

Contrastive Energy Fields for Inference-Time Procedure Planning in Instructional Videos

Mohamed Afham^{1,3} , Christoph Reich^{1,2,3,4} , Oliver Hahn¹ ,
Daniel Cremers^{2,3,4} , and Stefan Roth^{1,3,5} 

¹ TU Darmstadt ² TU Munich ³ ELIZA ⁴ MCML ⁵ hessian.AI
afham.af1al@visinf.tu-darmstadt.de
<https://visinf.github.io/cefito>

Abstract. Procedure planning seeks to estimate a sequence of actions to transition from an observed initial state to a given goal state. Current procedure planning approaches directly predict action sequences from latent representations using feed-forward neural networks or diffusion-based inference. These paradigms treat every action as plausible, lacking the ability to enforce task-specific logical constraints that render certain actions irrelevant or not plausible. We propose CEFITO, a procedure planning approach that learns a predictor to express an action-conditioned representation space. Based on this representation space, we formulate procedure planning as a task-constrained optimization problem. Unlike prior methods, CEFITO explicitly reasons over the action space by omitting irrelevant actions during inference-time planning. This reformulation enables effective procedure planning and achieves state-of-the-art accuracy on two established procedure planning benchmarks.

Keywords: Procedure Planning · Contrastive Learning · Inference Time Optimization

1 Introduction

Planning is a central component of human intelligence [38]. Humans can reason about the next actions to take and their consequences in order to reach a desired goal. The ability to plan is crucial in real-world tasks, including robotic navigation [4,57], autonomous driving [26], virtual reality [45], and healthcare [58]. Procedure planning in instructional videos [12] seeks to plan a sequence of actions given a visual initial and a desired goal state. Unlike classical planning formulations with explicit initial and goal states, procedure planning in instructional videos directly operates on raw visual observation (*i.e.*, imagery), providing a vision-driven planning setting grounded in real-world environments [52,65].

Current approaches from procedure planning employ various deep learning architectures, including transformers [51], diffusion models [55,64], and task-specific procedural knowledge graphs [49]. Still, these approaches use feed-forward or diffusion-based inference to map latent representations to a distribution over action sequences. In contrast, human cognitive planning is a dynamic and iterative process [7,8]. Humans do not predict a single sequence. Instead, humans

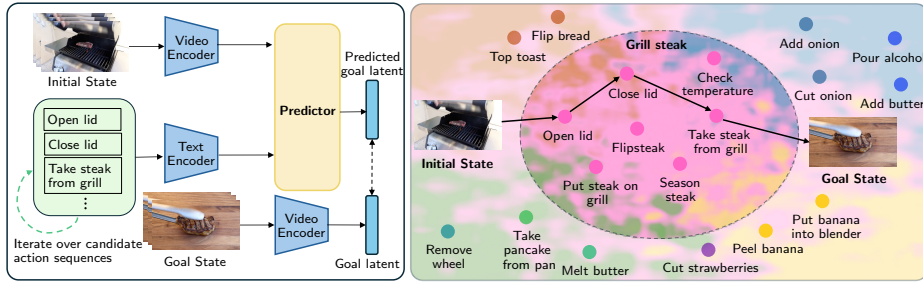


Fig. 1. Overview of CEFITO. *Left:* Given an initial observation and a sequence of actions, our method maps action sequences to a predicted goal latent. The distance to the encoded goal latent defines the candidate’s energy, which we minimize at inference. *Right:* We learn a task-conditioned energy field via contrastive learning. At inference, CEFITO predicts the task (e.g., “Grill steak”), restricting the search to an action subset (pink region). The action sequence with minimal energy is the predicted plan.

infer a set of possible actions and optimize over these to reach a desired goal. Current procedure planning approaches do not mimic this process and cannot, for example, disregard irrelevant actions during inference.

Instead of feed-forward inference, work in classical robot control and planning performs online optimization using predictive models [9]. More recently, action-conditioned world models have demonstrated effective planning using inference-time optimization [23,32,35]. During training, a predictive model is learned to express the transition from a current state to the next. At inference, this predictive model is used for planning by optimizing over a sequence of actions. Such approaches capture the dynamics of the underlying system and the relationship between actions and their outcomes. In procedure planning, modeling action-conditioned visual transitions is crucial, yet challenging due to the scarcity of training data, including intermediate observations. Existing approaches use Large Language Models [42,61], interpolation [64], or probabilistic graphs [41,49] as proxies for unknown intermediate states. However, these approaches rely on rigid structures or require huge training datasets and large-scale pre-training.

Motivated by the effectiveness of inference-time optimization using predictive models, we introduce CEFITO—**C**ontrastive **E**nergy **F**ields for **I**nfere**n**c**e**-**T**ime **O**ptimization in Procedure Planning in Instructional Videos. In contrast to existing procedure planning approaches, CEFITO (*cf.* Fig. 1) decomposes procedure planning into two sub-problems. *First*, CEFITO learns an action-conditioned representation space (*i.e.*, an energy field) in the form of a predictor model, using contrastive learning. This predictor models the transition from an initial state to a goal state, given a sequence of actions. *Second*, at inference-time, we formulate the planning of an action sequence as a task-constrained optimization problem using our predictor model. Unlike prior approaches, CEFITO can explicitly reason over the action space, allowing for the omission of irrelevant actions during planning. This allows CEFITO to perform effective and accurate procedure planning in instructional videos.

Specifically, we make the following contributions: *(i)* We formulate procedure planning as energy minimization over an action-conditioned representation, learned with a contrastive objective that does not require intermediate state supervision. *(ii)* We introduce a task-constrained search procedure that exploits this energy to plan over a restricted, task-relevant action subset at inference time. *(iii)* We evaluate CEFITO on established procedure planning benchmarks of instructional videos. CEFITO achieves state-of-the-art accuracy on two datasets and different planning horizons, while not relying on large language models.

2 Related Work

Procedure Planning in Instructional Videos. Procedure planning in instructional videos [12] aims to plan a sequence of actions transitioning from an initial state to a desired goal state. Both the initial and the goal state are given as visual observations (*i.e.*, video frames). This task is closely related to general task planning [12,22]. Initial approaches for procedure planning employed sequence modeling via sequential latent spaces [12] or adversarial policy learning [6]. While DDN [12] and Ext-GAIL [6] learn using sequential action supervision, subsequent works employed additional supervision such as language [54,62]. In particular, P³IV [62] explores supervision with natural language representations of the action labels instead of one-hot labels, while EGPPP [54], in addition, leverages task labels to condition the planning model. PlaTe [51] introduced a transformer-based model, reducing compounding prediction errors common with single-step models. PDPP [55] and MTID [64] employ diffusion to probabilistically model the prediction action sequences. KEPP [41] leverages a probabilistic procedural knowledge graph to guide state transition. Building on this, ViterbiPlanNet [49] introduced a differentiable viterbi layer upon the probabilistic knowledge graph to enable end-to-end training. A different line of work approached procedure planning using large language models (LLMs) [42,61]. SCHEMA [42] uses an LLM to explicitly model state transitions, while PlanLLM [61] fine-tunes an LLM and trains a step decoder. Different from these approaches that utilize feed-forward neural networks or diffusion models, we learn an action-conditioned representation space using contrastive learning. This allows us to express an energy function and perform task-constrained optimization-based inference.

Inference-Time Optimization for Planning. Inference-time optimization is a well-established paradigm in robotics control and planning, where action sequences are optimized online using forward predictive models [9]. Early visual foresight methods established this direction by combining pixel-level rollouts or learned dynamics with model predictive control to plan trajectories at inference time, without updating policy parameters during inference [16,19,40]. The introduction of world models [2,23,32] has further advanced this idea by moving from high-dimensional pixel space to latent representations, enabling online trajectory optimization at inference time [24,27]. More recently, large-scale visual foundation models [2,11,43] have been used to build world models that support

planning entirely during inference [4,35,63]. Motivated by classical methods, we reformulate procedure planning as an inference-time optimization problem using an action-conditioned energy field.

Contrastive Learning. Contrastive learning aims to learn expressive representations in a self-supervised [10,13], weakly [46], or supervised fashion [31]. At its core, contrastive learning enforces discrimination between pairs of positive and negative data points and has been applied for downstream tasks, including action recognition [30,39,56], action anticipation [44], and temporal action localization [20,28]. A central aspect for the effectiveness of contrastive learning is the choice and number of negative samples [47]. Specifically, hard negatives, which are semantically similar to positive samples, provide an informative training signal. In this work, we utilize supervised contrastive learning [31] in the form of a contrastive triplet-loss [48] to learn an action-conditioned representation space suitable for planning at inference time.

3 Method: CEFITO

In this section, we present CEFITO for procedure planning in instructional videos. We will first revisit the task definition of procedure planning and reformulate the problem as energy minimization over an action-conditioned energy field (*cf.* Sec. 3.1). Next, we introduce a contrastive learning approach for obtaining this action-conditioned energy field in the form of a predictor model (*cf.* Sec. 3.2). Finally, we propose our task-constrained inference-time optimization for planning action sequences (*cf.* Sec. 3.3).

3.1 Procedure Planning as Inference-Time Optimization

Problem Formulation. Procedure planning seeks to estimate a sequence of actions $a_{1:T}$ from a predefined corpus of actions, transforming a given initial state v_s into a given goal state v_g . T represents the planning horizon, *i.e.*, the number of action steps required to achieve the goal state v_g . Both the initial state and the goal state are provided as visual observations (*i.e.*, video frames). Following the established paradigm [12,41,42,51,55,61,64], we encode the visual initial v_s and goal state v_g using a visual feature extractor (*i.e.*, S3D [36] pre-trained on the HowTo100M dataset [37]), resulting in the latent initial state $x_s \in \mathbb{R}^E$ and the latent goal state $x_g \in \mathbb{R}^E$. E denotes the latent dimension of the visual features.

