P^{3}IV: Probabilistic Procedure Planning from Instructional Videos with Weak Supervision

He Zhao[1,2] Isma Hadji[1] Nikita Dvornik[1] Konstantinos G. Derpanis[1,2]
Richard P. Wildes[1,2] Allan D. Jepson[1]
[1]Samsung AI Centre Toronto, [2]York University
{zhufl, kosta, wildes}@eecs.yorku.ca, {isma.hadji, n.dvornik, allan.jepson}@samsung.com

Abstract

In this paper, we study the problem of procedure planning in instructional videos. Here, an agent must produce a plausible sequence of actions that can transform the environment from a given start to a desired goal state. When learning procedure planning from instructional videos, most recent work leverages intermediate visual observations as supervision, which requires expensive annotation efforts to localize precisely all the instructional steps in training videos. In contrast, we remove the need for expensive temporal video annotations and propose a weakly supervised approach by learning from natural language instructions. Our model is based on a transformer equipped with a memory module, which maps the start and goal observations to a sequence of plausible actions. Furthermore, we augment our model with a probabilistic generative module to capture the uncertainty inherent to procedure planning, an aspect largely overlooked by previous work. We evaluate our model on three datasets and show our weakly-supervised approach outperforms previous fully supervised state-of-the-art models on multiple metrics.

1. Introduction

Procedure planning is a natural task for humans – one must plan out a sequence of actions that takes one from the current state to the desired goal. While effortless for humans, procedure planning is notoriously hard for artificial agents. Nevertheless, solving procedure planning is of great importance for building next-level artificial intelligence systems capable of analyzing and mimicking human behaviour, and eventually assisting humans in goal-directed problem solving, *e.g.,* cooking, assembling furniture or tasks that can be represented as a clear set of instructions. Traditionally, procedure planning has been addressed in structured environments, such as object manipulation on a table surface [13, 43]. While restricting the environment helps improve planning, it also limits the range of possible applications. Here, we follow more recent work [8] and tackle procedure planning in the realm of instructional videos [46,54]. Given visual observations of the start and goal states, the task is to predict a sequence of high-level actions needed to achieve a goal; see Fig 1. This task is particularly challenging as it requires parsing unstructured environments, recognizing human activities and understanding human-object interactions. Yet, the range of applications for such planners is broad, which motivates research efforts on this problem.

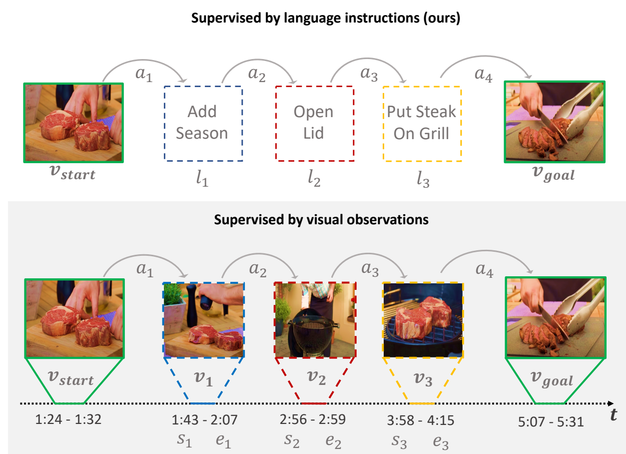

Figure 1. Illustration of weak language supervision for procedure planning. Fully supervised approaches (bottom row) learn models from step labels, a_i, and intermediate visual representations, v_i, over T finite steps. This strategy requires knowing the starting, s_i, and ending, e_i, timestamps, for each intermediate step. In contrast, our approach (top row) exploits natural language representations, l_i, of the intermediate labels, a_i, as a surrogate supervision, which only requires labeling the order of events. Note that the action label, a_i, is a discrete variable, whereas the action language representation, l_i, is a pre-trained continuous embedding.

Current approaches for procedure planning from instructional videos share a serious limitation – reliance on strong supervision with expensive annotations [6, 8, 45]. Specifically, all such methods require access to (i) a list of action labels used to transition from start to the goal state and (ii) the visual representation of the intermediate states. Using such intermediate visual representations entails very expensive annotation of the start and end times of each intermediate instructional step; see Fig 1 (bottom). In contrast, our work removes the need for intermediate visual states during training and instead uses their linguistic representation for supervision. Relying on language representations allows us to better leverage instructional videos and significantly reduce the labeling effort; see Fig. 1 (top). For example, language annotations for the intermediate instructional steps could be extracted from general procedure descriptions available in recipes or websites, *e.g.,* WikiHow [21]. In contrast, to obtain the timestamps for intermediate instructional steps one must watch the entire instructional video. In addition, a language representation can be a more stable supervisory signal [40], as the description of a given step (*e.g*., *add seasoning*) remains the same, while its visual observation varies across different videos.

Previous work on procedure planning from video relies on a two-branch autoregressive approach while adopting different architectures to model these branches [6, 8, 45]. In such models, one branch is dedicated to predicting actions based on the previous observation, while the other approximates the observation given the previous action in a step-by-step manner. Such models are cumbersome and compound errors, especially for longer sequences. In contrast, we rely on a single branch non-autoregressive model, implemented as a transformer [47] that generates all intermediate steps in parallel conditioned on the start and goal observations.

Another important factor in procedure planning is to model the uncertainty inherent to the prediction task. For example, given a set of ingredients and the goal of making a pancake, the intermediate steps could be either (i) [*add wet ingredients* $\rightarrow$ *add dry ingredients* $\rightarrow$ *whisk mixture*] or (ii) [*add dry ingredients* $\rightarrow$ *add wet ingredients* $\rightarrow$ *whisk mixture*]. This example shows that in realistic scenarios some plans can vary even under a shared common goal. This observation is usually handled in physical path planning tasks (*e.g*., robotic arms are allowed to follow multiple feasible trajectories [43]); yet, effort is lacking on probabilistic modeling of procedure planning from instructional videos. While previous work included a probabilistic component at training time [6], we are the first to use and benefit from multiple plausible plans at inference. We explicitly handle uncertainty in procedure planning with a dedicated generative module that can produce multiple feasible plans.

