PREGO: online mistake detection in PRocedural EGOcentric videos

Alessandro Flaborea*♦♣ Guido Maria D'Amely di Melendugno*♦ Leonardo Plini♦ Luca Scofano♦
Edoardo De Matteis♦ Antonino Furnari♠ Giovanni Maria Farinella‡♠ Fabio Galasso‡♦
{flaborea,damely,dematteis,galasso}@di.uniroma1.it {plini,scofano}@diag.uniroma1.it
{antonino.furnari,giovanni.farinella}@unict.it

♦Sapienza University of Rome, Italy ♣ItalAI (italailabs.com) ♠University of Catania, Italy

Abstract

Promptly identifying procedural errors from egocentric videos in an online setting is highly challenging and valuable for detecting mistakes as soon as they happen. This capability has a wide range of applications across various fields, such as manufacturing and healthcare. The nature of procedural mistakes is open-set since novel types of failures might occur, which calls for one-class classifiers trained on correctly executed procedures. However, no technique can currently detect open-set procedural mistakes online. We propose PREGO, the first online one-class classification model for mistake detection in PRocedural EGOcentric videos. PREGO is based on an online action recognition component to model the current action, and a symbolic reasoning module to predict the next actions. Mistake detection is performed by comparing the recognized current action with the expected future one. We evaluate PREGO on two procedural egocentric video datasets, Assembly101 and Epic-tent, which we adapt for online benchmarking of procedural mistake detection to establish suitable benchmarks, thus defining the Assembly101-O and Epic-tent-O datasets, respectively. The code is available at https://github.com/aleflabo/PREGO.

1. Introduction

Egocentric procedure learning is gaining attention due to advancements in Robotics and Augmented Reality (AR) technologies. These technologies are pivotal to enhancing online[1] monitoring systems, offering real-time feedback, and improving operator efficiency in various fields. Recent works have produced numerous datasets [4, 8, 9, 12, 14, 18, 26, 27, 27, 28, 32, 35, 40], methodologies aimed at advancing procedure learning [12, 14, 17, 32, 34, 39] and error detection models [8, 27, 35]. Despite these advancements, as outlined in Table 1, state-of-the-art methods typically focus on supervised and offline mistake detection. They are unsuitable for situations requiring dynamic decision-making, specifically within an *online* setting, or when errors occur unpredictably, thus defining these instances as open-set conditions.

In this work, we propose the first model to detect PRocedural errors in EGOcentric videos (PREGO), which operates online, thus causal, and can recognize unseen procedural mistakes, fitting for open-set scenarios. We prioritize egocentric videos due to their highly detailed perspective, essential for accurately identifying steps within procedures.

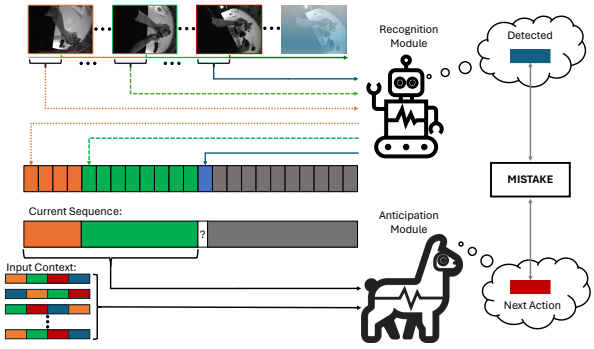

Figure 1. PREGO is based on two main components: The recognition module (top) processes the input video in an online fashion and predicts actions observed at each timestep; the anticipation module (bottom) reasons symbolically via a Large Language Model to predict the future action based on past action history and a brief context, such as instances of other action sequences. Mistakes are identified when the current action detected by the step recognition method differs from the one forecasted by the step anticipation module (right).

*Authors contributed equally.

‡Co-senior role.

[1]Most workflows can be aided by *online* monitoring algorithms, which provide feedback to the operator in due course. However, they may lag due to processing or connectivity delays. We distinguish online from real-time, whereby the second has strict requirements of instantaneous response.

Additionally, the widespread use of egocentric cameras in industries [22] necessitates the development of online error detection techniques to improve the safety and efficiency of workers. The online attribute is achieved by analyzing input videos sequentially up to a given frame t, ensuring that no future actions influence the current step recognition. On the other hand, open-set learning is performed by exclusively exposing PREGO to correct procedural sequences when predicting mistakes, following the One-Class Classification (OCC) paradigm [11, 38]. Any step within a procedure that significantly diverges from the expected correct patterns is identified as an error, allowing PREGO to recognize a wide range of procedural mistakes without being confined to a restricted set of predefined ones.

PREGO's architecture is dual-branched, as depicted in Fig. 1. The first branch, the *step-recognition branch*, analyzes frames in a procedural video up to a current time t, aiming to classify the action being undertaken by the operator. This branch can exploit the current state-of-the-art video-based online step recognition model, [2, 34]. Concurrently, the second branch is in charge of *step-anticipation*, tasked to predict the action at time t, based solely on the steps up to $t-1$. We propose using a pre-trained Large Language Model (LLM) [33] for zero-shot symbolic reasoning through contextual analysis [13, 19, 31]. An error is detected upon a misalignment between the currently recognized action and the anticipated one, thereby signaling a deviation from the expected procedure. Utilizing correctly executed procedures as instances in the query prompt obviates the necessity for additional model fine-tuning and leverages the pattern-completion abilities of LLMs. Our proposed approach is an abstraction from the video content. Using labels allows for longer-term reasoning, as a label summarizes several frames. Also, this approach is an alternative to the carefully constructed action inter-dependency graphs [3]. We demonstrate that symbolic reasoning subsumes understanding lengthy procedures and the action inter-dependencies, suggesting repositioning from semantic-based expressions of procedures to an implicit representation, where only patterns of symbols have to be recognized and predicted. By representing procedures as sequences and their steps as symbols, we let the predictor focus on the patterns that characterize the correct procedures.