Reformulating Procedure Planning. Prior approaches map (x_s, x_g) directly to an action sequence through a feed-forward or diffusion-based network [51,55,64]. We reformulate procedure planning as an inference-time search over a learned action-conditioned energy field, decomposing it into two sub-tasks. *First*,

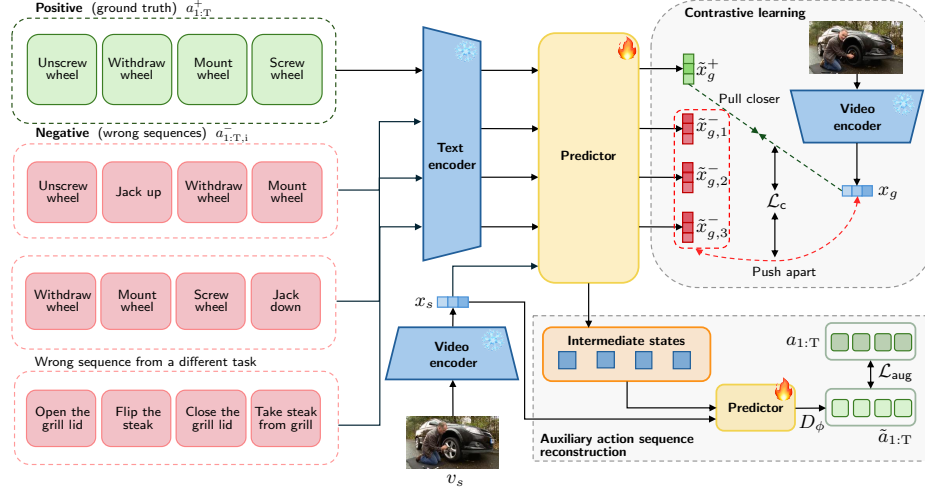


Fig. 2. CEFITO training framework. During training, a positive and N negative (wrong) action sequences are provided. Each sequence is tokenized using a text encoder and fed into the predictor P_θ , alongside the initial latent state x_s . For each sequence we predict a goal state $\tilde{x}_g$, while predictions of negative sequences $\tilde{x}_{g,i}^-$ are pushed away from the target $\tilde{x}_g^+$ and for positive sequences $\tilde{x}_g^+$ pulled together. An auxiliary loss $\mathcal{L}_{aux}$ supervises auxiliary latents of P_θ via a decoding head D_ϕ .

we learn a predictor model that maps a candidate action sequence and the initial state x_s to the resulting goal embedding, acting as an energy. *Second*, at inference, we optimize over candidate action sequences and select the one whose predicted goal embedding is closest to the observed latent state x_g .

3.2 Action-Conditioned Energy Field

To perform planning at inference time, we aim to learn an action-conditioned energy field capturing the transition from x_s to x_g , given an action sequence. Perpetually, we train a predictor model P_θ , mapping an action sequence $a_{1:T}$ (textual form) and an initial state x_s to a predicted goal state $\tilde{x}_g \in \mathbb{R}^E$ by

$$\tilde{x}_g = P_\theta(x_s, a_{1:T}). \quad (1)$$

The objective of P_θ is to predict $\tilde{x}_g$ close to the observed goal x_g when conditioned on the correct action sequence $a_{1:T}^+$, and far from x_g for any incorrect sequence $a_{1:T}^-$. During inference, we can search for an action sequence that minimizes distance in latent space. Formally, $d(P_\theta(x_s, a_{1:T}), x_g)$ can be seen as an energy function, where $d(\cdot, \cdot)$ denotes a distance metric.

We implement our predictor model P_θ as a transformer [53]. We use the initial latent state x_s as an input token. The action sequence is encoded in language form using CLIP [46]. The resulting embeddings are used as additional input tokens. We visualize the predictor architecture in Fig. 2.

Contrastive Training. We train our predictor P_θ using contrastive learning. In particular, we use a margin-based triplet loss [48]. Given the *correct* action sequence $a_{1:T}^+$, we obtain $\tilde{x}_g^+ \in \mathbb{R}^E$ using P_θ and x_s (*cf.* Eq. (1)). Analogously, given N *wrong* sequences $\{a_{1:T,1}^-, \dots, a_{1:T,N}^-\}$, we obtain $\tilde{x}_{g,i}^- \in \mathbb{R}^{C \times N}$. Given the ground truth latent goal state, we compute our contrastive loss $\mathcal{L}_c$ using

$$\mathcal{L}_c = \frac{1}{N} \sum_{i=1}^N \max(\|x_g - \tilde{x}_g^+\|_2 - \|x_g - \tilde{x}_{g,i}^-\|_2 + \tau_i, 0). \quad (2)$$

Here, $\tau_i \in \mathbb{R}^+$ is the triplet margin per negative sample i , enforcing a minimum separation and preventing the degenerate solution that collapses all embeddings. We train our predictor’s weights θ to minimize Eq. (2).

Negative Action Sequence Sampling. Negative samples are critical for obtaining a discriminative space [29,47]. To learn an energy field that captures both high-level (*e.g.*, discriminate between tasks) and low-level structure (*e.g.*, detect wrong action sequence order), we employ mixed-negative sampling. By generating both hard and easy negative samples, we impose high and low-level structure.

Given a ground truth action sequence with initial, goal state, and task label $c \in \mathcal{C}$, we generate N negative samples, *i.e.*, wrong sequences. Specifically, rN of the negative samples are hard negatives and $(1-r)N$ are easy negatives. $r \in [0, 1]$ is the hard/easy negative sample ratio, controlling the balance between both. Hard negative samples share the same high-level task c but differ in ordering or composition. Easy negatives are sequences sampled from tasks other than c .

Adaptive Margins. The triplet-loss margin τ_i (*cf.* Eq. (2)) enforces a minimum distance between the positive $\|x_g - x_g^+\|_2$ and negative pair $\|x_g - x_{g,i}^-\|_2$. Using a fixed margin for every negative is suboptimal because negatives differ in similarity to the ground truth: An action sequence only differing from the ground truth in the ordering of the actions (*i.e.*, hard negative sample) should ideally entail a smaller distance than an action sequence of unrelated actions (*i.e.*, easy negative sample). To enforce this structure, we use adaptive margins, assigning a margin to each negative sample based on the action overlap with the positive sample. In particular, given the positive action sequence $a_{1:T}^+$ (*i.e.*, ground truth) and a negative action sequence $a_{1:T,i}^-$, we extract the sets of unique actions of the positive $\mathbb{A}^+$ and of the negative sequence $\mathbb{A}_i^-$. Next, we compute τ_i using

$$\tau_i = \tau_{\min} + (\tau_{\max} - \tau_{\min}) \frac{|\mathbb{A}_i^- \setminus \mathbb{A}^+|}{|\mathbb{A}_i^-|}. \quad (3)$$

Here, the hyperparameters $\tau_{\min} \in \mathbb{R}^+$ denotes the minimum and $\tau_{\max} \in \mathbb{R}^+$ the maximum margin, bounding the per negative margin $\tau_i \in [\tau_{\min}, \tau_{\max}]$.

Auxiliary Action Sequence Reconstruction While Eq. (2) supervises the final output representation, we further enforce expressive intermediate representations using an action sequence reconstruction approach. As visual representations of the intermediate states are not available, we employ a mask token

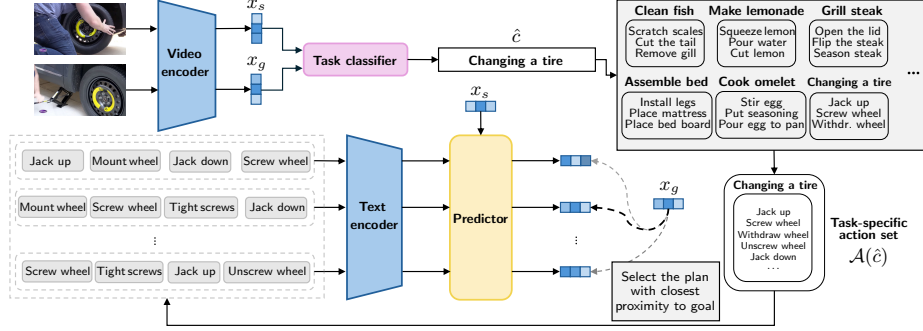


Fig. 3. CEFITO inference. A task classifier predicts $\hat{c}$ from (x_s, x_g) and restricts the search to $\mathcal{A}(\hat{c})$. The predictor P_θ scores every candidate sequence in $\mathcal{A}(\hat{c})$. The sequence approximating x_g best, forms the action sequence prediction $\tilde{a}_{1:T}$.

modeling [15,25] approach. We feed the initial latent state x_s and the action sequence $a_{1:T}$ (tokenized using the text encoder) to our predictor P_θ . Additionally, while omitted in Eq. (1) for clarity, we also feed a learnable masked token for each action into the predictor. The predictor outputs the goal state representation $\tilde{x}_g$, we further extract the intermediate representation $\{\tilde{x}_l\}_{l=1}^T$ corresponding to each of the masked tokens. After obtaining $\{\tilde{x}_l\}_{l=1}^{T-1}$, now reverse the masking and all actions $a_{1:T}$ and feed $\{\tilde{x}_l\}_{l=1}^{T-1}$ into our predictor and a decoding head D_ϕ . This yields a reconstruction of actions $\tilde{a}_{1:T}$. Each reconstructed action is supervised using the ground-truth actions $a_{1:T}$ using $\mathcal{L}_{\text{aux}} = \frac{1}{T} \sum_{k=1}^T \mathcal{L}_{\text{CE}}(\tilde{a}_k, a_k)$. Here, $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss, and actions are mapped to the vocabulary of actions. We omit this mapping for the sake of compactness.