Contributions. In summary, the main technical contributions of our work are threefold. (i) We introduce a weakly supervised approach for procedure planning, which leverages powerful language representations extracted from pre-trained text-video embeddings. (ii) We tackle the task with a simpler single branch model, which can generate all intermediate steps in parallel, rather than relying on the two-branch auto-regressive approach used in previous work. (iii) We propose a generative adversarial framework, trained with an extra adversarial objective, to capture the stochastic property of planned procedures. We evaluate our approach on three widely used instructional videos datasets and show state-of-the-art performance across different prediction time horizons, even while relying on weaker supervision. We also show the advantage of modeling uncertainty. Our code is available at: https://github.com/SamsungLabs/procedure-planning.

2. Related work

Procedure planning. Traditionally, goal-conditioned planning has been studied mostly in physical environments, *e.g*., robotic motion planning [14, 16, 23] and human pedestrian trajectory planning [33]. Recently, the task of procedure planning from instructional videos was introduced [8]. Various approaches have made use of recurrent neural networks (RNNs) [8], transformers [45] and adversarial policy planning [6]; all have used two-branches and strong visual supervision. In contrast, we model the actions directly using a non-autoregressive transformer-based architecture. More importantly, we use low-budget weak supervision in the form of language instructions instead of supervising the model with "costly" visual observations as done by all existing approaches.

Supervision with natural language. A common alternative to training visual models using manually defined label sets is to exploit semantic supervision from natural language. Using natural language as supervision has several advantages: (i) language annotations can be collected automatically [37]; (ii) modeling language and vision jointly can produce stronger representations [10]; (iii) such supervision can achieve better generalization to unseen domains [40]. Such benefits resulted in a growing interest in using language as supervision for variety of tasks, *e.g*., image classification [10, 17, 22], representation learning [36, 50], video retrieval [15, 37], step localization [11, 36] and navigation with instruction following [38]. We use advances in joint video and language modeling for procedure planning. We use pre-trained features [36] to map language and video to a common space and replace expensive video supervision with readily available language instructions

Sequence modeling with transformers. Procedure planning is a task of conditional sequence prediction and thus it directly benefits from recent advances in sequence modeling. One of the strongest recent approaches to sequence modeling is the transformer architecture [47], which has been adopted for a wide variety of tasks, *e.g*., images [26], videos [4] and multi-modal data [31, 32, 50] tasks. Recent work adopted the transformer decoder architecture for fixed-size set prediction via learnable input queries [7, 52]. We build upon similar ideas by setting the first and last queries to correspond to the start and goal observations, while making the intermediate queries learnable. To improve long-range sequence modeling and help overall sequence coherence, recent work augmented transformers with an explicit external memory [29, 49]. For the same reasons, we also integrate a learnable memory module.

Future prediction. The task of procedure planning is closely related to future prediction, where only past observations are provided as input. A key consideration in future prediction is modeling prediction uncertainty from partial initial observations. One common approach for modeling uncertainty is a Variational Auto Encoder (VAE) [25] that captures the distribution over future actions. Another approach is to use generative adversarial networks (GANs) to forecast multiple, distinct and high quality future activities [39,53]. In this work, we adopt the generative modeling framework to model distributions over possible plans.

3. Technical approach

Here, we present our approach to procedure planning that relies on three main components. First, we predict all steps in the plan in parallel using a non-autoregressive transformer decoder. To obtain coherent plan predictions, our transformer is augmented with a learned memory shared across all possible tasks in a given dataset (Sec. 3.2). Second, to model the uncertainty inherent to the task, we include a generative component trained with an adversarial loss. As a result, we can infer multiple feasible plans conditioned on start and goal observations (Sec. 3.3). Third, to supervise the transformer's outputs, we use the cross-entropy loss on action predictions and a contrastive loss to match visual state predictions with the corresponding language descriptions (Sec. 3.4). Fig. 2 provides an overview of our approach, which we detail next.

3.1. Problem formulation

Given a start visual observation, v_{start}, and a desired visual goal, v_{goal}, our task is to predict a *plan* defined as the sequence of T intermediate action steps, $\tilde{\pi} = \tilde{a}_{1:T}$, taken to transition from v_{start} to v_{goal}. We overscore with $\sim$ to indicate our predictions, while the lack of such overscoring indicates ground truth (GT). At training time, given v_{start} and v_{goal}, we predict a plan, $\tilde{\pi}$, and corresponding visual observations, $\tilde{v}_{1:T}$. We use the intermediate action labels, $a_{1:T}$, to train the plan prediction, $\tilde{\pi}$, and corresponding language descriptions (embedded with a pre-trained text encoder), $l_{1:T}$, to supervise the intermediate visual observations, $\tilde{v}_{1:T}$. That is, we substitute the visual information about intermediate instruction steps, $v_{1:T}$, with their language counterparts, $l_{1:T}$, to train the planner; see Fig. 2. We believe such supervision substitution is meaningful, as we are using a strong pre-trained vision-language encoder [36], mapping visual activities and their descriptions in a common space, thus making visual, v_t, and language, l_t, features, corresponding to the same activity, interchangeable for training. In contrast, previous work assumes access to the set of intermediate action-observation pairs (*i.e.*, $a_{1:T}, v_{1:T}$) [6, 8, 45], thereby requiring strong supervision to identify all intermediate visual observations. At inference time, we only use the start and goal observation, to predict a plan, $\tilde{\pi} = \tilde{a}_{1:T}$, for a given time horizon, T.