To support the evaluation of PREGO, we adapt the procedural benchmarks of Assembly101 [28] and Epic-tent [14], formalizing the novel task of online procedural mistake detection. In the adapted online mistake detection benchmarks, which we dub Assembly101-O and Epic-tent-O, the model is tasked with detecting when a procedural mistake is made, thus compromising the procedure. The compromising mistake may be a wrong action or a relevant action performed in such an order that the action dependencies are not respected.

We summarize our contributions as follows:

- We present PREGO, the first method designed for online and open-set detection of procedural errors in egocentric videos. PREGO's online feature ensures causal analysis by sequentially processing input videos up to a given frame, preventing future actions from influencing current step recognition.
- PREGO achieves open-setness by exclusively relying on correct procedural sequences at training time, following the One-Class Classification (OCC) paradigm. This allows PREGO to identify a wide range of procedural mistakes, avoiding confinement to a predefined set of errors and avoiding the need for fine-grained mistake annotations.
- We propose using a pre-trained LLM for zero-shot symbolic reasoning through contextual analysis to predict the next action.
- To evaluate PREGO, we introduce the novel task of online procedural mistake detection and re-arrange existing datasets to provide two new benchmarks, referred to as Assembly101-O and Epic-tent-O.

2. Related Work

2.1. Procedural Mistake Detection

Procedural learning has seen significant advancements with the creation of diverse datasets [9, 18, 26, 32, 40] that provide insights into both structured [4, 25, 35] and unstructured [6, 14] tasks, covering a spectrum from industrial assembly [25–28] to daily cooking activities [6, 15, 30]. Despite the increased focus on this area, there is a notable lack of a unified methodology for mistake detection, resulting in fragmented literature and scarce evaluations.

Datasets. ATA [12] is a procedural dataset designed for offline mistake detection in assembling activities. It only reports video-level mistakes annotations, making it impractical for frame-based applications. Assembly101 [28] is a large-scale video dataset that annotates frame-level mistakes. The videos represent actors assembling toys, and the dataset offers synchronized Ego-Exo views and hand-positions data. Another recent assembling dataset with frame-level annotations is IndustReal [27]. However, the authors consider a single toy, which results in a single procedure to be learned. Epic-tent [14] is a dataset with a different domain, as it reports actors building up a tent in an outdoor scenario. The participants have different degrees of expertise, and they naturally commit mistakes that have been annotated in Epic-tent. Holoassist [35] is a recent dataset that presents egocentric videos of people performing several manipulating tasks instructed by an expert. In this study, we employ [14, 28] datasets since they give insights into errors happening during procedures in two different contexts, i.e., controlled industrial and outdoor environments[2].

[2] At the time of writing [27, 35] were unavailable publicly.

Table 1. Comparison among relevant models. In the modalities column, *RGB* stands for RGB images, *H* for hand poses, *E* for eye gaze, *K* for keystep labels, *D* for depth. Differently from previous works, we are the first to consider an egocentric one class and online approach to mistake detection.

	Ego	OCC	Online	Modalities	Task	Datasets
Ding et al. [8] - *Arxiv '23*				*K*	Mistake Detection	Assembly101 [28]
Wang et al. [35] - *ICCV '23*	✓			*RGB+H+E*	Mistake Detection	HoloAssist [35]
Ghoddoosian et al. [12] - *ICCV '23*				*RGB*	Unknown sequence detection	ATA [12] and CSV [24]
Schoonbeek et al. [27] - *WACV '24*	✓		✓	Multi	Procedure Step Recognition	IndustReal[27]
PREGO	✓	✓	✓	*RGB*	Mistake Detection	*Assembly101-O*, *Epic-tent-O*

Methods. In Table 1, we report the main features of the recent approaches to Mistake Detection in procedural videos. In [12], the authors train an action recognizer model and consider error detection a semantic way of evaluating the segmentation results. Their method is thus explicitly offline, while PREGO aims to promptly detect procedure mistakes as soon as they occur. By contrast, Assembly101 [28] and Holoassist [35] apply the same error detection baselines on varying granularity but also operate offline, requiring video segmentation. Ding et al. [8] use knowledge graphs for error identification, bypassing video analysis and extracting procedural steps from transcripts, presenting a distinct methodology within the procedural learning field. PREGO diverges from these works as it leverages the video frames to detect the steps of the procedure online and leverages symbolic reasoning for an online assessment of the procedure's correctness. Moreover, acknowledging that the mistake detection task shares many aspects with the established field of Video Anomaly Detection, we design PREGO to work in an OCC framework. As motivated in [11, 38], this choice ensures that PREGO is not constrained to detect only specific kinds of errors, as it is trained on sequences that do not contain mistakes.

