resnet	$\mathcal{E}$	25.98	54.82	77.59	23.75	48.38	72.19	19.59	38.44	61.87
	$\mathcal{P}$	29.29	63.07	88.96	27.71	56.60	84.75	23.21	46.79	76.86
	$\mathcal{R}$	29.39	61.14	85.41	27.89	55.82	82.17	23.97	46.54	73.29
Slowfast	$\mathcal{E}$	27.68	55.73	77.45	25.51	48.98	70.90	21.09	37.82	60.58
	$\mathcal{P}$	32.77	63.22	90.43	31.21	58.82	86.82	27.25	50.70	79.49
	$\mathcal{R}$	32.90	63.97	89.29	31.47	59.26	85.32	27.89	51.62	77.27
VideoMAE	$\mathcal{E}$	28.12	51.76	73.00	26.38	46.16	67.87	21.35	37.12	57.81
	$\mathcal{P}$	38.86	64.86	84.05	37.41	60.32	80.63	32.24	51.46	71.88
	$\mathcal{R}$	37.44	63.08	80.90	35.11	57.30	77.38	30.76	49.19	69.43
Omnivore	$\mathcal{E}$	40.40	67.51	87.69	38.32	62.31	82.82	33.41	53.01	72.85
	$\mathcal{P}$	48.16	75.96	93.41	45.82	70.34	90.51	41.16	62.00	84.73
	$\mathcal{R}$	44.81	73.71	93.34	42.76	68.14	89.82	37.19	56.93	81.86

Table 5: **RobustMSL** evaluation results.

$\mathcal{B}$	$\mathcal{D}$	$\mathcal{T}$	$\mathcal{I}_t = 0.1$			$\mathcal{I}_t = 0.3$			$\mathcal{I}_t = 0.5$		
			mAP	**R@1**	**R@5**	**mAP**	**R@1**	**R@5**	**mAP**	**R@1**	**R@5**
VideoMAE	$\mathcal{E}$	$\mathcal{T}_n$	24.44	38.22	52.48	22.97	34.77	49.51	18.67	28.57	42.68
		$\mathcal{T}_e$	7.53	13.54	20.52	6.93	11.4	18.36	5.63	8.55	15.13
	$\mathcal{P}$	$\mathcal{T}_n$	26.78	37.43	46.28	25.68	34.79	44.6	22.02	29.43	39.81
		$\mathcal{T}_e$	16.98	27.43	37.76	16.46	25.53	36.03	14.64	22.03	32.07
	$\mathcal{R}$	$\mathcal{T}_n$	26.27	37.15	46.93	24.71	34.06	45.03	21.51	29.36	40.44
		$\mathcal{T}_e$	15.43	25.94	33.97	14.44	23.23	32.35	12.96	19.83	28.99
Omnivore	$\mathcal{E}$	$\mathcal{T}_n$	34.65	47.91	60.63	33.06	44.77	58.36	28.59	38.38	51.9
		$\mathcal{T}_e$	12.51	19.6	27.06	11.66	17.54	24.45	9.94	14.63	20.96
	$\mathcal{P}$	$\mathcal{T}_n$	32.5	44.45	52.47	31.13	41.53	50.91	28.39	37.03	47.97
		$\mathcal{T}_e$	21.28	31.51	40.93	20.12	28.81	39.6	18.08	24.96	36.77
	$\mathcal{R}$	$\mathcal{T}_n$	30.22	42.43	52.11	28.94	39.47	50.49	25.15	32.65	46.51
		$\mathcal{T}_e$	19.54	31.28	41.24	18.4	28.66	39.33	16.27	24.28	35.35

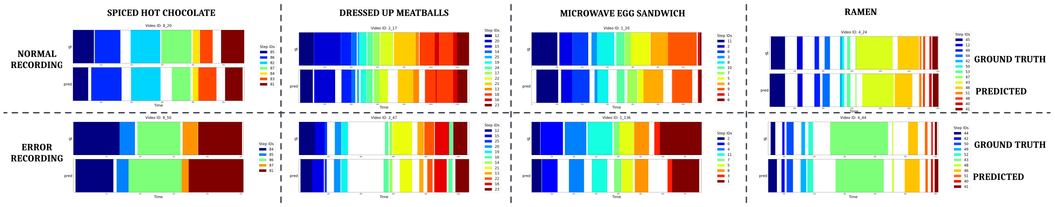


Figure 7: **MSL** qualitative results of the Omnivore-based model trained using the split $\mathcal{R}$.