3.2. Memory augmented transformer decoder

To implement our planner, we use a non-autoregressive transformer decoder architecture [7, 52]. Our transformer decoder takes two input types; namely, learnable queries augmented with the start and goal observations and a learned memory component, and outputs action and intermediate state predictions, as illustrated in Fig. 2.

Conditioned learned-query input. The first input is the query set, $\mathbf{Q} = [q_{start}, q_1, \dots, q_{T-1}, q_{goal}]$, where the first and last inputs correspond to the representations of our initial and goal visual observations, v_{start} and v_{goal}, respectively, while $q_{1:T-1}$ are a set of learned queries. Queries, $[q_1, \dots, q_{goal}]$, are associated with the action labels, $a_{1:T}$, that we wish to predict. To communicate information about the order of elements to the decoder, we add to each query a fixed cosine positional embedding [5], p_t, as follows

$$\mathbf{Q} = [q_{start} + p_0, \dots, q_t + p_t, \dots, q_{goal} + p_T], \qquad (1)$$

where q_t and p_t all are encoded as d dimensional embeddings, (*i.e.*, $q_t, p_t \in \mathbb{R}^d$) and $t = 1, \dots, T-1$.

Learned memory input. The second input to our transformer decoder is a learned memory component that is common across all examples in a given dataset. The memory is defined as a set of d-dimensional vectors

$$\mathbf{M} = [m_1, m_2, \dots, m_n] \in \mathbb{R}^{d \times n}, \qquad (2)$$

where n is the number of learnable vectors in the memory bank. Notably, the size of the memory (*i.e.*, number of entries, n, in the memory) is a hyperparameter that is independent from the prediction time horizon. We use read-only memory [34] and share it among all layers for simplicity.

Memory-augmented transformer decoder. Our architecture is a stack of standard transformer decoder blocks [47] (see Fig. 3), where each such block has access to the global learnable memory, (2). Specifically, the memory-augmented transformer block consists of two key operations. First, the input is processed with the self-attention operation. Second, the cross-attention module attends to the learnable memory to generate the output. The input to

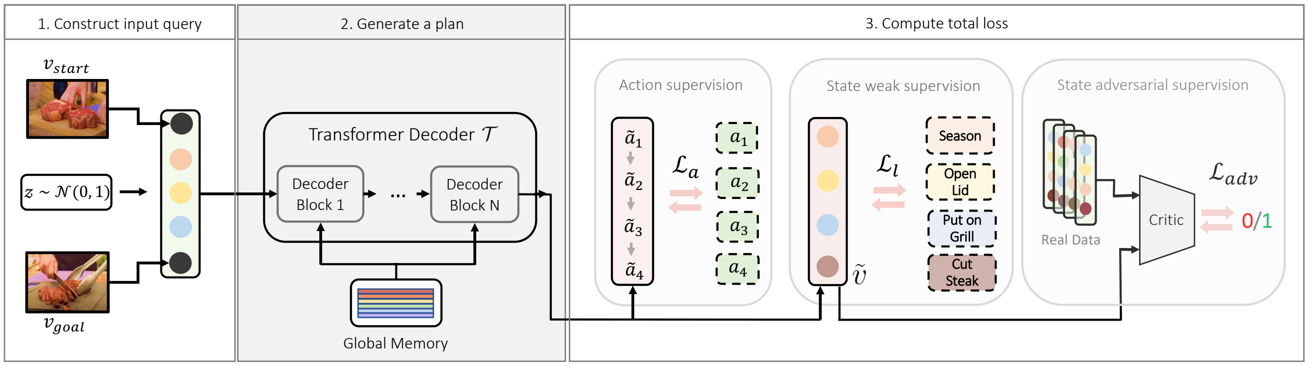

Figure 2. Overview of our procedure planning approach. First, we embed the visual observations of the start and goal states (*i.e.*, black nodes), attach them to the sequence of learned queries (*i.e.*, colored nodes) and add random noise to the resulting input sequence. Second, we pass the input to the transformer decoder that interacts with the global memory to generate feasible procedure plans. Third, we produce state and action vectors, $\{\tilde{v}_i, \tilde{a}_i\}$, and use a number of losses, $\mathcal{L}$ (described in Sec 3.4), to supervise our architecture.

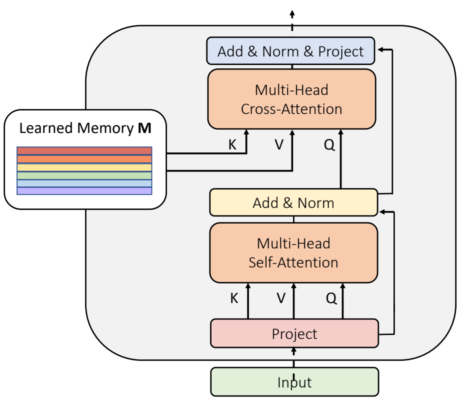

Figure 3. Illustration of the transformer block with self-attention and cross-attention with memory. The learned memory, $\mathbf{M}$, serves as an external memory bank and is globally shared across all transformer blocks. K, V and Q symbolize key, value and query, resp.

self-attention in the first transformer block corresponds to the query, $\mathbf{Q}$. All cross-attention blocks access the same memory, $\mathbf{M}$. Intuitively, the memory module can be seen as a collection of learnable plan embeddings shared across the entire dataset. Empirically, we show that the memory module plays a key role in our framework.