2.2. Steps recognition and anticipation

Step recognition is the task of identifying actions within a procedure. Indeed, a procedure is an ordered sequence of steps that bring to the completion of a task. Step recognition is crucial in areas such as autonomous robotics and educational technology. Recent contributions in this domain include [29], which uses a novel loss for self-supervised learning and a clustering algorithm to identify key steps in unlabeled procedural videos. [17] introduces an action segmentation model using an attention-based structure with a Pairwise Ordering Consistency loss to learn the regular order of the steps in a procedure. They devise a weakly supervised approach, using only the set of actions occurring in the procedure as labels, avoiding frame-level annotations. [39] approaches the task by leveraging online instructional videos to learn actions and sequences without manual annotations, blending step recognition with a deep probabilistic model to cater to step order and timing variability. Notably, An et al. [2] proposed miniROAD explicitly targeting online action detection. They leverage an RNN architecture and regulate the importance of the losses during training to perform active action recognition.

On the other hand, step anticipation focuses on predicting forthcoming actions in a sequence crucial for real-time AI decision-making. [1] addresses this by generating multiple potential natural language outcomes, pretraining on a text corpus to overcome the challenge of diverse future realizations. Additionally, the framework of [23] proposes solutions to future activity anticipation in egocentric videos, using contrastive loss to highlight novel information and a dynamic reweighing mechanism to focus on informative past content, thereby enhancing video representation for accurate future activity prediction. Unlike prior works, PREGO is the first model that anticipates actions via LLM symbolic reasoning in the label space.

2.3. Large Language Modelling and Symbolic Reasoning

LLMs are trained on large datasets and have many parameters, giving them novel capabilities compared to previous language models [36]. LLMs have shown remarkable abilities in modeling many natural language-related [33] and unrelated tasks [5, 13, 36]. Their next-token prediction mechanism aligns with our action anticipation branch, where both systems aim to infer future actions based on collected data.

Recent research [10, 13, 16, 19, 21] has explored LLMs' ability to operate as *In-Context Learners* (ICLs), which means they can solve novel and unseen tasks. Given a query prompt with a context of input-output examples, LLMs can comprehend and address the problems in this setting without further fine-tuning. LLMs as ICLs have been used for a variety of tasks, including planning [21], programming [13, 16], logical solvers [10], and symbolic reasoning [19].

Some work has shown that LLMs can generate semantically significant patterns [19], while [37] has explored LLMs' in-context capabilities on semantically unrelated labels, where there is no relationship between a token and its meaning. Recent works [20, 21] studied the opportunity to employ LLMs for devising plans to accomplish tasks. In our mistake detection pipeline, we leverage ICL using an

LLM as our action anticipation branch. Given examples of similar procedures, such LLM continues sequences of steps in a procedure, represented as symbols. The LLM acts as a symbolic pattern machine, continuing the pattern of actions given a context of sequences performed goal-oriented, even if the sequences do not follow a semantic scheme. This combines the challenges of predicting future actions and of having no semantics.

3. Benchmarking online open-set procedural mistakes

This section presents the benchmark datasets and the evaluation metrics used in our experiments. First, we introduce the reviewed online variants of Assembly101 and Epic-tent (Sec. 3.1), and then we define the proposed online metrics in Sec. 3.2.

3.1. Datasets

We propose *Assembly101-O* and *Epic-tent-O* as a refactoring of the original datasets [14, 28], detailing the selected labeling for online benchmarking, and the novel arrangement of training and test splits, to account for open-set procedural mistakes.

3.1.1 Assembly101-O

Assembly101 [28] is a large-scale video dataset that enables the study of procedural video understanding. The dataset consists of 362 procedures of people performing assembly and disassembly tasks on 101 different types of toy vehicles. Each procedure is recorded from static (8) and egocentric (4) cameras and annotated with multiple levels of granularity, such as more than 100K coarse and 1M fine-grained action segments and 18M 3D hand poses. The dataset covers various challenges, including action anticipation and segmentation, mistake detection, and 3D pose-based action recognition.

Assembly101 for online and open-set mistake detection (*Proposed*) We introduce a novel split of the dataset [28] that enables online, open-set mistake detection by design. Assembly101-O mainly encompasses two edits on [28], namely, a new train/test split and a revision of the length of the procedures. The novel split encloses all the correct procedures in the train set, leaving the videos with mistakes for the test and validation set. This modification is needed to allow models to learn the sequences of steps that characterize correct procedures in a one-class classification fashion. In this way, models do not undergo the bias of learning specific kinds of mistakes during training; instead, as they are exposed exclusively to correct processes, they adhere to the OCC protocol and consider mistakes all actions that diverge from the learned normalcy. As a further advantage, this saves all mistaken annotated videos for the test set, granting better balanced correct/mistaken validation and test sets and a more comprehensive evaluation of mistake detection. The second revision involves evaluating each video for benchmarking until the procedure is compromised, meaning until a mistake occurs due to incorrect action dependencies. Indeed, coherently with the OCC protocol, models are tasked with learning the correct flows of steps that allow procedures to be efficiently completed and considering sub-process after a mistake occurs creates a gap between the actions in the train set and those in the test, which prevents the models from recognizing or correctly anticipating the procedure steps. Moreover, this work proposes to focus on egocentric videos to be consistent with real-world applications. Hence, we only leverage a single egocentric video from the four views available for each video in [28].