Insights. We observed that our models based on Omnivore features consistently outperformed others. We also noticed that models trained using features extracted from longer video sub-segments (>1sec) outperformed those trained using features extracted from 1-sec video sub-segments (App.B). However, all our trained models using the modified dataset exhibited poor performance on $\mathcal{T}_e$ compared to $\mathcal{T}_n$.

Remark. We conjecture that exploiting semantic information from task graphs and employing probabilistic filtering methods such as particle filters to refine predictions could enhance performance.

4.3 Procedure Learning

Given long, untrimmed videos of procedural activities where the sequences of steps can be performed in multiple orders, self-supervised procedure learning methods aim to identify relevant frames across videos of an activity and estimate the sequential steps required to complete the activity. In this section, we address the question **Q4** by simultaneously answering *(a) Can we infer the underlying procedure (recipe text) from the videos of a particular recipe and (b) How does the self-supervised procedure learning methods in literature perform on the proposed dataset?*. Specifically, we answered both parts by comparing the performance of our models trained on the proposed dataset using self-supervised procedure learning methods [1, 8] against the random setting defined by EgoProceL [1](see Tab. 6). We followed the setup described in EgoProceL and trained two embedder networks, one using the Cycleback Regression loss ($\mathcal{C}$) [$\mathcal{M}_1$] [8] and the other using a blend of two loss functions: Cycleback Regression loss ($\mathcal{C}$) and Contrastive - Inverse Difference Moment loss ($\mathscr{C}$) [$\mathcal{M}_2$][1]. The combined loss function is $\mathcal{C} + \lambda \times \mathscr{C}$, where λ is a hyperparameter. While we exclusively used these loss functions to train the embedder networks, we continued using the Pro-Cut Module to categorize frames into key steps. We presented evaluation results for 5 recipes Tab. 6 and all recipes in App. B.

Table 6: **Procedure Learning.** Here, $\mathcal{P}$ represents precision, R represents recall, and I represents IOU.

Recipe	Random			$\mathcal{M}_1$ [8]			$\mathcal{M}_2$[1]		
	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{I}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{I}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{I}$
BlenderBananaPancakes	7.40	3.83	2.26	12.65	9.50	5.16	15.54	9.96	5.72
Coffee	6.54	3.87	2.17	13.68	9.91	5.49	15.76	10.25	5.63
MugCake	5.45	4.00	2.12	16.12	12.95	6.87	10.32	8.85	4.40
PanFriedTofu	5.35	3.97	1.54	8.86	10.39	3.75	9.34	12.44	3.87
Pinwheels	6.54	4.28	2.13	13.58	11.96	5.92	16.08	13.06	7.05
Average of 24 recipes	**7.61**	**3.92**	**2.22**	**15.62**	**10.85**	**5.78**	**15.78**	**10.68**	**5.82**

Insights. Our models significantly outperformed the predefined random setting, demonstrating the feasibility of inferring procedural steps from our dataset. However, these models scored lower on our dataset compared to existing procedure learning datasets. We believe this drop in performance is mainly due to our dataset's unique challenge, which includes videos with longer key step durations.

Additional Results. We provide several analyses in Appendix B, including (a) Error Category Recognition, (b) Early Error Recognition, (c) Anomaly Detection and (d) Ablation studies for MSL.

5 Discussion, Limitations and Future Work

Discussion. We introduced a novel egocentric dataset for understanding errors in procedural activities. Our dataset consists of synchronized egocentric views, audio, and depth information specifically designed for tasks such as 3D activity analysis, Procedure Learning, Error Recognition, and more. While current methods have yielded promising outcomes, they continue to struggle to tackle these challenges adequately with satisfactory results, as demonstrated by our experimental assessment. This indicates the need for further exploration in this domain. **Limitations.** We aimed to capture deviations during procedural activities from an egocentric perspective. Since such data cannot be sourced from crowd-sourced platforms we captured participant data while performing procedural activities. By the nature of the problem, errors that occur when performing procedural activities are combinatorial and can have a compounding effect. Thus, our work has the following limitations: (1) For each activity, the errors captured and presented in the dataset form a subset of the whole combinatorial space; (2) Capturing 4D data in real kitchen environments posed logistical and equipment training challenges. As a result, we were compelled to limit the data collection to a specific geographic area. **Future Work.** Our work opens up several avenues for future work. First, an exciting direction is the extension of the dataset to include activities from other domains. By incorporating tasks such as executing hardware-related activities (e.g., working with cars or computer parts), the dataset can encompass a wider range of activities. Second, the dataset can be used to compare and develop methods for solving various tasks such as Few-Shot Error Recognition using visual/textual prompts, Semantic Role Labelling, Long Video Understanding, Procedure Planning, Reducing Errors, etc.