Our full predictor loss composes both the contrastive loss $\mathcal{L}_c$ and our auxiliary loss $\mathcal{L}_{\text{aux}}$, weighted by λ , *i.e.*, $\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_{\text{aux}}$. We visualize our training in Fig. 2.

3.3 Task-Constrained Optimization for Planning

Equipped with our predictor P_θ , we now seek to infer an action sequence, given the initial x_s and goal state x_g , in latent space (*cf.* Fig. 3). We formulate inference as a discrete search over the action space $\mathcal{A}$. Optimizing over the full space is expensive. To this end, we learn a lightweight classifier that estimates the high-level task $c \in \mathcal{C}$. Implemented as a multilayer perceptron, the classifier takes in (x_s, x_g) , and predicts the task $\hat{c} \in \mathcal{C}$. We train the classifier using cross-entropy.

Using the estimated high-level task $\hat{c}$, we reduce the search space from the full action space to a task-specific subspace $\mathcal{A}(\hat{c})$. This subspace only captures actions relevant to the specific task and is derived from the training data. Using $\mathcal{A}(\hat{c})$, the initial state x_s , and goal state x_g , we infer an action sequence $\tilde{a}_{1:T}$ by

$$\tilde{a}_{1:T} = \arg \min_{\tilde{a}_{1:T} \in \mathcal{A}(\hat{c})} \|P_\theta(x_s, \tilde{a}_{1:T}) - x_g\|_2. \quad (4)$$

This inference approach enables procedure planning by considering only relevant actions, ignoring irrelevant ones. Our inference approach is shown in Fig. 3.

Table 1. Results on CrossTask [65]. We compare CEFITO with the state of the art on CrossTask val., using different planning horizons, and report SR, mAcc, and mIoU (all in %, $\uparrow$). Best results highlighted in red ; second-best results in orange . For completeness, we report baselines not adhering to the protocol by [49] in gray .

Method	T=3			T=4		
	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$
WLTD0 [17]	1.87	21.64	31.70	0.77	17.92	26.43
UAAA [18]	2.15	20.21	30.87	0.98	19.86	27.09
UPN [50]	2.89	24.39	31.56	1.19	21.59	27.85
DDN [12]	12.18	31.29	47.48	5.97	27.10	48.46
PlaTe [51]	16.00	36.17	65.91	14.00	35.29	55.36
Ext-GAIL [6]	21.27	49.46	61.70	16.41	43.05	60.93
P ³ IV [62]	23.34	49.96	73.89	13.40	44.16	70.01
EGPP [54]	26.40	53.02	74.05	16.49	48.00	70.16
Qwen2.5-VL-32B [3]	11.48	36.35	69.52	5.56	31.22	66.31
Qwen2.5-32B [60]	25.14	56.10	80.92	9.22	46.32	76.15
Gemini 2.5 Pro [21]	29.18	57.90	81.48	14.00	51.33	78.58
Qwen3-30B [59]	23.37	55.96	81.16	10.59	49.06	78.03
Qwen3-30B + PKG [49]	23.31	56.15	81.06	10.96	48.77	77.48
PKG beam search [49]	22.38 ± 0.26	55.74 ± 0.25	80.92 ± 0.26	9.30 ± 0.22	47.65 ± 0.54	78.25 ± 0.42
PDPP [55]	36.73 ± 0.59	61.96 ± 0.59	83.20 ± 0.33	21.47 ± 2.09	55.66 ± 1.64	80.68 ± 0.83
KEPP [41]	34.93 ± 2.60	60.34 ± 1.61	82.67 ± 0.69	22.34 ± 0.43	55.24 ± 0.30	80.58 ± 0.25
PlanLLM [61]	36.84 ± 1.21	61.56 ± 1.03	83.23 ± 0.53	22.91 ± 1.39	55.29 ± 1.54	81.03 ± 0.47
SCHEMA [42]	37.24 ± 0.60	62.69 ± 0.28	83.94 ± 0.23	24.18 ± 0.47	57.02 ± 0.64	81.46 ± 0.19
ViterbiPlanNet [49]	38.45 ± 0.32	63.07 ± 0.17	83.89 ± 0.16	24.64 ± 0.30	57.00 ± 0.42	81.18 ± 0.44
CEFITO (Ours)	39.62 ± 0.24	64.12 ± 0.31	84.29 ± 0.21	24.76 ± 0.43	57.53 ± 0.37	81.58 ± 0.25

4 Experiments

We evaluate CEFITO on two established procedure planning benchmarks and compare against existing state-of-the-art approaches (*cf.* Sec. 4.1). Moreover, we analyze the main components and hyperparameters of our approach (*cf.* Sec. 4.2).

Datasets. We evaluate on two procedure planning datasets, CrossTask [65] and COIN [52]. CrossTask contains 2750 videos spanning 18 high-level tasks with a total of 105 distinct actions. Each video entails 7.6 actions on average. COIN comprises 11 827 videos of 180 different tasks, covering 778 unique actions. We adopt the unified evaluation protocol by Seminara *et al.* [49], where we train using five random seeds and report the mean and the 90% confidence interval.

Metrics. We follow existing work [41,42,49,55,61,62,64] and utilize three standard metrics to measure procedure planning accuracy. *First*, we utilize the Success Rate (SR), measuring the percentage of predicted action sequences that exactly match the corresponding ground truth sequences. *Second*, we report the Mean Accuracy (mAcc), measuring the average proportion of correctly predicted actions across all step-wise positions. *Third*, we use the Mean Intersection over Union (mIoU), computing the overlap between the predicted and ground truth action sequences. In particular, we follow the element-wise mIoU formulation suggested by Seminara *et al.* [49]. Among these three metrics, SR is the strictest metric. All metrics are reported in %.

Table 2. Results on COIN [52]. We compare CEFITO with the state of the art on COIN val., using different planning horizons, and report SR, mACC, and mIoU (all in %, $\uparrow$). Best results are highlighted in red ■ and second-best results in orange ■. For completeness, we report baselines not adhering to the protocol by [49] in gray ■.

Method	T=3			T=4		
	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$
DDN [12]	13.90	20.19	64.78	11.13	17.71	68.06
P ³ IV [62]	15.40	21.67	76.31	11.32	18.85	70.53
EGPP [54]	19.57	31.42	84.95	13.59	26.72	84.72
Qwen2.5-VL-32B [3]	3.65	17.51	52.10	1.87	17.05	55.66
Qwen2.5-32B [60]	14.97	36.34	78.74	4.98	27.45	71.64
Gemini 2.5 Pro [21]	17.02	38.87	78.73	8.10	31.90	71.70
Qwen3-30B [59]	14.52	36.56	78.07	4.64	28.85	70.45
Qwen3-30B + PKG [49]	14.63	36.53	78.11	4.78	29.00	71.04
PKG beam search [49]	13.32 $\pm$ 0.34	37.42 $\pm$ 1.19	78.93 $\pm$ 2.06	5.14 $\pm$ 0.60	31.29 $\pm$ 3.64	74.26 $\pm$ 5.38
PDPP [55]	22.37 $\pm$ 0.57	44.60 $\pm$ 0.16	83.00 $\pm$ 0.42	15.21 $\pm$ 0.34	41.01 $\pm$ 0.32	81.64 $\pm$ 0.48
KEPP [41]	13.85 $\pm$ 7.49	28.40 $\pm$ 12.26	62.54 $\pm$ 14.35	15.20 $\pm$ 1.27	33.39 $\pm$ 0.73	67.79 $\pm$ 1.29
PlanLLM [61]	33.44 $\pm$ 0.15	51.05 $\pm$ 0.46	84.66 $\pm$ 0.41	23.19 $\pm$ 0.32	45.70 $\pm$ 0.33	83.44 $\pm$ 0.39
SCHEMA [42]	32.89 $\pm$ 0.61	50.84 $\pm$ 0.47	83.98 $\pm$ 0.67	22.33 $\pm$ 0.92	45.21 $\pm$ 1.05	82.93 $\pm$ 0.25
ViterbiPlanNet [49]	33.99 $\pm$ 0.23	50.87 $\pm$ 0.17	83.88 $\pm$ 0.31	23.92 $\pm$ 0.29	45.63 $\pm$ 0.55	82.56 $\pm$ 0.44
CEFITO	34.11 $\pm$ 0.24	51.18 $\pm$ 0.22	84.70 $\pm$ 0.47	24.25 $\pm$ 0.34	46.13 $\pm$ 0.63	83.24 $\pm$ 0.26

Implementation Details. We implement our predictor model as a four-layer transformer [53] with 6 attention heads, and a hidden dimension of 384. We train the predictor using the AdamW optimizer [33] and a learning rate of 5×10^{-4} . We sample 50 negative samples per positive training sequence with $r = 0.8$. The minimum $\tau_{\min}$ and maximum margin $\tau_{\max}$ are set to 0.01 and 0.1, respectively. For obtaining the visual representations of the initial state and the goal state, we follow existing work [12,41,42,51,55,61,64] and use a frozen S3D [36] encoder pre-trained on HowTo100M [37]. Textual action sequences are encoded using the pre-trained text encoder from CLIP-ViT-B [46] and kept frozen during training.