Epic-tent is a dataset of egocentric videos that capture the assembly of a camping tent outdoors. The dataset was collected from 24 participants who wore two head-mounted cameras (GoPro and SMI eye tracker) while performing the task. The dataset contains 5.4 hours of video recordings and provides annotations for the action labels, the task errors, the self-rated uncertainty, and the gaze position of the participants. The dataset also reflects the variability and complexity of the task, as the participants interacted with non-rigid objects (such as the tent, the guylines, the instructions, and the tent bag) and exhibited different levels of proficiency and uncertainty in completing the task.

Epic-tent for online and open-set mistake detection (*Proposed*) This section introduces a novel split for the Epic-tent dataset [14], designed to be adapted for the open-set mistake detection task. It is labeled with nine distinct mistake types. However, among these, "*slow*", "*search*", "*misuse*", "*motor*", and "*failure*" do not represent procedural errors, since, when they occur, the procedure is not tainted. On the other hand, the categories "*order*", "*omit*", "*correction*", and "*repeat*" are procedural mistakes, which we consider for our task. Epic-tent is designed for the supervised error detection task and, differently from [28], every reported procedure includes some mistakes, hampering the reproduction of the split procedure proposed for Assembly101-O. Nonetheless, this dataset provides the confidence scores assigned to each frame by the performer, indicating their self-assessed uncertainty during the task. Thus, we define a strategy for splitting , reported in Sec. C of the supplementary materials, in which videos featuring the most confident performers form the train set, while those showing higher uncertainty (and thus potentially more prone to errors) populate the test set. This partitioning strategy holds encouraging promise, especially in real-world scenarios where the accurate labeling of erroneous frames is hard to achieve or where the training of a mistake detector can initiate immediately post-recording without necessitating the completion of the entire annotation process. The resulting split comprises 14 videos

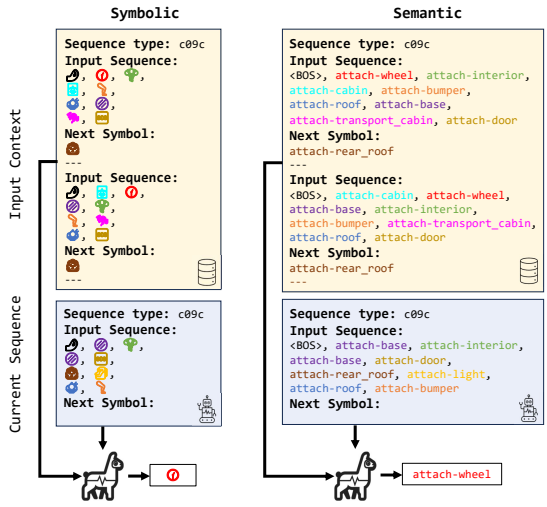

Figure 2. Two different representations of the actions in the prompt for the LLM model. On the left, the prompt is represented using symbolic labels. On the right, the prompt encompasses the names of the actions in the transcript. The context part of the prompt is fixed and retrieved from the dataset, while the recognition module extracts the current sequence.

for the training set and 15 for the test set.

The Epic-tent dataset showcases only egocentric videos recorded through Go-Pro cameras. This further highlights the practicality and relevance of the proposed novel benchmark in open-scene contexts. The videos in the test set are also trimmed up to the last frame of the first mistake occurring in the video, while those representing correct procedures are maintained unaltered.

3.2. Metrics

To assess the performance of our procedural mistake detection model, we use True Positives as a measure of the model correctly identifying errors and True Negatives as a measure of accurately labeling steps that are not errors. Thus, we rely on the Precision, Recall, and F1 score metrics to evaluate the performance of our model. These metrics offer valuable insights into the model's capability to identify and classify mistakes within procedural sequences. More specifically, precision quantifies the accuracy when predicting mistakes, minimizing false positives. Recall assesses the model's capability to retrieve all mistakes, reducing the number of false negatives. Finally, the F1 score is the harmonic mean of precision and recall, and it balances failures due to missing mistakes and reporting false alarms.

4. Methodology

PREGO exploits a dual-branch architecture that integrates procedural step recognition with anticipation modeling, as depicted in Fig. 1. In the following sections, we elaborate on the problem formalization (Sec. 4.1), present the branches for step-recognition (Sec. 4.2) and step-anticipation (Sec. 4.3), and finally we illustrate the mistake detection procedure (Sec. 4.4).

4.1. Problem Formalization

We consider a finite set of N procedures $\{p_i\}_{i=1}^{N}$ that encodes the sequence of actions as $p_i = \{a_k\}_{k=1}^{K_i}$ where K varies depending on the specific procedure i and $a_k \in \mathcal{A} = \{a | a \text{ is a possible action}\}$. Each procedure is also represented by a set of videos $\{v_i\}_{i=1}^{N}$ that are composed of frames $v_i = \{f_\tau\}_{\tau=1}^{M_i}$ where M_i is the total number of frames in the video i.

Fixed a frame f_τ from a given video v_i, PREGO's task is double-folded: it has to (1) recognize the action a_τ corresponding to the frame f_τ in the video and (2) predict the action a_τ that will take place at time τ considering only past observations until time $\tau - 1$.

The step recognition task is performed by a module ρ that takes as input the encoded frames of v_i up to τ and returns an action a_τ^ρ. We then feed the module ξ, responsible for the anticipation task, with all the $a_1^\rho, ..., a_{\tau-1}^\rho$ actions to have a prediction a_τ^ξ for the next action in the obtained sequence. Finally, we compare a_τ^ρ with a_τ^ξ and we deem as mistaken the actions where a misalignment between the outputs of the two branches occurs. For clarity, in the remainder of this section, we consider a single procedure p associated with a video v.

