LAP: A LANGUAGE-AWARE PLANNING MODEL FOR PROCEDURE PLANNING IN INSTRUCTIONAL VIDEOS

Anonymous authors
Paper under double-blind review

ABSTRACT

Procedure planning requires a model to predict a sequence of actions that transform a start visual observation into a goal. While most existing methods rely primarily on visual observations as input, they often struggle with the inherent ambiguity where different actions can appear visually similar. In this work, we argue that language descriptions offer a more distinctive representation in the latent space for procedure planning. We introduce Language-Aware Planning (LAP), a novel method that leverages the expressiveness of language to bridge visual observation and planning. LAP uses a VLM to translate visual observations into text embeddings, predicts actions, and extracts corresponding embeddings. Finally, a diffusion model conditioned on the text embeddings of the start and goal actions and augmented text embeddings of intermediated actions generates the plan. We evaluate LAP on three procedure planning benchmarks: CrossTask, Coin, and NIV. LAP achieves new state-of-the-art performance across multiple metrics and time horizons by large margins, demonstrating the significant advantage of language-aware planning.

1 INTRODUCTION

A fundamental goal of future human-AI interaction is to enable AI systems that proactively assist humans in accomplishing complex tasks Huang et al. (2023); Shi et al. (2023); Patel et al. (2023); Kim et al. (2024). Procedure planning in instructional videos represents a critical step toward this vision. The procedure planning task Chang et al. (2020) is to plan the intermediate action, given start observations and the goal observation. This capability is crucial for building AI systems that can assist in real-world tasks Hu et al. (2023); Castelo et al. (2023); Luo et al. (2025).

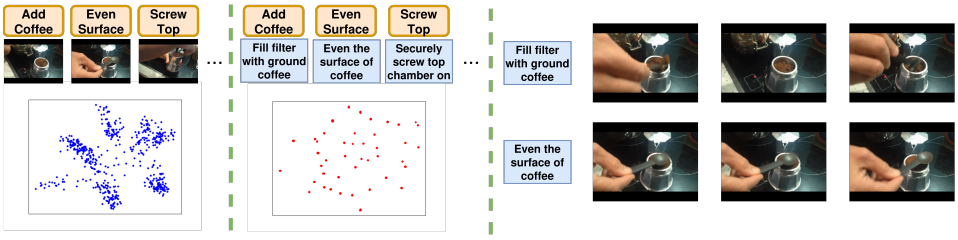


Figure 1: Orange boxes show the labels of actions. Blue boxes show the language descriptions of actions. **Left**: Visualisation of extracted features of visual observations in the latent space. Each point represents one video. **Middle**: Visualisation of text embeddings of actions in the latent space. **Right**: Start visual observation of "Fill filter with ground coffee" and "Even the surface of coffee".

While visual observations provide rich information about the action step, they have, however, a critical limitation: visual observations of two different actions can sometimes look similar. The right part of Figure 1 exemplifies this limitation; the upper row shows the start observation of the action "Add Coffee", while the lower row shows the start observation of the action "Even Surface". Most content in the frames of both actions looks similar (e.g., the background, objects, hand, and so on), even though they are two different actions. This makes it difficult for a model to distinguish actions based on visual observation. On the other hand, using natural languages to describe actions

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

are more distinctive. In the left and middle parts of Figure 1, we show the visual observations and the language descriptions of actions in the latent space. The visual observations are more cluttered than the language descriptions, which makes it easier for models to plan actions and to ground them in the language modality Persson et al. (2019).

Most previous works in procedure planning have primarily focused on learning representations from visual observations Chang et al. (2020); Sun et al. (2022); Li et al. (2023); Wang et al. (2023b); Nagasinghe et al. (2024). Some of the works leveraged text embeddings of action labels indirectly via visual observations, for instance, as language supervision Zhao et al. (2022), or as additional noise in diffusion models Shi et al. (2025). Other works also incorporated Large Language Models (LLMs) to augment descriptions of actions Niu et al. (2024); Yang et al. (2025), or to plan action sequences based only on text Liu et al. (2023). Although these works incorporated language as auxiliary components in their methods, they still use visual observation for the start and goal state, overlooking the potential of using language for the start and goal state, which is the the most critical part of procedure planning.