4.1 Comparison to the State of The Art

We compare CEFITO to recent state-of-the-art approaches for procedure planning in instructional videos. Table 1 presents the results on CrossTask for a planning horizon of $T = 3$ and $T = 4$. CEFITO outperforms the recent state of the art on all procedure planning accuracy metrics and both planning horizons. Notably, CEFITO outperforms recent approaches that utilize large language models (LLMs). Without relying on LLMs, CEFITO, for example, outperforms the LLM-based approach SCHEMA [42] by 2.38 % in SR for $T = 3$. In comparison to the recent state-of-the-art ViterbiPlanNet [49], CEFITO improves success rate by 1.17 % and 0.12 % for $T = 3$ and $T = 4$, respectively.

In Tab. 2, we report the results on the COIN dataset for a planning horizon $T = 3$ and $T = 4$. CEFITO achieves the best mean SR, mAcc, and mIoU at $T = 3$, and the best SR and mAcc at $T = 4$. Improvements over the recent state-of-the-art ViterbiPlanNet [49] are smaller on COIN than on CrossTask (0.12 % and 0.33 % SR at $T = 3$ and $T = 4$, respectively). We attribute the

Table 3. CEFITO training analysis. Component-wise analysis of CEFITO on CrossTask val. and report SR, mACC & mIoU (all in %, $\uparrow$). We add components to the previous row until achieving CEFITO. For reference, we include the recent state-of-the-art ViterbiPlanNet. Best results in red ■; second-best results in orange ■.

Configuration	T=3			T=4		
	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$
ViterbiPlanNet [49]	38.45	63.07	83.89	24.64	57.00	81.18
Baseline	13.65	47.82	69.53	9.24	35.44	67.76
+ triplet contrastive loss	37.58	63.21	83.67	23.54	56.18	80.91
+ adaptive margin	38.74	63.79	84.02	24.23	57.01	81.22
+ auxiliary regularization (CEFITO)	39.62	64.12	84.29	24.76	57.53	81.58

Table 4. Text encoder analysis. We analyze the choice of text encoder for CEFITO on CrossTask val. and report SR, mACC, and mIoU (all in %, $\uparrow$). Best results are highlighted in red ■ and second-best results in orange ■.

Text Encoder	T=3			T=4		
	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$
Random embedding	35.10	61.20	82.50	21.30	54.10	79.80
Flan-T5-base [14]	37.80	63.00	83.60	23.10	56.20	80.90
CLIP [46]	39.62	64.12	84.29	24.76	57.53	81.58

smaller benefits to COIN’s much wider task distribution (180 tasks *vs.* 18 on CrossTask), which reduces the number of training videos available per task and thereby weakens the intra-task hard-negative signal that drives our contrastive training. Even so, CEFITO achieves state-of-the-art accuracy on COIN without significant additional training data, confirming the trend observed on CrossTask.

4.2 Analyzing CEFITO

Learning Objective. Contrastive learning is the core component of CEFITO, as it explicitly enforces a discriminative energy field in which the predicted goal is encoded near x_g for correct sequences and far from x_g for incorrect ones. In Tab. 3, we report the contribution of each core component on CrossTask. The baseline replaces our contrastive objective with a plain $\mathcal{L}_2$ regression to x_g and severely underperforms w.r.t. ViterbiPlanNet. Adding the triplet contrastive loss yields a +23.93 jump in SR for $T = 3$. The adaptive margin contributes an additional +1.16% in SR by allowing hard negatives to receive a smaller separation than easy ones. Our auxiliary action-reconstruction loss further increases SR by +0.88%, encouraging informative auxiliary latents.

Text Encoder Analysis. We compare three options for embedding the action labels fed to P_θ : a randomly initialized embedding trained from scratch, frozen Flan-T5-base [14], and frozen CLIP (text encoder only) [46]. Table 4 reports the results on CrossTask for $T = 3$ and $T = 4$ using the different options. Both pre-trained encoders outperform the random embedding across all metrics, indicating

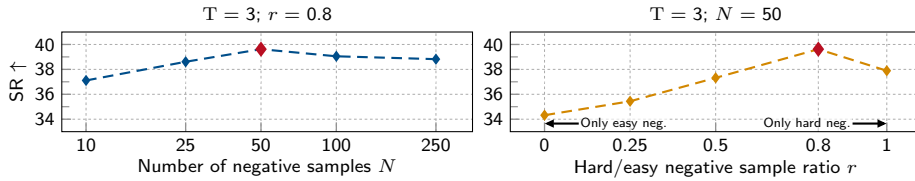


Fig. 4. Negative sampling analysis. We analyze the impact of the number of negative samples N (*left*) and the hard/easy negative sample ratio r (*right*) on CrossTask with $T = 3$ using SR (in %, ↑). For $r = 0$, only easy negatives are used. *Vice versa*, $r = 1$ generates only hard negatives. We indicate our default values in red ■.

Table 5. Task classifier results. We report the accuracy (in %, ↑) of our task classifier on the high-level class tasks of CrossTask [65] as well as COIN [52] for two different planning horizons.

Dataset	T=3	T=4
CrossTask	92.43	92.98
COIN	79.42	79.42

Table 6. Oracle experiment. We report oracle results by replacing the task classifier prediction used for inference with the ground truth task on CrossTask and COIN, using SR, mACC & mIoU (all in %, ↑). For reference, we also report CEFITO (*i.e.*, w/ task classifier) in ■.

Datasets	Setting	T=3			T=4		
		SR ↑	mAcc ↑	mIoU ↑	SR ↑	mAcc ↑	mIoU ↑
CrossTask	CEFITO	39.62	64.12	84.29	24.76	57.53	81.58
	Oracle	41.32	66.42	88.24	25.52	58.85	85.02
COIN	CEFITO	34.11	51.18	84.70	24.25	46.13	83.24
	Oracle	38.33	59.62	98.15	29.51	55.85	97.95

that pre-training provides a stronger foundation than what can be learned from the procedure-planning supervision alone. CLIP consistently outperforms Flan-T5-base despite being the smaller model. We attribute this to CLIP’s multimodal contrastive pre-training, which aligns text representations with visual concepts and is well-aligned with our planning task.

Negative-Sequence Selection. We analyze the two hyperparameters of our negative-sampling strategy: the number of negatives N and the hard/easy negative sample ratio r . Figure 4 reports the success rate on CrossTask with $T = 3$. Success rate is low for $N = 10$ due to insufficient contrastive supervision and peaks at $N = 50$. Further increasing N does not lead to improvements; accuracy saturates. The hard-negative ratio interpolates between only easy ($r = 0$, inter-task) and only hard ($r = 1$, intra-task) negatives. Performance peaks at $r = 0.8$, indicating that a mixture biased towards hard intra-task negatives provides the strongest supervision while retaining inter-task separation.

Task Classifier Analysis. During inference, we use the prediction of our task classifier to effectively constrain the search space and reduce runtime (*cf.* Fig. 5). However, an incorrect task prediction always results in a wrong action sequence. To analyze this, we report the accuracy of our classifier in Tab. 5. On CrossTask, we achieve an accuracy above 90 %, introducing only a minor error. On COIN,

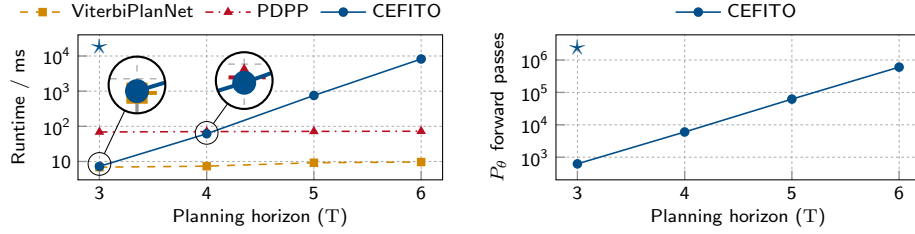


Fig. 5. Inference runtime results. *Left:* We report inference runtime (in ms, $\downarrow$) over different planning horizons for inferring a single action sequence on the CrossTask dataset [65]. All runtimes are reported using the same hardware (*single* A100 80 GB GPU). *Right:* We report the average number of forward passes through our predictor P_θ required for inference. While CEFITO (in $\blacksquare$) provides a comparable inference runtime for $T \leq 4$ to diffusion (PDPP [55] in $\blacktriangle$) and feed-forward (ViterbiPlanNet [49] in $\blacksquare$) approaches, the runtime is significantly worse for $T \geq 5$. For reference, we also report CEFITO without constraining the search space, indicated using $\star$.

our classifier introduces a more significant error. In about 20 % of the validation samples, our task classifier is the cause of a wrong action sequence prediction.

We demonstrated that our task classifier can make initial errors, causing inference to fail (*cf.* Tab. 5). To analyze the impact of task classifier errors on the downstream accuracy, we perform an oracle experiment in Tab. 6. In particular, we replace the classifier’s high-level task prediction with the ground truth and then run inference. This provides an upper bound on the downstream accuracy of CEFITO. While using the ground-truth class for inference (oracle setting) yields consistent improvements on both datasets, the improvements on COIN are more significant. This demonstrates that an improved task classification accuracy can directly improve downstream planning.