Our transformer decoder, $\mathcal{T}$, consists of a stack of N such memory-augmented blocks. We add two output heads (implemented as multi-layer perceptrons (MLPs)) at the final decoding layer. One head, h_a, yields intermediate predicted actions, $\tilde{a}_t$, while the second, h_v, yields corresponding intermediate visual representations, $\tilde{v}_t$, according to

$$\tilde{a}_{1:T} = h_a(\mathcal{T}(\mathbf{Q}, \mathbf{M})), \quad \tilde{v}_{1:T} = h_v(\mathcal{T}(\mathbf{Q}, \mathbf{M})), \qquad (3)$$

3.3. Adversarial generative modeling

To capture the uncertainty in prediction, where multiple plans from v_{start} to v_{goal} are plausible, we augment our model with a stochastic component using generative adversarial learning [18]. To make the generation process conditional on the input, we augment the entire query input, (1), with a random noise vector, $z \sim \mathcal{N}(0, 1), z \in \mathbb{R}^{d'}$, through concatenation. The new query input sequence to our transformer, $\mathcal{T}$, thus becomes

$$\mathbf{Q}^z = \{[q_t; z] \mid q_t \in \mathbf{Q}\}. \qquad (4)$$

We employ adversarial training wherein the generator, G, is trained to produce realistic action sequences, while the critic, C, provides the supervisory signal for training G [3]. In our case, we treat the memory-augmented transformer, $\mathcal{T}$, as the generator, G (*i.e.*, $G = h_v(\mathcal{T}(\mathbf{Q}^z, \mathbf{M}))$), while the critic is modeled by a simple MLP. More precisely, we pass the output of our transformer, $\tilde{v}_{1:T}$, concatenated along the temporal dimension, to the critic, C, which outputs a value between 0 and 1, indicating its ability to discriminate between the predicted and ground truth sequence, as depicted in Fig. 2. Notably, to avoid the notorious issue of mode collapse associated with training GANs [44] (*i.e.*, regardless of variations in random latent noise, z), we follow previous work [30, 51, 53] and include the normalized distance regularizing loss, $\mathcal{L}_{reg}$, defined in the supplement.

3.4. Training

To supervise our transformer, we rely on two complementary loss functions that enforce our transformer to decode the correct set of action labels in the procedure as well as corresponding visual representations. We also use an adversarial loss to train the stochastic component of our model.

Visual step supervision. One of the outputs of our model at training time is the sequence of visual features, corresponding to the procedure steps, $\tilde{v}_{1:T}$. To supervise the visual features with corresponding language features, $l_{1:T}$, we adopt contrastive learning [19]. For each feature, $\tilde{v}_t$, predicted by the transformer's head, h_v, we use the corresponding ground truth language embedding, l_t, as the positive example and all the other embeddings in the language vocabulary, $\{l_j\}$, as negative examples. The contrastive loss is calculated as

$$\mathcal{L}_l = - \sum_{t=1}^{T} \left[\log \frac{\exp(l_t \cdot \tilde{v}_t)}{\sum_j \exp(l_j \cdot \tilde{v}_t)}\right], \tag{5}$$

where $(\cdot)$ denotes the dot-product operator. Note, we use all examples in the language vocabulary as negatives as our vocabulary is typically small ($<$ 1K elements) and doing so allows for better training compared to per-batch negative sampling, *e.g.*, [20].

Action plan supervision. We also enforce the action prediction head, h_a, to output sequences of action probabilities, $\tilde{a}_t$, corresponding to ground truth one-hot labels, a_t. For this purpose, we use the cross-entropy loss

$$\mathcal{L}_a = - \sum_{t=1}^{T} a_t \log \tilde{a}_t. \tag{6}$$

Adversarial supervision. To model uncertainty, we use adversarial training on the visual state predictions, $\tilde{v}_{1:T}$. The goal is to make predicted visual observation sequences indistinguishable from feature sequences composed of the ground truth language step description, $l_{1:T}$. We optimize the generator, G, (our transformer) and the critic, C, (an MLP) using an adversarial loss [3]

$$\mathcal{L}_{adv} = \min_G \max_C \mathcal{V}(G, C, \mathbf{Q}^z, \mathbf{M}), \tag{7}$$

where $\mathcal{V}$ is the standard GAN objective, defined as $\mathbb{E}_{l \sim p_{data}}[\log C(l)] + E_{z \sim p_z}[\log(1 - C(h_v(\mathcal{T}(\mathbf{Q}^z, \mathbf{M}))))]$, with $l \sim p_{data}$ and $z \sim p_z$ denoting the data distributions of the language representation and random noise, resp.

Complete loss. Overall, our full loss function is defined as

$$\mathcal{L}(\theta) = \lambda_1 \mathcal{L}_l + \lambda_2 \mathcal{L}_{\mathrm{a}} + \lambda_3 \mathcal{L}_{adv} + \lambda_4 \mathcal{L}_{reg}, \tag{8}$$

where θ refers to the parameters associated with all learnable modules, *i.e.*, queries, memory module, as well as the transformer decoder and discriminator parameters and $\lambda_{1:4}$ are empirically determined loss weights.

3.5. Inference

At inference time, we use our transformer as a generative model to sample multiple procedure plans, $\tilde{\pi}^k = \tilde{a}^k_{1:T}$, for the same input start and goal observations. This operation is achieved by drawing $\mathcal{K}$ latent noise vectors, z^k, and forwarding them through our transformer, $\mathcal{T}$, conditioned on a single start-goal observation, as follows

$$\tilde{\pi}^k = h_a(\mathcal{T}(\mathbf{Q}^{z^k}, \mathbf{M})),\ z^k \sim \mathcal{N}(0, 1), \tag{9}$$

for $k = 1, \ldots, \mathcal{K}$.

To obtain a probability distribution over actions at each timestep, t, of the plan, given by our model, we calculate action frequencies as follows:

$$\bar{\Pi} = \bar{a}_{1:T} = \frac{1}{\mathcal{K}} \sum_{k=1}^{\mathcal{K}} [\tilde{a}^k_1, \ldots, \tilde{a}^k_T]. \tag{10}$$

Given that a^k_t are one-hot vectors, each $\bar{a}_t$ results in a marginal distribution over actions at a specific timestep, t.