4.2. Step Recognition

The step recognition module, denoted as ρ, receives encoded frames from v_i up to τ as input and generates the action a_τ^ρ. This module can be designed in a modular fashion under the condition that the model operates online, meaning it lacks knowledge of future events. In our approach, we leverage MiniRoad [2], renowned for its state-of-the-art performance in online action detection, its efficiency in computational complexity (measured in GFlops), and parameter count.

Within this framework, with w representing the size of window W, the model forecasts the action a_τ by considering frames $f_{\tau-w}, .., f_\tau$. However, this approach yields redundant outcomes as the model frequently predicts the same action for consecutive frames. We adopt a simple procedure to ensure consistency: we only consider unique actions whenever the model predicts the same action for consecutive frames. The loss for this step recognition module is calculated through a Cross Entropy Loss, comparing the actual action a_τ with the predicted action a_τ^ρ.

4.3. Step Anticipation

We introduce a novel approach for step forecasting in procedural learning by harnessing the power of symbolic rea-

soning [19] via a Language Model (LM). Specifically, we employ a Large Language Model (LLM) as our ξ model for next-step prediction, feeding it with prompts from procedural video transcripts. These prompts are structured in two parts: the first part comprises contextual transcripts C, *Input Context* in Fig. 2, extracted from similar procedures to inform the LLM about typical step sequences and order. The second part, *Sequence* in the Figure, includes the current sequence of actions up to a specific frame, f_τ, detected by our module ρ, i.e.,

$$s_\tau = [a_1^\rho, ..., a_{\tau-1}^\rho] \tag{1}$$

This approach enables the LLM to utilize in-context learning, eliciting its ability to anticipate subsequent actions. Our framework operates in a zero-shot fashion, relying on the LLM's ability to retrieve the correct sequence continuation without specific training or fine-tuning but only leveraging the positive examples within the input prompts. Additionally, our method employs symbolic representations of the steps, converting the set of actions $\mathcal{A}$ into a symbolic alphabet Ω through an invertible mapping γ. Therefore, we can express the symbolic predicted sequence as:

$$\gamma(s_\tau) = [\gamma(a_1^\rho), ..., \gamma(a_{\tau-1}^\rho)] = [\omega_1, ..., \omega_{\tau-1}] \tag{2}$$

This conversion abstracts the actions from their semantic content, allowing the LLM to focus on pure symbols and sequences, thus simplifying the complexity of predicting the following action.
Finally, the ξ module, given the examples C and the current symbolic transcript $\gamma(s_\tau)$ described in its prompt, is required to output the most probable symbol ω_τ to continue the sequence (see Figure 2). At this point, we apply the inverse function of γ to retrieve the underlying step label, i.e., $a_\tau^\xi = \gamma^{-1}(\omega_\tau)$.

4.4. Mistake Detection

We finally compare the outputs of the two modules to detect procedural mistakes. Precisely, we consider as correct all the steps where the outputs of the two modules align with each other, while we deem as an error the cases for which the two outputs diverge. That is:

$$\begin{cases} a_\tau^\rho \neq a_\tau^\xi & \text{MISTAKE} \\ a_\tau^\rho = a_\tau^\xi & \text{CORRECT} \end{cases} \tag{3}$$

5. Experiments

In this section, we present the results of our experiments on online and open-set mistake detection in procedural videos. We contrast PREGO with several baselines that employ different mistake detector techniques or use the ground truth as an oracle. The oracular scenario represents an upper bound for a given anticipation method since the recognition branch does entirely rely on the ground truth. All the baselines are assessed on the Assembly101-O and Epic-tent-O datasets, detailed in section 3.1. Evaluation metrics include precision, recall, and F1 score, as outlined in 3.2. Baselines are introduced in section 5.1, and the primary results are analyzed in 5.2. Furthermore, we explore the influence of different prompt types in 5.3 and the context in 5.4. Lastly, implementation specifics are discussed in 5.5, along with addressing certain limitations.

5.1. Baselines

To estimate the effectiveness of PREGO, we evaluate its performance by comparing it against the following baseline models based on the metrics presented in Sec. 3.2:

One-step memory We define a *transition matrix* considering only the correct procedures. Specifically, given the set of the actions $\mathcal{A}$ in the training set with $|\mathcal{A}| = C$, we define a transition matrix $M \in \mathbb{R}^{C\times C}$ which stores in position (l, m) the occurences that action m follows action l. We then label as *mistake* the actions occurring in the test split that do not correspond to transitions recorded in the training set.

OadTR for mistake detection The work [34] proposes a framework for online action detection called OadTR that employs a Vision Transformer to capture the temporal structure and context of the video clips. The framework consists of an encoder-decoder architecture. The encoder takes the historical observations as input and outputs a task token representing the current action. The decoder takes the encoder output and the anticipated future clips as input and outputs a refined task token incorporating the future context. In the context of procedural error detection, a mistake is identified when the output from the encoder does not align with the one from the decoder.