To this end, we propose **LAP**: A **L**anguage-**A**ware **P**lanning model for procedure planning in instructional videos. We leverage the expressiveness of natural language descriptions and their distinctiveness in the latent space to bridge the visual observation and action planning domains for the task of procedure planning. Specifically, we propose to finetune a pretrained Vision Language Model (VLM) with a regularizer to translate visual observations into language descriptions and predict the start and goal action for planning. In the planning stage, we propose an ensemble planning model which consists of a base planning model and a language-conditioned planning model. Both models are based on diffusion models as they have been shown to be successful with temporal data in various domains Jiao et al. (2024); Yan et al. (2023). We use the denoising diffusion probabilistic model (DDPM) Ho et al. (2020) as the diffusion model. For the base planning model, it generates an intermediate action sequence as the plan. The model conditions the generation process on the text embeddings of the predicted start and goal actions. The language-conditioned planning model further conditions on clustered text embeddings of predicted intermediate actions. The final plans are averaged predictions from base and language-conditioned planning model. We evaluate our proposed LAP on three procedure planning benchmarks: CrossTask Zhukov et al. (2019), Coin Tang et al. (2019), and NIV Alayrac et al. (2016). The experimental results demonstrate that LAP achieves State-Of-The-Art (SOTA) performance on all metrics at all time horizons across all three datasets. We thereby show that text embeddings are more distinctive than visual observations and that procedure planning can further benefit from such embeddings.

Our contributions are threefold. First, we introduce LAP, a novel procedure planning method that uses the expressive and discriminative properties of language descriptions to bridge visual observations and procedure planning. Second, we achieve SOTA performance on three challenging procedure planning datasets (CrossTask, Coin, and NIV) across multiple evaluation metrics and planning horizons by large margins. Third, we empirically demonstrate that procedure planning can benefit from using text embeddings of language descriptions, as they provide a more distinctive representation than visual observations.

2 Related Work

2.1 Procedure Planning in Instructional Videos

Instructional videos provide step-by-step instructions for accomplishing various tasks Miech et al. (2019); Zhukov et al. (2019); Zhou et al. (2018). Procedure planning aims to model the temporal structure of a task by predicting a sequence of actions given the start state and the goal state in an instructional video. Early approaches use Recurrent Neural Networks (RNNs) Chang et al. (2020) or Reinforcement Learning (RL) Bi et al. (2021), as well as Transformers Sun et al. (2022), to model action dependencies. E3P Wang et al. (2023a) proposes an event-guided paradigm where the action sequences are predicted by visual observation and inferred events. SkipPlan Li et al. (2023) decomposes long action sequences into shorter ones and skips less specific actions to improve the sequence prediction.

2.2 Large Language Models in Planning

Another paradigm seeks to incorporate Large Language Models (LLMs) for planning. LLMs have been used widely in text-image-based planning Wu et al. (2021); Lu et al. (2023), where the task is to generate text descriptions and images for future steps. In robotic planning tasks, LLMs are used for task decomposition Singh et al. (2022); Yang et al. (2024), or as a high-level planner Dagan et al. (2023); Guan et al. (2023). For procedure planning, SCHEMA Niu et al. (2024) uses LLMs to generate language descriptions of states and injects the description features into the cross-attention in transformers to decode action sequences. PlanLLM Yang et al. (2025) also adopts the use LLM as in SCHEMA and further uses a finetuned VLM to refine action sequences and generate free-form predictions. LAP also uses LLMs and VLMs in the pipeline, however, the motivation of including LLMs/VLMs and how they are used is fundamentally different from SCHEMA and PlanLLM. LAP uses a VLM to transform the start and goal state from visual domain to text domain to avoid confusion. PlanLLM and SCHEMA still use visual observation for start and goal state, the use of VLM/LLM is for refinement. Moreover, PlanLLM uses visual observation of intermediate actions and ground truth action text of all actions, which gives unfair advantages and makes the contributions of LLM and VLM unclear and unverified.

2.3 Generative Models for Planning

Generative models have demonstrated a strong capacity to model the complex distribution of action sequences. P^{3}IV Zhao et al. (2022) adds adversarial generative modules to transformers for training. During inference, the transformers are used as a generator to sample multiple action sequences for planning. PDPP Wang et al. (2023b) constructs the start and goal observation as well as the predicted task as input and uses Denoising Diffusion Probabilistic Models (DDPMs) to predict the action sequences. ActionDiffusion Shi et al. (2025) further incorporates action embeddings into the noise-adding phase of DDPM to enhance temporal dependencies in actions. KEPP Nagasinghe et al. (2024) explores integrating knowledge graphs learned from full action sequences to guide DDPMs. CLAD Shi & Bulling (2025) uses text embeddings to constrain the latent space of diffusion model to reduce the search space of actions. MTID Zhou et al. (2025) introduces a temporal interpolation module to generate interpolated latent features from start and goal observations. The interpolated latent features are further integrated into the denoising process of a diffusion model to improve the temporal relations.