Most standard benchmark metrics for procedure planning, such as Success Rate, Accuracy or Intersection over Union (IoU), require a single action sequence output for evaluation (see Section 4.1). To compute the most probable action sequence, induced by our action distribution, $\bar{\Pi}$, we use the *Viterbi* algorithm [48], as commonly seen in sequential labeling work [27,28,41,42]. More specifically, we use $\bar{\Pi}$ as the emission matrix in the *Viterbi* formulation, while the transition matrix is estimated from action co-occurrence frequencies in the training set (details in the supplement). Our Viterbi post-processing step can be viewed as biasing sample selections from $\{\tilde{\pi}^k\}^{\mathcal{K}}_{k=1}$, toward plans that are more likely under a first-order model of action transitions. An alternative approach to select a likely action sequence is simply to select the mode from the set $\{\tilde{\pi}^k\}^{\mathcal{K}}_{k=1}$. We empirically demonstrate the superiority of the Viterbi approach, which proved especially useful for smaller datasets.

3.6. Implementation details

Our planner operates on video and language features, pre-extracted by a model trained for joint video-text embedding [36] using the HowTo100M [37] dataset and self supervision. We use a memory-augmented transformer with two layers and eight heads, and optimize it for 200 epochs with ADAM [24] on a single V100 GPU. Additional training and architecture details are provided in the supplement.

4. Experiments

In this section, we evaluate the role of each module in our approach (Sec. 4.2) and demonstrate its performance across three different datasets. We include evaluation on the largest labeled instructional video dataset, which has not been used previously for the task of procedure planning due to the need of strong supervision in previous work (Sec. 4.3). Finally, we provide prediction uncertainty evaluation in procedure planning for the first time, which sheds light on our approach and the task of planning itself (Sec. 4.4).

4.1. Evaluation protocol

Datasets. For evaluation, we use three different instructional video datasets, namely, CrossTask [54], the Narrated Instructional Videos datasets [2] (NIV) and COIN [46]. CrossTask contains 2750 videos, depicting 18 different procedure and an average of 7.6 actions/video; the NIV dataset

Datasets	Loss Objective	SR ↑	mAcc ↑	mIoU ↑
CrossTask	L_a	16.90	44.20	57.56
	L_a + L_l	22.12	45.57	67.40
	L_a + L_l + L_{adv}	**23.34**	**49.96**	**73.89**
	w/o *Viterbi*	22.66	45.95	67.52
COIN	L_a	8.48	12.19	68.15
	L_a + L_l	14.41	20.25	73.49
	L_a + L_l + L_{adv}	**15.40**	**21.67**	**76.31**
	w/o *Viterbi*	14.18	21.01	75.62
NIV	L_a	17.81	42.35	69.42
	L_a + L_l	24.05	46.67	73.89
	L_a + L_l + L_{adv}	**24.68**	**49.01**	**74.29**
	w/o Viterbi	20.18	47.73	73.09

Table 1. Performance of our model trained with different loss functions on three datasets. The last row of each block represents the results of the total loss but without the *Viterbi* algorithm.

is much smaller with 150 videos, five procedures and 9.5 actions/video on average. COIN is the largest dataset in our evaluation. It contains 11827 videos, 778 procedures and 3.6 actions/video. Depicted procedures vary widely, *e.g.*, *Make Taco Salad* and *Change Car Tire*. We follow previous work [8] and adopt 70%/30% to create our train/test splits and we use 20% of the training data for validation. We also follow the data pre-processing steps outlined in the original procedure planning paper [8] to select $\{\text{start}, \text{goal}\}$ observations and curate the dataset into plans covering different time horizons. More details are in the supplement.

Metrics. Following previous work [6, 8, 45], we evaluate the performance using three increasingly strict metrics. (i) mean Intersection over Union (mIoU) treats the predicted and ground truth action sequences as sets, and measures the overlap between these sets. mIoU is agnostic to the order of actions and only indicates whether the model captures the correct set of steps needed to complete the plan. (ii) mean Accuracy (mAcc) performs element-wise comparisons between the predicted and ground truth action sequences, thereby considering the order of the actions as well. (iii) Success Rate (SR) considers a plan successful only if it *exactly* matches the ground truth.

We also evaluate the stochastic nature of our model by measuring the following probabilistic metrics: (i) the KullbackLeibler (KL) divergence between our predicted plan distributions and ground truth; (ii) how well the ground truth modes are covered by our results (Mode Recall); (iii) how often our plans correspond to the ground truth mode (Mode Precision). To this end, for each $\{\text{start}, \text{goal}\}$ observation, we draw ($\mathcal{K}$ = 1500) samples from our generative model and explicitly approximate a distribution, as described in Sec. 3.5. For completeness, we also evaluate using more standard probabilistic prediction metrics, including Negative Log Likelihood (NLL) and the cosine distance [9, 35, 53].

Baselines. We compare to all previous approaches to procedure planning from instructional videos [6, 8, 45] as well as other fully supervised planning approaches [1, 12, 43].

Memory Size	SR ↑	mAcc ↑	mIoU ↑
0	7.49	22.76	31.33
64	16.30	43.62	55.66
128	**23.34**	**49.96**	**73.89**
256	20.81	44.61	59.70

Table 2. Ablation study on the impact of external memory sizes for prediction horizon, $T = 3$, with CrossTask. All results are obtained using our transformer, with two layers and eight heads.