BERT [7] We leverage the capability of BERT utilizing its specific [CLS] token to predict the correct or erroneous sequence of action. More specifically, we fine-tune BERT using the next-sentence-prediction task, where the model is trained to predict whether one sentence logically follows another within a given text. In our context, we apply this to determine whether step B can follow another step A within a procedure. Here, steps are defined as sets of two words, such as *attach wheel*, representing coarse actions. To perform this, BERT is presented with pairs of sentences corresponding to actions A and B, tasking it with predicting the sequential relationship between them. BERT's advantage lies in pre-training on a vast text corpus, followed by fine-tuning for our specific scenario. This process enables BERT to grasp contextual connections between sentences, rendering it effective for tasks like classifying procedures and comprehending the logical flow of information in text.

Table 2. A comparative assessment between PREGO and the chosen baseline methods is conducted to detect procedural mistakes using the Assembly101-O and Epic-tent-O datasets.

			Assembly101-O			Epic-tent-O		
	Step Recog.	**Step Antic.**	**Precision**	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
One-step memory	*Oracle*		16.3	30.7	21.3	6.6	26.6	10.6
BERT [7]	*Oracle*		78.2	20.0	31.8	75.0	5.6	10.4
PREGO	*Oracle*	*GPT-3.5*	29.2	75.8	42.1	9.9	73.3	17.4
PREGO	*Oracle*	*LLAMA*	30.7	94.0	46.3	10.7	86.7	19.1
OadTR for MD [34]	*OadTR [34]*	*OadTR [34]*	24.3	18.1	20.7	6.7	21.7	10.2
PREGO	*OadTR [34]*	*LLAMA*	22.1	**94.2**	35.8	9.5	**93.3**	**17.2**
PREGO	*MiniRoad [2]*	*GPT-3.5*	16.2	87.5	27.3	4.3	66.6	8.0
PREGO	*MiniRoad [2]*	*LLAMA*	**27.8**	84.1	**41.8**	**8.6**	20.0	12.0

5.2. Results

We evaluate PREGO's performance on two datasets, Assembly101-O and Epic-tent-O, and detail the results in Table 2. We replaced the step recognition branch's predictions with ground truth action labels to assess the upper bound on performance without step detection bias defining the *Oracle* setting. This approach simulates a scenario where the video branch perfectly recognizes actions in the videos. The One-step memory method considers only the previous action, while BERT reasons at a higher level of abstraction and leverages past actions more effectively. This reduces false alarms but introduces a conservative bias in the form of missing mistakes. PREGO outperformed all baselines by leveraging symbolic reasoning for richer context modeling. PREGO$_{LLama}$ achieved the highest F1-score with a 45.6% improvement over BERT, demonstrating the effectiveness of symbolic reasoning. Among PREGO configurations, PREGO$_{LLAMA}$ performed 9% better than PREGO$_{GPT-3.5}$ on Assembly101-O, due to its more powerful symbolic representation. Similar trends are observed on Epic-tent-O with metric values influenced by dataset characteristics (Epic-tent-O allows for more diverse assembly procedures compared to Assembly101-O).

We move beyond oracle methods that rely on ground truth information and compare PREGO's performance against the established method OadTR [34] per-frame action detection and forecasting. PREGO$_{Llama}$, using the same method for step recognition, significantly outperforms OadTR for MD achieving a 102% improvement in F1-score (refer to Table 2 for detailed results). OadTR is restricted to processing fixed-size video segments with a default window of 64 frames, resulting in the smallest F1-score. Indeed, it is insufficient for capturing the context of long procedures lasting an average of 7 minutes in Assembly101. The improvement can also be attributed to PREGO's symbolic step anticipation branch. Symbolic reasoning allows PREGO to operate at a higher level of abstraction than video-based methods like OadTR. This advantage mitigates video-based approaches' challenges with occlusion and forecasting fine-grained actions.

PREGO$_{LLama}$ can better learn the normal patterns of the procedures and detect deviations from them, achieving the best results in terms of F1-score. In addition, PREGO$_{GPT-3.5}$ incurs costs that scale with the number of processed tokens, hindering its suitability for large-scale studies. LLAMA, being open-source, facilitates cost-effective exploration of PREGO at scale. Compared to their oracle counterparts, PREGO$_{GPT-3.5}$ and PREGO$_{LLama}$ could potentially gain 54% and 11% improvement in F1-score, respectively. This suggests that the video branch's accuracy bottlenecks overall performance. However, the oracle recognition experiment also highlights the potential for improvement within PREGO itself. Other factors influencing performance include the quality of symbolic inputs, semantic prompts, and the underlying LLM architecture.

5.3. Performance of Different Prompt Types

We investigate the effect of different action representations in the prompt for the Step Anticipation task. Following [19], we consider three ways of representing an action: numerical, semantic, or random symbols. Numerical representation means that an action label is replaced with an index in the range $[0, \mathcal{A}]$, where $\mathcal{A}$ is the total number of actions. Semantic representation implies that the action is represented by its action label. Random symbol indicates that each action is assigned to a different symbol, such as a set of emojis. This allows us to examine how the LLM can manage different levels of abstraction and expressiveness of the input prompt. Fig. 2 illustrates an example of the same prompt in two representations, symbolic and semantic.

Table 3 shows the experiment results using the described representations. We observe that all the different representations achieve close performance, with the random representation achieving the highest F1 score, 41.8, followed by the semantic and numerical representations, with 41.4 and 39.9, respectively. We hypothesize that employing a numerical system to represent different actions might inadvertently introduce a form of bias related to ordering. This type of bias occurs because the relationship between specific actions and their corresponding numerical values is inherently

Table 3. Performance of PREGO with different prompt representations for Procedural Mistake Detection evaluated via F1 score, precision and recall on the Assembly101-O dataset.