Limitations & Failed Experiments. CEFITO achieves state-of-the-art accuracy on CrossTask (*cf.* Tab. 1) and COIN (*cf.* Tab. 2), demonstrating that learning an expressive action-condition energy field and performing task-constrained optimization is feasible and effective. While we restrict the set of possible actions using a learned task constraint, significantly reducing runtime (*cf.* Fig. 5), search complexity still grows exponentially with the planning horizon T . We demonstrate this empirically in Fig. 5. For shorter sequences, inference runtime is manageable and comparable to existing methods (*cf.* Fig. 5 (*left*)). For larger sequences $T \geq 5$, the runtime of CEFITO significantly increases as more predictor forward passes need to be performed (*cf.* Fig. 5 (*right*)). Concretely, for a planning horizon of 5, CEFITO is about one order of magnitude slower than PDPP. For $T = 6$ CEFITO is even two orders of magnitude slower. To overcome this limitation, approximate optimization strategies, such as beam search [34], offer a potential avenue to reduce inference runtime. While we optimize over a set of discrete actions, adapting gradient-based inference [5] to this setting could provide an additional avenue for improving inference-time optimization runtime.

Table 7. Results on NIV [1]. We compare CEFITO with the state-of-the-art on NIV val., using different planning horizons, and report SR, mACC, and mIoU (all in %, $\uparrow$). Best results are highlighted in red and second-best results in orange .

Method	T=3			T=4		
	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$	SR $\uparrow$	mAcc $\uparrow$	mIoU $\uparrow$
Qwen2.5-VL-32B [3]	7.41	27.65	59.73	5.26	28.84	60.21
Qwen2.5-32B [60]	24.07	43.46	71.88	23.25	41.89	73.91
Gemini 2.5 Pro [21]	24.07	43.46	71.86	22.37	40.35	73.05
Qwen3-30B [59]	24.81	42.84	70.80	22.37	41.23	73.90
Qwen3-30B + PKG [49]	25.19	43.95	71.98	21.93	41.67	74.43
PKG beam search [49]	24.96 ± 1.93	43.46 ± 2.42	72.18 ± 0.55	21.23 ± 0.96	40.86 ± 0.83	72.69 ± 0.75
PDPP [55]	26.52 ± 1.56	45.58 ± 1.85	74.89 ± 0.85	21.40 ± 0.53	40.20 ± 2.00	72.82 ± 1.84
KEPP [41]	27.56 ± 1.48	45.93 ± 2.37	74.36 ± 0.97	22.54 ± 1.93	42.46 ± 1.49	73.11 ± 0.94
PlanLLM [61]	30.00 ± 1.41	44.35 ± 2.52	73.60 ± 1.66	23.42 ± 1.40	41.95 ± 2.81	72.32 ± 0.91
SCHEMA [42]	26.30 ± 1.49	42.77 ± 2.12	73.04 ± 1.42	24.39 ± 1.84	41.14 ± 3.62	73.13 ± 1.97
ViterbiPlanNet [49]	32.37 ± 0.96	46.96 ± 1.75	73.85 ± 0.85	27.54 ± 0.70	45.55 ± 1.89	74.71 ± 1.19
CEFITO (Ours)	22.46 ± 1.43	40.37 ± 2.21	70.26 ± 1.43	19.67 ± 1.76	37.25 ± 1.97	70.75 ± 1.39

Beyond the strong results on CrossTask and COIN (*cf.* Tabs. 1 & 2), we also present a failed experiment. In particular, we report results on NIV [1] in Tab. 7. CEFITO yields a suboptimal accuracy. We root this in the fact that NIV is significantly smaller than both CrossTask and COIN. COIN comprises over 11 k videos, NIV only contains 150, about two orders of magnitude less. We suspect that contrastive learning collapses for a small number of training videos.

5 Conclusion

We introduced CEFITO, an inference-time optimization framework for procedure planning in instructional videos. By learning a predictor model that maps a candidate action sequence and the initial state to a predicted goal embedding, we can approach planning using task-constrained inference-time optimization. CEFITO demonstrates state-of-the-art accuracy on CrossTask and COIN. Unlike current approaches, we plan using optimization at inference-time and do not rely on large language models.

Acknowledgments. This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 866008). Additionally, this project is also funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy (EXC-3066/1 “The Adaptive Mind”, Project No. 533717223, EXC-3057/1 “Reasonable Artificial Intelligence”, Project No. 533677015). Mohamed Afham & Christoph Reich are supported by the Konrad Zuse School of Excellence in Learning and Intelligent Systems (ELIZA) through the DAAD programme Konrad Zuse Schools of Excellence in Artificial Intelligence, sponsored by the Federal Ministry of Education and Research. This work was also supported by the ERC Advanced Grant SIMULACRON, the Georg Nemetschek Institute project AI4TWINNING, and the DFG project 4D-YouTube CR 250/26-1.

References

1. Alayrac, J., Bojanowski, P., Agrawal, N., Sivic, J., Laptev, I., Lacoste-Julien, S.: Unsupervised learning from narrated instruction videos. In: CVPR. pp. 4575–4583 (2016). <https://doi.org/10.1109/CVPR.2016.495>
2. Assran, M., Bardes, A., Fan, D., Garrido, Q., Howes, R., Komeili, M., Muckley, M.J., Rizvi, A., et al.: V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. arXiv:2506.09985 [cs.AI] (2025). <https://doi.org/10.48550/arXiv.2506.09985>
3. Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., et al.: Qwen2.5-VL technical report. arXiv:2502.13923 [cs.CV] (2025). <https://doi.org/10.48550/arXiv.2502.13923>
4. Bar, A., Zhou, G., Tran, D., Darrell, T., LeCun, Y.: Navigation world models. In: CVPR. pp. 15791–15801 (2025). <https://doi.org/10.1109/CVPR52734.2025.01472>
5. Belanger, D., McCallum, A.: Structured prediction energy networks. In: ICML. vol. 48, pp. 983–992 (2016), <https://proceedings.mlr.press/v48/belanger16.html>
6. Bi, J., Luo, J., Xu, C.: Procedure planning in instructional videos via contextual modeling and model-based policy learning. In: ICCV. pp. 15591–15600 (2021)
7. Botvinick, M.M., Toussaint, M.: Planning as inference. Trends Cogn. Sci. **16**(10), 485–488 (2012). <https://doi.org/10.1016/j.tics.2012.08.006>
8. Callaway, F., Lieder, F., Krueger, P.M., Griffiths, T.L.: Rational use of cognitive resources in human planning. Nat. Hum. Behav. **6**(8), 1115–1125 (2022). <https://doi.org/10.1038/s41562-022-01332-8>
9. Camacho, E.F., Bordons, C.: Model Predictive Control. Springer (2007). <https://doi.org/10.1007/978-0-85729-398-5>
10. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. In: NeurIPS (2020), <https://proceedings.neurips.cc/paper/2020/file/70feb62b69f16e0238f741fab228fec2-Paper.pdf>
11. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: ICCV. pp. 9630–9640 (2021). <https://doi.org/10.1109/ICCV48922.2021.00951>
12. Chang, C., Huang, D., Xu, D., Adeli, E., Fei-Fei, L., Niebles, J.C.: Procedure planning in instructional videos. In: ECCV. pp. 334–350 (2020). https://doi.org/10.1007/978-3-030-58621-8_20
13. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. In: ICML. vol. 119, pp. 1597–1607 (2020), <https://proceedings.mlr.press/v119/chen20j.html>
14. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dezhghani, M., Brahma, S., Webson, A., Gu, S.S., Dai, Z., Suzgun, M., et al.: Scaling instruction-finetuned language models. J. Mach. Learn. Res. **25**(1), 3381–3433 (2024), <https://jmlr.org/papers/v25/23-0870.html>
15. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL. pp. 4171–4186 (2019). <https://doi.org/10.18653/v1/N19-1423>
16. Ebert, F., Finn, C., Dasari, S., Xie, A., Lee, A.X., Levine, S.: Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. arXiv:1812.00568 [cs.RO] (2018). <https://doi.org/10.48550/arXiv.1812.00568>
17. Ehsani, K., Bagherinezhad, H., Redmon, J., Mottaghi, R., Farhadi, A.: Who let the dogs out? Modeling dog behavior from visual data. In: CVPR. pp. 4051–4060 (2018). <https://doi.org/10.1109/CVPR.2018.00426>