4.2. Ablation study

Impact of different loss functions. We evaluate the role of each loss component by gradually introducing each objective. The results in Table 1 show the pivotal role of language-based supervision, as evidenced by increased performance across all metrics, and the complementarity of the three objectives. Notably, improvements from the adversarial loss may seem marginal (*e.g.*, $\sim$ 1% in SR), as the metrics only compare a single prediction to a single ground truth plan. We show its strict superiority to deterministic models for modelling distributions of procedures in Sec. 4.4.

Impact of Viterbi post-processing. Throughout the empirical results, we use the Viterbi algorithm on top of the predicted action probabilities, $\bar{\Pi}$ (10), to produce the optimal plan at inference time. Notably, the Viterbi post-processing is optional, and one may directly use the set $\tilde{\pi}^k$ to produce the final plan by simply selecting the mode across the set. Comparing the last two rows in each block of Table 1 shows the added advantage of using Viterbi to model the optimal action order in procedure plans explicitly, for all datasets. Notably, Viterbi post-processing is especially helpful on the NIV dataset. We hypothesize that the data scarcity in NIV results in a weaker predictive model at training; therefore, explicitly modeling the optimal transition between the actions with Viterbi plays a more important role in this case.

Impact of model configuration. We also include an ablation evaluating the adopted memory augmented transformer decoder. Table 2 shows that the size of our memory plays a key role in our architecture. Indeed, excluding the memory component yields the worst results, while too large a memory degrades performance. These results suggest that the memory component helps capture dataset content, where it yields stronger results when the number of memory entries is large enough to properly span the actions present in the entire dataset. Notably, while tuning the memory size for each dataset might yield better results, for simplicity, we elect to use the best setting of CrossTask for all datasets.

4.3. Comparison to alternative approaches

CrossTask (short-horizon). Table 3 compares our weakly-supervised approach to a number of alternatives, including the fully supervised state of the art, across the two prediction horizons typically reported in this task. Our results are

Models	*Supervision*	$T=3$			$T=4$		
		SR ↑	mAcc ↑	mIoU ↑	SR ↑	mAcc ↑	mIoU ↑
Random	-	<0.01	0.94	1.66	<0.01	0.83	1.66
Retrieval-Based	-	8.05	23.30	32.06	3.95	22.22	36.97
WLTDO [12]	-	1.87	21.64	31.70	0.77	17.92	26.43
UAAA [1]	-	2.15	20.21	30.87	0.98	19.86	27.09
UPN [43]	V	2.89	24.39	31.56	1.19	21.59	27.85
DDN [8]	V	12.18	31.29	47.48	5.97	27.10	48.46
Ext-GAILw/o Aug. [6]	V	18.01	43.86	57.16	-	-	-
Ext-GAIL [6]	V	21.27	49.46	61.70	**16.41**	43.05	60.93
Ours	L	**23.34**	**49.96**	**73.89**	13.40	**44.16**	**70.01**

Table 3. Evaluation of procedure planning results on CrossTask for prediction horizon $T \in \{3, 4\}$. The column name *Supervision* denotes the type of state supervision applied in training, with V and L denoting visual and language state representation, resp.

Models	$T=3$	$T=4$	$T=5$	$T=6$
	SR↑	SR↑	SR↑	SR↑
Retrieval-Based	8.05	3.95	2.40	1.10
DDN [8]	12.18	5.97	3.10	1.20
Ours (Protocol 1)	**23.34**	**13.40**	**7.21**	**4.40**
PlaTe [45]	18.5	14.0	10.0	7.5
Ours (Protocol 2)	**24.4**	**15.8**	**11.8**	**8.3**

Table 4. Success Rate evaluation of procedure planning results on CrossTask [54] that extends to longer prediction horizon, T.

consistently better, except for the the success rate (SR) at $T = 4$, where we are the second best approach. The performance improvement in SR is especially striking at short-term horizon, $T = 3$, where we outperform the previous best (*i.e.*, Ext-GAIL [6]) by more than 2%, while using weaker supervision. Notably, Ext-GAIL achieves its level of performance via data augmentation, which allows it to have 30% more training data. In a more similar setup (*i.e.*, when Ext-GAIL does not use data augmentation) the performance gain of our method over "Ext-GAIL w/o Aug" is 5.3%. Importantly, our results are obtained with weaker supervision, which speaks decisively in favor of our approach.

We also notice a larger gain in mIoU compared to previous work, *i.e.*, 73.89% $vs.$ 61.70% for $T = 3$ and 70.01% $vs.$ 60.93% for $T = 4$. This result suggests that our approach is better at capturing feasible action steps than other approaches (*e.g.*, never producing *pour water* when input observations are related to *making a salad*). We hypothesize that this performance is enabled by the language-based contrastive learning, which is more effective at clustering latent representations than its vision counterpart. For example, while some visual observations can look similar (*e.g.*, *pour water* and *add oil*), the distinction between the two is clearer in natural language. Notably, the level of improvements in mIoU and mAcc is typically higher than the gain in SR. We attribute this result to the uncertainty inherent to the task (*i.e.*, multiple feasible plans for the same start and goal observations). We explore this aspect in greater detail in Sec. 4.4. Finally, thanks to the non-autoregressive nature, we are **4x** faster at inference, *e.g.*, **6.75ms** (ours) vs. **27.34ms** (DDN [8]) on CrossTask for $T = 3$.

Horizons	Methods	Sup.	NIV			COIN		
			SR↑	mAcc↑	mIoU↑	SR↑	mAcc↑	mIoU↑
$T=3$	Random	-	2.21	4.07	6.09	<0.01	<0.01	2.47
	Retrieval	-	-	-	-	4.38	17.40	32.06
	DDN [8]	V	18.41	32.54	56.56	13.9	20.19	64.78
	Ext-GAIL [6]	V	22.11	42.20	65.93	-	-	-
	Ours	L	**24.68**	**49.01**	**74.29**	**15.4**	**21.67**	**76.31**
$T=4$	Random	-	1.12	2.73	5.84	<0.01	<0.01	2.32
	Retrieval	-	-	-	-	2.71	14.29	36.97
	DDN [8]	V	15.97	27.09	53.84	11.13	17.71	68.06
	Ext-GAIL [6]	V	19.91	36.31	53.84	-	-	-
	Ours	L	**20.14**	**38.36**	**67.29**	**11.32**	**18.85**	**70.53**

Table 5. Procedure planning results on NIV [2] and COIN [46] for prediction horizon $T \in \{3, 4\}$. The column *Sup.* denotes the type of state supervision applied in training.