	Precision	Recall	F1 score
Numerical	26.7	78.6	39.9
Semantic	**27.8**	81.3	41.4
Random	**27.8**	**84.1**	**41.8**

Table 4. Impact of prompt variations on PREGO - Unreferenced-Context, Elaborate, and Referenced-Context prompts. Evaluated via F1 score, precision and recall on the Assembly101-O dataset.

	Precision	Recall	F1 score
Elaborate	26.9	82.4	40.5
Unreferenced-Context	27.3	85.2	41.4
Referenced-Context (*PREGO*)	**27.8**	**84.1**	**41.8**

arbitrary, lacking a natural or logical sequence. As a result, the numerical mapping can obscure the characteristics of the actions being represented, leading to potential challenges in accurately anticipating or predicting future actions based on these numerical representations. Remarkably, the semantic representation achieves a comparable performance even though words can introduce bias or ambiguity into the model. This indicates that PREGO can handle the natural language input and extract the relevant information for the step anticipation task. Surprisingly, the random symbol representation has the highest performance amongst the other representations, even though the model does not have any semantic or numerical association with them. This suggests that the model effectively learns the temporal structure of the actions from the input history, regardless of the symbol representation.

5.4. Performance of Different Prompt Context

We examine two alternative ways of writing a prompt (Table 4) for the PREGO method: prompting with a less representative context Vs. a more elaborate one. The less informative prompt, labeled as "Unreferenced-Context" in Table, requests PREGO to produce the next step without providing the model with the information that the contexts are sequences and that the output required is a symbol. The context is simply given as "Context", the current sequence is given as "Input", and the next step is requested as "Output". The more elaborate prompt, labeled "Elaborate" in Table, has a more complex prompt for both the context and the output. The context is given with the sentence "Given the sequences of the following type:", the sequence to be completed as "Complete the following sequence", and the output "Sequence is completed with". The three prompts are shown in Fig. 1 of the supplementary materials.

The results show that the referenced-context prompt achieves the best F1 score (41.8). The other two alternatives perform similarly, reaching an F1 score of 41.4 and 40.5. The detailed prompt structure is the most effective way of writing a prompt for the PREGO method, as it clearly conveys the essential information and the objective of the task.

5.5. Implementation Details

PREGO is trained on two P6000 GPUs using the Adam optimizer, a batch size of 128, a learning rate of $1e^{-5}$, and a weight decay of $1e^{-4}$. For Assembly-101-O, we use the pre-extracted TSN frame level features from [28]. For Epic-Tent-O, we extract the features using the same method. The training process takes approximately 4 hours. PREGO achieves 0.02 fps on an NVIDIA Quadro P6000, meeting our needs without real-time constraints.

Limitations Across the currently available procedural datasets with annotated mistakes, the number of procedures only ranges up to hundreds, which is a limitation for current deep learning techniques. The original Assembly101 [28] dataset encompasses 330 procedures; our proposed Assembly101-O inherits only the procedures without mistakes as the learning set, namely 190 procedures; similarly, both Epic-tent [14] and Epic-tent-O only include 29 videos depicting the same task. We acknowledge the need for a large-scale dataset for online mistake detection and leave it as a future work. Indeed, more procedures will likely let the models generalize better, improving their capability to deal with multiple plausible procedures.

6. Conclusion

We have introduced PREGO, a one-class, online approach for mistake detection in procedural egocentric video. PREGO predicts mistakes by comparing the current action predicted by an online step recognition model with the next action, anticipated through symbolic reasoning performed via LLMs. To evaluate PREGO, we adapt two datasets of procedural egocentric videos for the proposed task, thus defining the Assembly101-O and Epic-tent-O datasets. Comparisons against different baselines show the feasibility of the proposed approach to one-class online mistake detection. We hope that our investigation and the proposed benchmark and model will support future research in this field.

Acknowledgements

This work was carried out while Leonardo Plini was enrolled in the Italian National Doctorate on Artificial Intelligence run by Sapienza University of Rome. We thank DsTech S.r.l. and the PNRR MUR project PE0000013 Future Artificial Intelligence Research (FAIR) (CUP: B53C22003980006 and CUP: E63C22001940006) for partially funding the Sapienza University of Rome and University of Catania.

References

[1] Mohamed A. Abdelsalam, Samrudhdhi B. Rangrej, Isma Hadji, Nikita Dvornik, Konstantinos G. Derpanis, and Afsaneh Fazly. Gepsan: Generative procedure step anticipation in cooking videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2988–2997, 2023. 3

[2] Joungbin An, Hyolim Kang, Su Ho Han, Ming-Hsuan Yang, and Seon Joo Kim. Miniroad: Minimal rnn framework for online action detection. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10307–10316, 2023. 2, 3, 5, 7

[3] Kumar Ashutosh, Santhosh Kumar Ramakrishnan, Triantafyllos Afouras, and Kristen Grauman. Video-mined task graphs for keystep recognition in instructional videos. *arXiv preprint arXiv:2307.08763*, 2023. 2