18. Farha, Y.A., Gall, J.: Uncertainty-aware anticipation of activities. In: ICCVW. pp. 1197–1204 (2019). <https://doi.org/10.1109/ICCVW.2019.00151>
19. Finn, C., Levine, S.: Deep visual foresight for planning robot motion. In: ICRA. pp. 2786–2793 (2017). <https://doi.org/10.1109/ICRA.2017.7989324>
20. Gao, J., Chen, M., Xu, C.: Fine-grained temporal contrastive learning for weakly-supervised temporal action localization. In: CVPR. pp. 19967–19977 (2022)
21. Gemini Team: Gemini 2.5: Pushing the frontier with advanced reasoning, multi-modality, long context, and next generation agentic capabilities. arXiv:2507.06261 [cs.CL] (2025). <https://doi.org/10.48550/arXiv.2507.06261>
22. Ghallab, M., Nau, D., Traverso, P.: Automated Planning: Theory and practice. Elsevier (2004). <https://doi.org/10.1016/B978-1-55860-856-6.X5000-5>
23. Ha, D., Schmidhuber, J.: World models. arXiv:1803.10122 [cs.LG] (2018)
24. Hafner, D., Lillicrap, T.P., Fischer, I., Villegas, R., Ha, D., Lee, H., Davidson, J.: Learning latent dynamics for planning from pixels. In: ICML. vol. 97, pp. 2555–2565 (2019), <https://proceedings.mlr.press/v97/hafner19a.html>
25. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: CVPR. pp. 16000–16009 (2022). <https://doi.org/10.1109/CVPR52688.2022.01553>
26. Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., et al.: Planning-oriented autonomous driving. In: CVPR. pp. 17853–17862 (2023). <https://doi.org/10.1109/CVPR52729.2023.01712>
27. Janner, M., Du, Y., Tenenbaum, J.B., Levine, S.: Planning with diffusion for flexible behavior synthesis. In: ICML. vol. 162, pp. 9902–9915 (2022), <https://proceedings.mlr.press/v162/janner22a.html>
28. Ju, C., Zheng, K., Liu, J., Zhao, P., Zhang, Y., Chang, J., Tian, Q., Wang, Y.: Distilling vision-language pre-training to collaborate with weakly-supervised temporal action localization. In: CVPR. pp. 14751–14762 (2023). <https://doi.org/10.1109/CVPR52729.2023.01417>
29. Kalantidis, Y., Saryildiz, M.B., Pion, N., Weinzaepfel, P., Larlus, D.: Hard negative mixing for contrastive learning. In: NeurIPS. pp. 21798–21809 (2020), https://proceedings.neurips.cc/paper_files/paper/2020/file/f7cade80b7cc92b991cf4d2806d6bd78-Paper.pdf
30. Khorasgani, S.H., Chen, Y., Shkurti, F.: SLIC: Self-supervised learning with iterative clustering for human action videos. In: CVPR. pp. 16070–16080 (2022). <https://doi.org/10.1109/CVPR52688.2022.01562>
31. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. In: NeurIPS. vol. 33, pp. 18661–18673 (2020), <https://proceedings.neurips.cc/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf>
32. LeCun, Y.: A path towards autonomous machine intelligence. Open Review (2022), <https://openreview.net/pdf?id=BZ5a1r-kVsf>
33. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2017). <https://doi.org/10.48550/arXiv.1711.05101>
34. Lowerre, B.T.: The HARPY speech recognition system. Carnegie Mellon University (1976), <https://stacks.stanford.edu/file/druid:rq916rn6924/rq916rn6924.pdf>
35. Maes, L., Lidec, Q.L., Scieur, D., LeCun, Y., Balestriero, R.: LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. arXiv:2603.19312 [cs.LG] (2026). <https://doi.org/10.48550/arXiv.2603.19312>
36. Miech, A., Alayrac, J., Smaira, L., Laptev, I., Sivic, J., Zisserman, A.: End-to-end learning of visual representations from uncurated instructional videos. In: CVPR. pp. 9876–9886 (2020). <https://doi.org/10.1109/CVPR42600.2020.00990>

37. Miech, A., Zhukov, D., Alayrac, J., Tapaswi, M., Laptev, I., Sivic, J.: HowTo100M: Learning a text-video embedding by watching hundred million narrated video clips. In: ICCV. pp. 2630–2640 (2019). <https://doi.org/10.1109/ICCV.2019.00272>
38. Miller, G.A., Galanter, E., Pribram, K.H.: Plans and the structure of behavior. Henry Holt and Co. (1960). <https://doi.org/10.1037/10039-000>
39. Morgado, P., Vasconcelos, N., Misra, I.: Audio-visual instance discrimination with cross-modal agreement. In: CVPR. pp. 12475–12486 (2021). <https://doi.org/10.1109/CVPR46437.2021.01229>
40. Nagabandi, A., Kahn, G., Fearing, R.S., Levine, S.: Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In: ICRA. pp. 7559–7566 (2018). <https://doi.org/10.1109/ICRA.2018.8463189>
41. Nagasinghe, K.R.Y., Zhou, H., Gunawardhana, M., Min, M.R., Harari, D., Khan, M.H.: Why not use your textbook? Knowledge-enhanced procedure planning of instructional videos. In: CVPR. pp. 18816–18826 (2024). <https://doi.org/10.1109/CVPR52733.2024.01780>
42. Niu, Y., Guo, W., Chen, L., Lin, X., Chang, S.: SCHEMA: State changes matter for procedure planning in instructional videos. In: ICLR (2024), <https://openreview.net/pdf?id=abL5LJNZ49>
43. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: DINOv2: Learning robust visual features without supervision. Trans. Mach. Learn. Res. (2024), <https://openreview.net/pdf?id=a68SUt6zFt>
44. Qi, Z., Wang, S., Su, C., Su, L., Huang, Q., Tian, Q.: Self-regulated learning for egocentric video activity anticipation. IEEE Trans. Pattern Anal. Mach. Intell. **45**(6), 6715–6730 (2023). <https://doi.org/10.1109/TPAMI.2021.3059923>
45. Queisner, M., Eisenträger, K.: Surgical planning in virtual reality: A systematic review. J. Med. Imaging **11**(6), 062603–062603 (2024). <https://doi.org/10.1117/1.JMI.11.6.062603>
46. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML. vol. 139, pp. 8748–8763 (2021), <https://proceedings.mlr.press/v139/radford21a>
47. Robinson, J.D., Chuang, C., Sra, S., Jegelka, S.: Contrastive learning with hard negative samples. In: ICLR (2021), <https://openreview.net/pdf?id=CR1XOQ0UTh->
48. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A unified embedding for face recognition and clustering. In: CVPR. pp. 815–823 (2015). <https://doi.org/10.1109/CVPR.2015.7298682>
49. Seminara, L., Moltisanti, D., Furnari, A.: ViterbiPlanNet: Injecting procedural knowledge via differentiable Viterbi for planning in instructional videos. In: CVPR. pp. 7331–7341 (2026). <https://doi.org/10.48550/arXiv.2603.04265>
50. Srinivas, A., Jabri, A., Abbeel, P., Levine, S., Finn, C.: Universal planning networks: Learning generalizable representations for visuomotor control. In: ICML. vol. 80, pp. 4739–4748 (2018), <https://proceedings.mlr.press/v80/srinivas18b/srinivas18b.pdf>
51. Sun, J., Huang, D., Lu, B., Liu, Y., Zhou, B., Garg, A.: PlaTe: Visually-grounded planning with transformers in procedural tasks. IEEE Robot. Autom. Lett. **7**(2), 4924–4930 (2022). <https://doi.org/10.1109/LRA.2022.3150855>
52. Tang, Y., Ding, D., Rao, Y., Zheng, Y., Zhang, D., Zhao, L., Lu, J., Zhou, J.: COIN: A large-scale dataset for comprehensive instructional video analysis. In: CVPR. pp. 1207–1216 (2019). <https://doi.org/10.1109/CVPR.2019.00130>

53. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS. pp. 5998–6008 (2017), https://papers.nips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
54. Wang, A., Lin, K., Du, J., Meng, J., Zheng, W.: Event-guided procedure planning from instructional videos with text supervision. In: ICCV. pp. 13519–13529 (2023). <https://doi.org/10.1109/ICCV51070.2023.01248>
55. Wang, H., Wu, Y., Guo, S., Wang, L.: PDPP: Projected diffusion for procedure planning in instructional videos. In: CVPR. pp. 14836–14845 (2023). <https://doi.org/10.1109/CVPR52729.2023.01425>
56. Wang, J., Bertasius, G., Tran, D., Torresani, L.: Long-short temporal contrastive learning of video transformers. In: CVPR. pp. 13990–14000 (2022). <https://doi.org/10.1109/CVPR52688.2022.01362>
57. Wu, F., Dong, Y., Cheng, Z.Q., Dai, Y., Chen, G., Wang, H., Dai, Q., Hauptmann, A.G.: GoViG: Goal-conditioned visual navigation instruction generation. arXiv:2508.09547 [cs.CV] (2025). <https://doi.org/10.48550/arXiv.2508.09547>
58. Xu, G., Li, X., Chen, Y., Duan, Y., Wu, S., Yu, H., Chiu, C.H., Ni, J., Tang, N., Li, T.J.J., Yuille, A., Jin, W., Shi, Y.: A comprehensive survey of AI agents in healthcare. *J. Biomed. Inform.* **179**, 105045 (2026). <https://doi.org/10.1016/j.jbi.2026.105045>
59. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al.: Qwen3 technical report. arXiv:2505.09388 [cs.CL] (2025). <https://doi.org/10.48550/arXiv.2505.09388>
60. Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., et al.: Qwen2.5 technical report. arXiv:2412.15115 [cs.CL] (2024). <https://doi.org/10.48550/arXiv.2412.15115>
61. Yang, D., Zhao, Z., Liu, Y.: PlanLLM: Video procedure planning with refinable large language models. In: AAAI. pp. 9166–9174 (2025). <https://doi.org/10.1609/aaai.v39i9.32992>
62. Zhao, H., Hadji, I., Dvornik, N., Derpanis, K.G., Wildes, R.P., Jepson, A.D.: P³iv: Probabilistic procedure planning from instructional videos with weak supervision. In: CVPR. pp. 2928–2938 (2022). <https://doi.org/10.1109/CVPR52688.2022.00295>
63. Zhou, G., Pan, H., Lecun, Y., Pinto, L.: DINO-WM: World models on pre-trained visual features enable zero-shot planning. In: ICML. vol. 267, pp. 79115–79135 (2025), <https://proceedings.mlr.press/v267/zhou25t.html>
64. Zhou, Y., Qi, Z., Lin, L., Jing, J., Chai, T., Zhang, B., Wang, S., Zhang, W.: Masked temporal interpolation diffusion for procedure planning in instructional videos. In: ICLR (2025), <https://openreview.net/forum?id=HnpDHiItd2>
65. Zhukov, D., Alayrac, J., Cinbis, R.G., Fouhey, D.F., Laptev, I., Sivic, J.: Cross-task weakly supervised learning from instructional videos. In: CVPR. pp. 3537–3545 (2019). <https://doi.org/10.1109/CVPR.2019.00365>

Contrastive Energy Fields for Inference-Time Procedure Planning in Instructional Videos

Supplementary Material

In this supplement, we provide additional details on our auxiliary action sequence reconstruction (*cf.* Sec. A), further implementation details (*cf.* Sec. B), a summary of the core baseline methods we compare against (*cf.* Sec. C), and qualitative results (*cf.* Sec. D).