CrossTask (long-horizon). We now evaluate our model's ability to predict plans for longer time horizons (*i.e.*, $T \in \{3, \dots, 6\}$). We compare to previous approaches that reported results on such horizons. There are two different protocols for these settings. (i) Protocol 1 [8] and (ii) Protocol 2 [45]; see details in supplement. For a fair comparison, we present our results using both protocols in Table 4 and show that our approach is the most effective on both.

NIV. Following previous work [6], we also evaluate our model on the smaller NIV dataset. The results in Table 5 are consistent with our results on CrossTask. Once again, our approach is the top performer across all metrics. This result suggests our language supervision works for smaller datasets as well.

COIN. To show the capability of our method to scale, we evaluate our method on the largest labeled instructional video dataset (*i.e.*, COIN). As we are the first ones to perform procedure planning on such a large-scale dataset, there is a lack of comparison methods. Therefore, we follow existing work [8] and include three baselines: (i) Random selection; (ii) Retrieval-based; (iii) and a re-implementation of the DDN model [8] , using our video features, described in Sec. 3.6. Table 5 shows that our model equipped with language supervision consistently outperforms the baselines, even strongly-supervised ones.

Metric ↓	Method	$T=3$	$T=4$	$T=5$	$T=6$
KL-Div	Ours - determinstic	2.31	4.47	6.30	8.81
	Ours - probabilistic	**2.11**	**3.50**	**4.26**	**6.89**
NLL	Ours - determinstic	5.13	6.25	6.68	8.49
	Ours - probabilistic	**4.89**	**5.48**	**6.24**	**7.67**

Table 6. Evaluation of the plan distributions produced by our probabilistic approach vs. the deterministic variant.

4.4. Evaluating probabilistic modeling

To evaluate our probabilistic modeling, we compare plan distributions produced by our model, with the ground truth distribution over feasible plans. We focus our evaluation of probabilistic modeling on CrossTask as it is the most suitable dataset in terms of variations in the set of feasible plans, as we demonstrate in the supplement.

Plan distribution modeling. Our approach is probabilistic by design as described in Secs. 3.3 and 3.5. To establish a deterministic baseline, we train our model without the adversarial loss and fix the latent noise vector, $z = 0$, both during training and testing. To build the ground truth distribution over goal-conditioned plans, we retrieve all action sequences of length T in the test set that share the given start and goal state. The plan distribution of our probabilistic model (conditioned on the start and goal observations) is obtained by sampling $\mathcal{K} = 1500$ different action sequences, as discussed in Sec. 3.5. For the deterministic baseline, the model produces only one plan (*i.e.*, $\mathcal{K} = 1$). In all cases, the probability of a plan is defined as its frequency in the obtained sample set. To evaluate the quality of the predicted plans, we measure the (dis-)similarity between plan distributions produced by each model (*i.e.*, ours vs. the deterministic variant) and that of the ground truth using the KL divergence and NLL. Table 6 shows that our probabilistic approach better matches the ground truth plan distribution (*i.e.*, it has lower KL and NLL). These results come about because our model is able to sample multiple *valid* plans with respect to the test set distribution, as opposed to the deterministic model that considers a single feasible plan.

Sample diversity and mode coverage. For given start and goal observations, our approach produces multiple plan hypotheses, using probabilistic sampling described in Sec. 3.5 (as visualized in Figure 4). In this section, we measure the diversity of our samples and their relation to the ground truth plans distribution. To characterize the ground truth distribution, we define ground truth modes as the set of unique action sequences in the test set that share the same start and goal state. To calculate the relation of our samples to the ground truth modes we define two metrics, Mode Recall (ModeRec) and Mode Precision (ModePrec). ModeRec reflects how well the GT modes are covered by our model and is calculated as the average number of GT modes captured by at least one sample from our model. In complement, ModePrec measures how often a sampled plan is feasible according to the test data; it is calculated as the average number of samples that match at least one GT mode. Intuitively, ModeRec and ModeRec measure not only how diverse the samples from our model are, but also how useful this diversity is with respect to the GT. By measuring the average cosine distance among model samples, *i.e.*, CosDist — a metric widely used in the GAN literature — we also show that our samples are diverse agnostic to the data distribution. The results in Table 7 suggest our probabilistic approach can produce both diverse and accurate plans, where it is superior on all metrics.

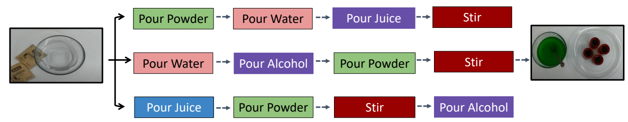

Figure 4. Sample *plausible* plans (seen in the test set) produced by our probabilistic model, for the same {start, goal} observations.

Metric ↑	Method	$T=3$	$T=4$	$T=5$	$T=6$
ModePrec	Ours - deterministic	27.61	17.21	7.41	4.97
	Ours - probabilistic	**36.61**	**18.55**	**12.48**	**6.58**
ModeRec	Ours - deterministic	56.24	37.33	18.38	8.85
	Ours - probabilistic	**66.13**	**46.56**	**26.46**	**12.67**
CosDist	Ours - probabilistic	0.384	0.302	0.2471	0.1658

Table 7. Evaluation of diversity and accuracy of our samples with respect to ground truth. Our approach improves *both* ModePrec and ModeRec metrics. We further provide averaged pair-wise cosine distance as another indicator for diversity. CosDist for the deterministic model is not provided, since it only produces a single result with no pairs to compare.