3 Method

3.1 Problem Definition

The task definition of procedure planning Chang et al. (2020) is, given the start observation o_s and goal observation o_g, a model should generate a plan π that contains the intermediate actions $[a_1 : a_T]$ with time horizon T. Our motivation is to use text information instead of visual observations, as text information is potentially more distinctive in the latent space and could benefit the learning of plan π. This requires transforming visual observation to text information, formally,

$$p(\pi \mid o_s, o_g) = \int p(\pi \mid \hat{T}_s, \hat{T}_g)\, p(\hat{T}_s, \hat{T}_g \mid o_s, o_g)\, d\hat{T}_s d\hat{T}_g, \tag{1}$$

where $\hat{T}_s$ and $\hat{T}_g$ are the text embeddings of predicted start and goal actions. The procedure planning task is essentially split into two steps: (1) obtaining text embeddings of the start and goal visual observations (Section 3.3), and (2) planning π based on text embeddings $\hat{T}_s$ and $\hat{T}_g$ (Section 3.4).

Figure 2 shows the overview of LAP. We first finetune a VLM to predict text descriptions (captions) from visual observations. Next, we transform the visual observations of the start and goal into text embeddings by incorporating the finetuned VLM. The generate text descriptions are then used for predicting the actions and extracting text embeddings of the start and goal. Finally, we deploy a diffusion model to generate the plan π.

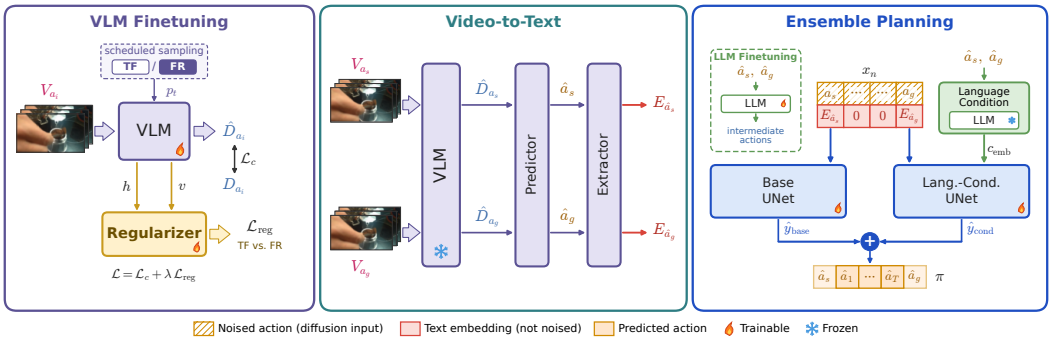


Figure 2: Overview of our proposed LAP. **VLM Finetuning**: We finetune a VLM with a regularizer to predict the language description $\hat{D}_{a_i}$ of an action a_i The ground truth language description D_{a_i} comes from dataset annotations. **Video-to-Text**: we use the finetuned VLM to generate descriptions of start and goal visual observation V_{a_s} and V_{a_g} and predict action $\hat{a}_s$ and $\hat{a}_g$. Then the extractor extracts the corresponding embeddings $E_{\hat{a}_s}$ and $E_{\hat{a}_g}$. **Ensemble Planning**: We use a diffusion model for both base planning model and language-conditioned planning model. In both models, we use the embeddings of start and end embeddings $E_{\hat{a}_s}$ and $E_{\hat{a}_g}$ as well as action sequence to construct the input to the diffusion model. The UNet in base model does not have any condition while the language-conditioned model use embeddings of intermediate actions that are predicted by a finetuned LLM.

3.2 VLM WITH SCHEDULED SAMPLING AND REGULARIZER

We finetune a VLM to predict text descriptions of an input video V_{a_i} using the language description D_{a_i} as supervision. VLM models are conventionally trained with teacher forcing, in which the next token is conditioned on previous ground-truth token at every step. This creates a mismatch between training and inference, where the next token instead conditions on model's own prediction. This induces exposure bias and potentially generates erroneous captions. We use scheduled sampling (Bengio et al., 2015) with a regularizer to train the VLM to overcome this limitation.