Acknowledgements

This work was supported in part by the DARPA Perceptually-Enabled Task Guidance (PTG) Program under contract number HR00112220005, by the DARPA Assured Neuro Symbolic Learning and Reasoning (ANSR) Program under contract number HR001122S0039, by the National Science Foundation grant IIS-1652835 and by the AFOSR award FA9550-23-1-0239. We would like to express our gratitude to all the reviewers for their insightful comments and queries, which have substantially enhanced the quality of our manuscript. We would also like to extremely thank Sai Krishna, Zhang Yang, Jihan Wang, Enoch Chiu, Pratyush Kaware, Saurabh Bhole, Ankit Upadhyay, Vashist Gupta, Samuel, Jyothise, Shruthi and Manogna for allowing us to collect data in their kitchens.

References

[1] Bansal, Siddhant, Arora, Chetan, and Jawahar, C. V. My View is the Best View: Procedure Learning from Egocentric Videos. *European Conference on Computer Vision*, july 2022.

[2] Mathilde P. Chevignard, Cathy Catroppa, Jane Galvin, and Vicki Anderson. Development and evaluation of an ecological task to assess executive functioning post childhood tbi: The children's cooking task. *Brain Impairment*, 11(2):125–143, 2010.

[3] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, William Price, Will Price, Will Price, and Michael Wray. The EPIC-KITCHENS Dataset: Collection, Challenges and Baselines. *arXiv: Computer Vision and Pattern Recognition*, April 2020. ARXIV_ID: 2005.00343 MAG ID: 3022491006 S2ID: 1badccbe4a3cbf8662b924a97bbeea14fe2f1ac7.

[4] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling Egocentric Vision: Collection, Pipeline and Challenges for EPIC-KITCHENS-100. *International Journal of Computer Vision*, October 2021.

[5] Fernando De la Torre, Jessica K. Hodgins, Adam W. Bargteil, Xavier Martin, J. Robert Macey, Alex Tusell Collado, and Pep Beltran. Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. In *Tech. report CMU-RI-TR-08-22, Robotics Institute, Carnegie Mellon University*, April 2008.

[6] Dibene, Juan C. and Dunn, Enrique. HoloLens 2 Sensor Streaming. *Cornell University - arXiv*, November 2022. ARXIV_ID: 2211.02648 MAG ID: 4308505718 S2ID: b19229b4f8667dae5017cae4df5c37086332da17.

[7] Bruce Draper. DARPA's Perceptually-enabled Task Guidance (PTG) program, 2021.

[8] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal cycle-consistency learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[9] Bernard Ghanem Fabian Caba Heilbron, Victor Escorcia and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015.

[10] 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. *Computer Vision and Pattern Recognition*, 2022.

[11] Alireza Fathi, Xiaofeng Ren, and James M. Rehg. Learning to recognize objects in egocentric activities. In *CVPR 2011*, pages 3281–3288. IEEE, June 2011.

[12] Christoph Feichtenhofer. X3D: Expanding Architectures for Efficient Video Recognition, April 2020.

[13] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast Networks for Video Recognition, October 2019.

[14] Torun G Finnanger, Stein Andersson, Mathilde Chevignard, Gøril O Johansen, Anne E Brandt, Ruth E Hypher, Kari Risnes, Torstein B Rø, and Jan Stubberud. Assessment of executive function in everyday life-psychometric properties of the norwegian adaptation of the children's cooking task. *Frontiers in human neuroscience*, 15:761755, 2021.

[15] 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, 2024.

[16] Yael Fogel, Sara Rosenblum, Renana Hirsh, Mathilde Chevignard, and Naomi Josman. Daily performance of adolescents with executive function deficits: An empirical study using a complex-cooking task. *Occupational therapy international*, 2020:3051809, 2020.

[17] Reza Ghoddoosian, Isht Dwivedi, Nakul Agarwal, Chiho Choi, and Behzad Dariush. Weakly-supervised online action segmentation in multi-view instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13780–13790, 2023.