5. Conclusion

We have introduced a weakly-supervised method for probabilistic procedure planning using instructional videos. Different from previous work, we echew the need for expensive visual supervision in favor of cheaper language supervision by capitalizing on pre-trained text-video embeddings, which, remarkably, leads to superior planning performance. We showed that modeling the interplay between intermediate visual states and actions step-by-step is not a necessity for procedure planning. Instead, we efficiently solve the problem with a "one-shot" transformer decoder architecture. In addition, we demonstrated the crucial role of modeling uncertainty in obtained plans to yield a principled approach to planning from videos. We introduced a way to evaluate such uncertainty on the test set and show that it is a powerful metric to better understand the model and the task of planning itself. Hopefully, future work will adopt the probabilistic view on procedure planning from instructional videos, not only in training but also in evaluation, such that the next generation of planners can confidently predict multiple feasible plans to achieve the desired goal.

References

[1] Yazan Abu Farha and Juergen Gall. Uncertainty-aware anticipation of activities. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 6, 7

[2] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 7

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017. 4, 5

[4] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViVit: A video vision transformer. *arXiv preprint arXiv:2103.15691*, 2021. 2

[5] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 3

[6] Jing Bi, Jiebo Luo, and Chenliang Xu. Procedure planning in instructional videos via contextual modeling and model-based policy learning. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 3, 6, 7

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[8] Chien-Yi Chang, De-An Huang, Danfei Xu, Ehsan Adeli, Li Fei-Fei, and Juan Carlos Niebles. Procedure planning in instructional videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 3, 6, 7

[9] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional GAN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 6

[10] Karan Desai and Justin Johnson. VirTex: Learning visual representations from textual annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[11] Nikita Dvornik, Isma Hadji, Konstantinos Derpanis, Animesh Garg, and Allan Jepson. Drop-DTW: Aligning common signal between sequences while dropping outliers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[12] Kiana Ehsani, Hessam Bagherinezhad, Joseph Redmon, Roozbeh Mottaghi, and Ali Farhadi. Who let the dogs out? Modeling dog behavior from visual data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6, 7

[13] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 1

[14] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International Conference on Machine Learning (ICML)*, 2018. 2

[15] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[16] Dibya Ghosh, Abhishek Gupta, and Sergey Levine. Learning actionable representations with goal-conditioned policies. *arXiv preprint arXiv:1811.07819*, 2018. 2

[17] Lluis Gomez, Yash Patel, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 4

[19] Urs Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010. 4

[20] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, 2019. 5

[21] Jack Herrick. wikiHow. www.wikihow.com, 2005. [Online how-to website]. 2

[22] Armand Joulin, Laurens Van Der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2

[23] Leslie Pack Kaelbling. Hierarchical learning in stochastic domains: Preliminary results. In *International Conference on Machine Learning (ICML)*, 1993. 2

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[25] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Machine Learning (ICML)*, 2014. 3

[26] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[27] Oscar Koller, Hermann Ney, and Richard Bowden. Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[28] Oscar Koller, Sepehr Zargaran, and Hermann Ney. Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent CNN-HMMs. In *Proceedings of the IEEE Conference*

on Computer Vision and Pattern Recognition (CVPR), 2017. 5

[29] Jie Lei, Liwei Wang, Yelong Shen, Dong Yu, Tamara Berg, and Mohit Bansal. MART: Memory-augmented recurrent transformer for coherent video paragraph captioning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2603–2614, 2020. 3

[30] Shaohui Liu, Xiao Zhang, Jianqiao Wangni, and Jianbo Shi. Normalized diversification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4

[31] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[32] Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. UniVL: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*, 2020. 2

[33] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[34] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. MANTRA: Memory augmented networks for multiple trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[35] Nazanin Mehrasa, Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. A variational autoencoder model for stochastic point processes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

[36] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3, 5

[37] 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 *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 2, 5

[38] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 2

[39] AJ Piergiovanni, Anelia Angelova, Alexander Toshev, and Michael S Ryoo. Adversarial generative grammars for human activity prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Krueger Gretchen, and Sutskever Ilya. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021. 2

[41] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with RNN based fine-to-coarse modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 754–763, 2017. 5

[42] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. NeuralNetwork-Viterbi: A framework for weakly supervised video learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[43] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning (ICML)*, 2018. 1, 2, 6, 7

[44] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton. VEEGAN: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 4

[45] Jiankai Sun, De-An Huang, Bo Lu, Yun-Hui Liu, Bolei Zhou, and Animesh Garg. PlaTe: Visually-grounded planning with transformers in procedural tasks. *arXiv preprint arXiv:2109.04869v1*, 2021. 1, 2, 3, 6, 7

[46] 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. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 5, 7

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2, 3

[48] Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967. 5

[49] Qingyang Wu, Zhenzhong Lan, Jing Gu, and Zhou Yu. Memformer: The memory-augmented transformer. *arXiv preprint arXiv:2010.06891*, 2020. 3

[50] Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. VideoCLIP: Contrastive pretraining for zero-shot video-text understanding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021. 2

[51] Dingdong Yang, Seunghoon Hong, Yunseok Jang, Tianchen Zhao, and Honglak Lee. Diversity-sensitive conditional generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2018. 4

[52] Chuhan Zhang, Ankush Gupta, and Andrew Zisserman. Temporal query networks for fine-grained video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[53] He Zhao and Richard P Wildes. On diverse asynchronous activity anticipation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3, 4, 6

[54] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 5, 7