[4] Yizhak Ben-Shabat, Xin Yu, Fatemeh Saleh, Dylan Campbell, Cristian Rodriguez-Opazo, Hongdong Li, and Stephen Gould. The ikea asm dataset: Understanding people assembling furniture through actions, objects and pose. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 847–859, 2021. 1, 2

[5] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023. 3

[6] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019. 6, 7

[8] Guodong Ding, Fadime Sener, Shugao Ma, and Angela Yao. Every mistake counts in assembly. *arXiv preprint arXiv:2307.16453*, 2023. 1, 3

[9] Ehsan Elhamifar and Zwe Naing. Unsupervised procedure learning via joint dynamic summarization. 2019. 1, 2

[10] Jiazhan Feng, Ruochen Xu, Junheng Hao, Hiteshi Sharma, Yelong Shen, Dongyan Zhao, and Weizhu Chen. Language models can be logical solvers. *ArXiv*, abs/2311.06158, 2023. 3

[11] Alessandro Flaborea, Luca Collorone, Guido Maria D'Amely di Melendugno, Stefano D'Arrigo, Bardh Prenkaj, and Fabio Galasso. Multimodal motion conditioned diffusion model for skeleton-based video anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10318–10329, 2023. 2, 3

[12] 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 (ICCV)*, pages 10128–10138, 2023. 1, 2, 3

[13] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962, 2023. 2, 3

[14] 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 *Int. Conf. Comput. Vis.*, 2019. 1, 2, 4, 8

[15] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of Computer Vision and Pattern Recognition Conference (CVPR)*, 2014. 2

[16] Jacky Liang, Wenlong Huang, F. Xia, Peng Xu, Karol Hausman, Brian Ichter, Peter R. Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500, 2022. 3

[17] Zijia Lu and Ehsan Elhamifar. Set-supervised action learning in procedural task videos via pairwise order consistency. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 19903–19913, 2022. 1, 3

[18] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019. 1, 2

[19] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. Large language models as general pattern machines. In *Proceedings of the 7th Conference on Robot Learning (CoRL)*, 2023. 2, 3, 6, 7

[20] Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Gpt3-to-plan: Extracting plans from text using gpt-3. *FinPlan 2021*, page 24, 2021. 3

[21] Vishal Pallagani, Bharath Muppasani, Keerthiram Murugesan, Francesca Rossi, L. Horesh, Biplav Srivastava, F. Fabiano, and Andrea Loreggia. Plansformer: Generating symbolic plans using transformers. *ArXiv*, abs/2212.08681, 2022. 3

[22] Chiara Plizzari, Gabriele Goletto, Antonino Furnari, Siddhant Bansal, Francesco Ragusa, Giovanni Maria Farinella, Dima Damen, and Tatiana Tommasi. An outlook into the future of egocentric vision. *ArXiv*, abs/2308.07123, 2023. 2

[23] Zhaobo Qi, Shuhui Wang, Chi Su, Li Su, Qingming Huang, and Qi Tian. Self-regulated learning for egocentric video activity anticipation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):6715–6730, 2023. 3

[24] Yicheng Qian, Weixin Luo, Dongze Lian, Xu Tang, Peilin Zhao, and Shenghua Gao. Svip: Sequence verification for procedures in videos. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 19890–19902, 2022. 3

[25] Francesco Ragusa, Antonino Furnari, Salvatore Livatino, and Giovanni Maria Farinella. The meccano dataset: Understanding human-object interactions from egocentric videos in an

industrial-like domain. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1569–1578, 2021. 2

[26] Francesco Ragusa, Rosario Leonardi, Michele Mazzamuto, Claudia Bonanno, Rosario Scavo, Antonino Furnari, and Giovanni Maria Farinella. Enigma-51: Towards a fine-grained understanding of human-object interactions in industrial scenarios. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024. 1, 2

[27] 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. 1, 2, 3

[28] F. Sener, D. Chatterjee, D. Shelepov, K. He, D. Singhania, R. Wang, and A. Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022. 1, 2, 3, 4, 8

[29] Anshul Shah, Benjamin Lundell, Harpreet Sawhney, and Rama Chellappa. Steps: Self-supervised key step extraction and localization from unlabeled procedural videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10375–10387, 2023. 3

[30] Sebastian Stein and Stephen J. McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. Association for Computing Machinery, 2013. 2

[31] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2023. 2

[32] Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1207–1216, 2019. 1, 2

[33] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 2, 3

[34] Xiang Wang, Shiwei Zhang, Zhiwu Qing, Yuanjie Shao, Zhengrong Zuo, Changxin Gao, and Nong Sang. Oadtr: Online action detection with transformers. In *Int. Conf. Comput. Vis.*, pages 7565–7575, 2021. 1, 2, 6, 7

[35] 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 *Int. Conf. Comput. Vis.*, pages 20270–20281, 2023. 1, 2, 3

[36] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed Huai hsin Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022, 2022. 3

[37] Jerry W. Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. Larger language models do in-context learning differently. *ArXiv*, abs/2303.03846, 2023. 3

[38] M Zaigham Zaheer, Arif Mahmood, M Haris Khan, Mattia Segu, Fisher Yu, and Seung-Ik Lee. Generative cooperative learning for unsupervised video anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14744–14754, 2022. 2, 3

[39] Y. Zhong, L. Yu, Y. Bai, S. Li, X. Yan, and Y. Li. Learning procedure-aware video representation from instructional videos and their narrations. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2023. 1, 3

[40] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *CVPR*, 2019. 1, 2