[18] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *CVPR*, 2023.

[19] Rohit Girdhar, Mannat Singh, Nikhila Ravi, Laurens van der Maaten, Armand Joulin, and Ishan Misra. Omnivore: A Single Model for Many Visual Modalities, March 2022.

[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 Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, 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 Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard 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. *arXiv*, October 2021.

[21] Chunhui Gu, Chen Sun, Sudheendra Vijayanarasimhan, Caroline Pantofaru, David A. Ross, George Toderici, Yeqing Li, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, and Jitendra Malik. Ava: A video dataset of spatio-temporally localized atomic visual actions. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2017.

[22] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning Spatio-Temporal Features with 3D Residual Networks for Action Recognition, August 2017.

[23] Youngkyoon Jang, Brian Sullivan, Casimir Ludwig, Iain Gilchrist, Dima Damen, and Walterio Mayol-Cuevas. Epic-tent: An egocentric video dataset for camping tent assembly. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[24] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`, 2014.

[25] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 780–787, 2014.

[26] Yin Li, Miao Liu, and James M. Rehg. In the eye of the beholder: Gaze and actions in first person video. *CoRR*, abs/2006.00626, 2020.

[27] Bin Lin, Bin Zhu, Yang Ye, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[28] Yicheng Qian, Weixin Luo, Dongze Lian, Xu Tang, Peilin Zhao, and Shenghua Gao. SVIP: sequence verification for procedures in videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 19858–19870. IEEE, 2022.

[29] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[30] Francesco Ragusa, Antonino Furnari, Salvatore Livatino, Salvatore Livatino, and Giovanni Maria Farinella. The MECCANO Dataset: Understanding Human-Object Interactions from Egocentric Videos in an Industrial-like Domain. *arXiv: Computer Vision and Pattern Recognition*, 2020.

[31] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. Timechat: A time-sensitive multimodal large language model for long video understanding. *ArXiv*, abs/2312.02051, 2023.

[32] Rishi Hazra. EgoTV: Egocentric Task Verification from Natural Language Task Descriptions. *arXiv.org*, 2023. ARXIV_ID: 2303.16975 S2ID: 1901745a3a592f5026abd1e9d8435019a2a25585.

[33] Marcus Rohrbach, Anna Rohrbach, Michaela Regneri, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. Recognizing fine-grained and composite activities using hand-centric features and script data. *International Journal of Computer Vision*, pages 1–28, 2015.

[34] Tim J. Schoonbeek, Tim Houben, Hans Onvlee, Peter H. N. de With, and Fons van der Sommen. IndustReal: A dataset for procedure step recognition handling execution errors in egocentric videos in an industrial-like setting, 2024.

[35] Gunnar A. Sigurdsson, Gül Varol, X. Wang, Ali Farhadi, Ivan Laptev, and Abhinav Kumar Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, 2016.

[36] Sebastian Stein and Stephen J. McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *UbiComp '13: Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738. Association for Computing Machinery, New York, NY, USA, September 2013.

[37] Yansong Tang, Dajun Ding, Dajun Ding, 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. *Computer Vision and Pattern Recognition*, June 2019.

[38] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Neural Information Processing Systems*, 2022.

[39] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[40] Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, Neel Joshi, and Marc Pollefeys. 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 (ICCV)*, pages 20270–20281, October 2023.

[41] Yue Zhao, Ishan Misra, Philipp Krahenbuhl, and Rohit Girdhar. Learning Video Representations from Large Language Models. *arXiv.org*, 2022. ARXIV_ID: 2212.04501 S2ID: 933b37b21e9d61139660088adb032ff3fdf56d86.

[42] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part IV*, volume 13664 of *Lecture Notes in Computer Science*, pages 492–510. Springer, 2022.

[43] Luowei Zhou, Chenliang Xu, and Jason J. Corso. Towards Automatic Learning of Procedures from Web Instructional Videos. *arXiv*, March 2017.

[44] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David F. Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. *arXiv: Computer Vision and Pattern Recognition*, March 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] They are part of this GitHub repo: CaptainCook4D
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In Appendix B
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] In Appendix B
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] In Appendix B
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] They can be accessed from this GitHub repo: CaptainCook4D
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] In Section 3.1
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] We discussed this in Section 3.1. Our data does not contain personally identifiable information.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] In Appendix C
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [Yes] In Section 3.1 and Appendix C
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] In Appendix C