A Auxiliary Action Sequence Reconstruction

In addition to the contrastive loss $\mathcal{L}_c$, which only supervises the predicted goal state $\tilde{x}_g$, we employ an auxiliary action sequence reconstruction loss $\mathcal{L}_{\text{aux}}$ to obtain expressive intermediate representations. The initial and goal observations v_s and v_g are encoded by the video encoder into latent states x_s and x_g , while the ground-truth action sequence is encoded by the text encoder into action tokens $a_{1:T}$. The predictor P_θ is then applied in two passes to reconstruct the actions from the predicted intermediate representations. First pass (state prediction): x_s , the action tokens $a_{1:T}$, and learnable mask tokens representing the unobserved intermediate and goal states ($\{x_l^{\text{mask}}\}_{l=1}^{T-1}, x_g^{\text{mask}}$) are fed jointly into P_θ , which produces the predicted latent states $\{\tilde{x}_l\}_{l=1}^{T-1}, \tilde{x}_g$. Second pass (action reconstruction): the initial state token x_s , the predicted intermediate state latents $\{\tilde{x}_l\}_{l=1}^{T-1}$, ground truth goal state token x_g , and learnable mask tokens representing the actions ($\{a_k^{\text{mask}}\}_{k=1}^T$) are fed back into the same predictor (weight-shared) P_θ , which this time predicts the actions $\tilde{a}_{1:T}$. The predicted action tokens are mapped back to the action space, and the auxiliary loss $\mathcal{L}_{\text{aux}}$ penalizes the discrepancy between the reconstructed actions $\tilde{a}_{1:T}$ and the ground-truth actions $a_{1:T}$. The auxiliary action sequence reconstruction pipeline is visualized in Fig. A.1.

B Implementation Details

Architecture. We implement our predictor P_θ as a causal (masked) transformer [53]. We use four transformer blocks, each with six attention heads and a hidden dimension of 384. In total, P_θ contains 8.5M learnable parameters. Into P_θ , we feed the visual features of the initial x_s and goal state x_g , as well as tokenized actions. We obtain visual features using the S3D [36] visual encoder (pre-trained on HowTo100M [37]) and a temporal window of 3-frames surrounding v_s and v_g , respectively. Before being fed into P_θ , x_s and x_g are normalized along the feature dimension before being linearly projected to the hidden dimension of P_θ . To feed textual actions into P_θ , we tokenize and encode actions in natural language using the pre-trained text encoder from CLIP-ViT-B [46], before linearly projecting the resulting tokens to the hidden dimension of P_θ . CLIP and the visual encoder are frozen; only the linear projections are learned. Our task classifier is implemented as a four-layer multilayer perceptron that takes in both x_s and x_g .

Training. We train the predictor’s weights θ using the AdamW optimizer [33]. AdamW uses a weight decay of 1×10^{-3} and a learning rate of 5×10^{-4} . 50 negative

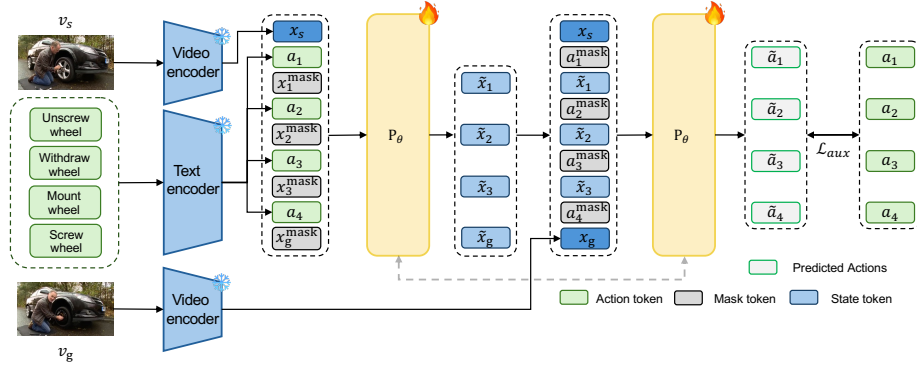


Fig. A.1. Auxiliary action sequence reconstruction. During training, the predictor P_θ is applied twice with shared weights: *first*, given x_s , action tokens $a_{1:4}$, and masked state tokens $x_{1:4}^{\text{mask}}$, P_θ predicts the intermediate and goal latents $\tilde{x}_{1:3}, \tilde{x}_g$, respectively. *Second*, given x_s, x_g , P_θ predicts the masked action tokens. These are supervised by the ground-truth action sequence $a_{1:4}$ using the auxiliary loss $\mathcal{L}_{\text{aux}}$.

samples are generated per positive training sequence. We set the hard/easy negative sample ratio r to 0.8. The minimum $\tau_{\min}$ margin is set to 0.01. The maximum margin $\tau_{\max}$ is set to 0.01. We train P_θ for 200 epochs, using early stopping, following KEPP [41]. The task classifier is trained using a learning rate of 1×10^{-4} with no weight decay.

C Baselines

PDPP [55] approaches procedure planning as a distribution fitting problem under given observations, the initial and goal state. Learned using diffusion, PDPP can sample action-sequence predictions during inference. In particular, PDPP performs conditional diffusion and learns using action-sequence ground truth, and can express uncertainty in its prediction.

KEPP [41] extends PDPP by introducing a probabilistic procedure knowledge graph to the model’s architecture. This knowledge graph is constructed using training data by iteratively adding and reweighing edges and serves as a retrieval signal for diffusion-based inference.

SCHEMA [42] demonstrates that state changes matter for procedure planning, and builds a structured state space by explicitly representing each step as a state change. A large language model (LLM) is used to generate language descriptions of these state changes. Then these descriptions are aligned with visual observations via cross-modal contrastive learning to track intermediate states.

PlanLLM [61] builds on SCHEMA’s use of LLM-generated state descriptions, but instead of decoding into a fixed, closed set of action vectors, it lets the LLM generate

free-form planning output. It adds an LLM-enhanced-planning module for flexible step decoding and a mutual-information-maximization module to link commonsense text with visual state

ViterbiPlanNet [49] approaches procedure planning by explicitly incorporating procedural knowledge into the end-to-end training, different from KEPP. It introduces a differentiable Viterbi layer that embeds a procedural knowledge graph into Viterbi decoding, using smooth relaxations to enable end-to-end training.

D Qualitative Results

In Figs. A.2 & A.3, we present qualitative examples of CEFITO for procedure planning on 8 different tasks of the CrossTask dataset. In each example, the top-ranked candidate plan (*i.e.*, the one with the lowest L2 distance to x_g) matches the ground truth action sequence. Still, all top-5 candidates are plausible plans, but incorrect alternatives which involve step reordering (*e.g.*, “Flip pancake” *vs.* “Take pancake from pan” in “Make pancakes”), action repetition (*e.g.*, “Flip steak” in “Grill steak”), and substitution with semantically related actions (*e.g.*, “Pour water” *vs.* “Pour lemon juice” in “Make lemonade”). The modest margin between the correctly predicted plan and the alternatives suggests that the model can distinguish fine-grained semantics and temporal orderings. We emphasize that these examples are illustrative of correctly predicted plans selected for qualitative analysis.

Failure cases. We show the failure examples of CEFITO on the CrossTask dataset in Figs. A.4 & A.5. In some examples, the correct action sequence is not top-ranked but is still within the top-5 action sequence candidates (*e.g.*, “Grill steak”, “Make pancakes”, “Make meringue”), while in others the models fail to infer the correct action sequence within the top-5 sequence candidates. We identify several failure modes. In many wrong predictions, the model correctly identifies the first and the last action steps and fails to infer the intermediate action steps (*e.g.*, “Make lemonade”, “Make pancakes”, “Make banana ice-cream”). We attribute this to the fact that intermediate visual observation is not available, and the model has to rely solely on start and goal observations. Another observation is that the model wrongly predicts the first action step but successfully infers the subsequent steps (*e.g.*, “Make meringue”).

Task Grill Steak		
	→	
Start		Goal
GROUND TRUTH		Flip steak, Take steak from grill, Cut steak
TOP-5 CANDIDATE PLANS		L2 score ↓
Flip steak, Take steak from grill, Cut steak		1.36
Take steak from grill, Cut steak, Cut steak		1.43
Check temperature, Take steak from grill, Cut steak		1.45
Flip steak, Flip steak, Take steak from grill		1.48
Flip steak, Flip steak, Cut steak		1.50

Task Make Meringue		
	→	
Start		Goal
GROUND TRUTH		Whisk mixture, Add sugar, Put mixture into bag
TOP-5 CANDIDATE PLANS		L2 score ↓
Whisk mixture, Add sugar, Put mixture into bag		1.47
Whisk mixture, Add sugar, Spread mixture		1.54
Add sugar, Put mixture into bag, Spread mixture		1.59
Add sugar, Whisk mixture, Put mixture into bag		1.64
Whisk mixture, Put mixture into bag, Spread mixture		1.68

Task Make French Toast		
	→	
Start		Goal
GROUND TRUTH		Melt butter, Put bread in pan, Flip bread
TOP-5 CANDIDATE PLANS		L2 score ↓
Melt butter, Put bread in pan, Flip bread		1.33
Melt butter, Dip bread in mixture, Put bread in pan		1.45
Whisk mixture, Dip bread in mixture, Put bread in pan		1.68
Melt butter, Dip bread in mixture, Flip bread		1.68
Melt butter, Flip bread, Remove bread from pan		1.69

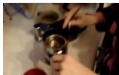
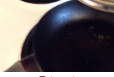
Task Make a Latte		
	→	
Start		Goal
GROUND TRUTH		Add coffee, Press coffee, Pour espresso
TOP-5 CANDIDATE PLANS		L2 score ↓
Add coffee, Press coffee, Pour espresso		1.50
Press coffee, Pour espresso, Steam milk		1.54
Add coffee, Press coffee, Steam milk		1.57
Press coffee, Pour water, Steam milk		1.62
Add coffee, Pour water, Steam milk		1.63

Fig. A.2. Qualitative examples of CEFITO on CrossTask. Procedural video results where the top-ranked candidate plan (highlighted in) matches the ground-truth action sequence, retrieved from the top-5 candidates ranked by L2 distance to the goal embedding x_g .

Task Make Pancakes	
	
Start	Goal
GROUND TRUTH	
Pour mixture into pan, Flip pancake, Take pancake from pan	
TOP-5 CANDIDATE PLANS	
L2 score ↓	
Pour mixture into pan, Flip pancake, Take pancake from pan	1.22
Pour mixture into pan, Flip pancake, Flip pancake	1.33
Pour mixture into pan, Pour mixture into pan, Flip pancake	1.66
Pour mixture into pan, Take pancake from pan, Flip pancake	1.69
Pour mixture into pan, Pour mixture into pan, Take pancake from pan	1.71

Task Make Banana Ice-cream	
	
Start	Goal
GROUND TRUTH	
Cut banana, Put bananas into blender, Mix ingredients	
TOP-5 CANDIDATE PLANS	
L2 score ↓	
Cut banana, Put bananas into blender, Mix ingredients	1.35
Cut banana, Pour milk, Mix ingredients	1.62
Cut banana, Put bananas into blender, Pour milk	1.64
Peel banana, Put bananas into blender, Mix ingredients	1.64
Peel banana, Cut banana, Mix ingredients	1.68

Task Add Oil to Your Car	
	
Start	Goal
GROUND TRUTH	
Close cap, Pull out dipstick, Wipe off dipstick	
TOP-5 CANDIDATE PLANS	
L2 score ↓	
Close cap, Pull out dipstick, Wipe off dipstick	1.35
Remove cap, Put funnel, Pour oil	1.51
Remove cap, Pour oil, Pull out dipstick	1.53
Close cap, Pour oil, Pull out dipstick	1.53
Remove cap, Pour oil, Close cap	1.59

Task Make Lemonade	
	
Start	Goal
GROUND TRUTH	
Cut lemon, Squeeze lemon, Pour lemon juice	
TOP-5 CANDIDATE PLANS	
L2 score ↓	
Cut lemon, Squeeze lemon, Pour lemon juice	1.32
Cut lemon, Add sugar, Pour water	1.46
Cut lemon, Add sugar, Pour water	1.50
Cut lemon, Squeeze lemon, Pour water	1.51
Cut lemon, Pour lemon juice, Pour water	1.62

Fig. A.3. Qualitative examples of CEFITO on CrossTask. Extension of Fig. A.2.

Task Grill Steak		
	→	
Start	Goal	
GROUND TRUTH		Put steak on grill, Close lid, Flip steak
TOP-5 CANDIDATE PLANS		L2 score ↓
Put steak on grill, Close lid, Open lid		1.37
Put steak on grill, Flip steak, Flip steak		1.41
Put steak on grill, Flip steak, Take steak from grill		1.47
Put steak on grill, Close lid, Flip steak		1.50
Put steak on grill, Open lid, Flip steak		1.51
Task Make Meringue		
	→	
Start	Goal	
GROUND TRUTH		Whisk mixture, Put mixture into bag, Spread mixture
TOP-5 CANDIDATE PLANS		L2 score ↓
Add sugar, Put mixture into bag, Spread mixture		1.31
Whisk mixture, Put mixture into bag, Spread mixture		1.43
Whisk mixture, Add sugar, Spread mixture		1.51
Add sugar, Whisk mixture, Spread mixture		1.68
Whisk mixture, Add sugar, Put mixture into bag		1.78
Task Make French Toast		
	→	
Start	Goal	
GROUND TRUTH		Flip bread, Remove bread from pan, Top toast
TOP-5 CANDIDATE PLANS		L2 score ↓
Put bread in pan, Flip bread, Remove bread from pan		1.39
Put bread in pan, Flip bread, Top toast		1.44
Put bread in pan, Remove bread from pan, Top toast		1.51
Dip bread in mixture, Flip bread, Remove bread from pan		1.55
Flip bread, Remove bread from pan, Top toast		1.57
Task Make Lemonade		
	→	
Start	Goal	
GROUND TRUTH		Squeeze lemon, Cut lemon, Add sugar
TOP-5 CANDIDATE PLANS		L2 score ↓
Squeeze lemon, Pour lemon juice, Add sugar		1.46
Squeeze lemon, Pour water, Add sugar		1.53
Squeeze lemon, Add sugar, Stir mixture		1.56
Squeeze lemon, Pour lemon juice, Stir mixture		1.58
Squeeze lemon, Pour lemon juice, Pour lemon juice		1.63

Fig. A.4. Failure cases of CEFITO on CrossTask. Procedural video results where the top-ranked candidate plan (highlighted in) fails to match the ground-truth action sequence (highlighted in). For some examples, the ground truth falls outside of the top-5 candidates.

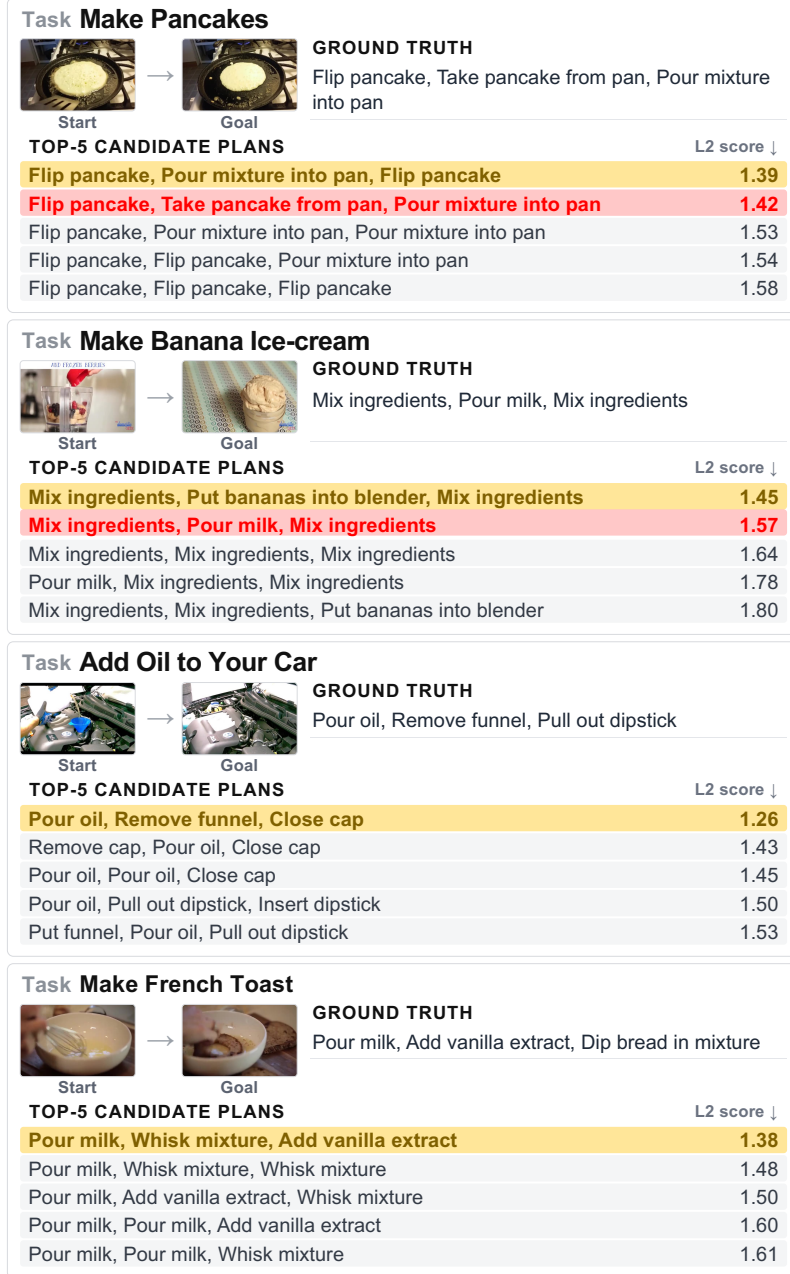


Fig. A.5. Failure cases of CEFITO on CrossTask. Extension of Fig. A.4.