Scheduled Sampling. Given a video input V_{a_i}, the VLM outputs tokens $\hat{T}_{a_i}$ through the forward pass. Under standard teacher forcing, the ground-truth previous token is fed as input at every decoding position. With scheduled sampling, the ground-truth previous token is instead replaced by the model's own prediction with a substitution probability p_{step}, and is kept otherwise. We increase p_{step} linearly from 0 to $p_{\max}$.

Regularizer Network. We introduce a multimodal network R_ϕ that uses the decoder's hidden states $h \in \mathbb{R}^{T \times d_h}$ and the visual tokens v, and outputs a scalar logit. R_ϕ encodes each modality with a bidirectional LSTM, fuses them with cross attention and passes the concatenated feature through a MLP classifier. The classifier predicts if the hidden states and visual tokens are from teacher-forcing cycle or free-running cycle. We supervise R_ϕ with the ground truth cycle indicator m_k (teacher-forcing or free-running) via a binary cross-entropy loss,

$$\mathcal{L}_{\text{reg}}(\theta, \phi) = \text{BCE}\big(R_\phi(h, v),\, m_k\big), \tag{2}$$

where θ is the VLM parameter and the total loss $\mathcal{L}(\theta)$ is,

$$\mathcal{L}(\theta) = \mathcal{L}_c(\theta) + \lambda\, \mathcal{L}_{\text{reg}}(\theta, \phi), \tag{3}$$

where $\mathcal{L}_c$ is token-level cross entropy loss λ is a fixed weight.

3.3 VIDEO TO TEXT

After finetuning, we transform the start and goal actions from visual domain to text domain. The VLM generates multiple text descriptions for the start and goal visual observation. We then select the descriptions that can best represent the start and goal action in the video and extract text embeddings for further use in planning.

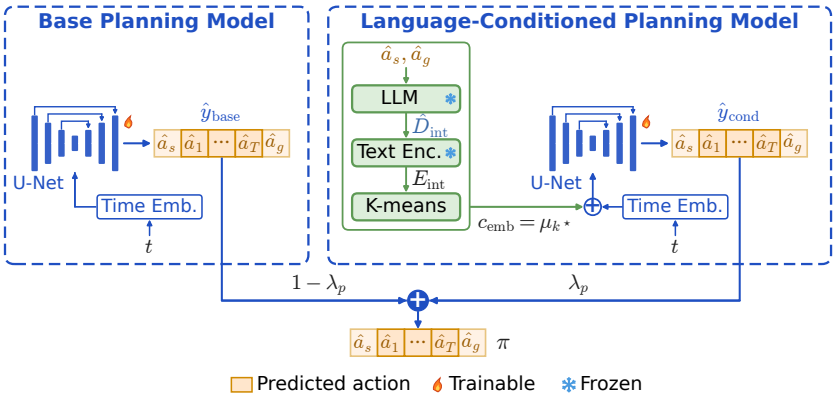


Figure 3: Details of ensemble planning in LAP. Both planning models are DDPMs with a U-Net denoiser and are trained separately. **Base Planning Model:** the U-Net predicts the action sequence $\hat{y}_{\text{base}}$. **Language-Conditioned Planning Model:** given the predicted start and goal actions $\hat{a}_s$ and $\hat{a}_g$, a finetuned LLM generates descriptions of the intermediate actions $\hat{D}_{\text{int}}$. A text encoder embeds these descriptions as E_{int}, which K-means assigns to the nearest cluster centroid $\mu_{k^\star}$. This centroid is used as the conditioning embedding c_{emb}, added to the time embedding, and injected into the U-Net, which predicts $\hat{y}_{\text{cond}}$. **Ensemble:** at inference, the final plan π is obtained from the weighted average $\hat{y} = \lambda_p\, \hat{y}_{\text{cond}} + (1 - \lambda_p)\, \hat{y}_{\text{base}}$.

Predictor: we get the action label of V_{a_i} using $\hat{D}_{a_i}$. We use a similar way as in sentence summarization Jia et al. (2021) to obtain actions from visual observations. After the VLM is finetuned, we generate M numbers of descriptions for each video input V_{a_i}. We calculate ROUGE-1 scores for all M generated descriptions against the ground truth descriptions of actions and compare them with a threshold. If a description in M descriptions whose ROUGE-1 score is higher than or equal to the threshold, we consider it represents the action of V_{a_i}, otherwise we assign a "unknown" label to V_{a_i}. When multiple generated descriptions exceed the threshold, the one with the highest score is selected. The intuition is that the generated descriptions can have the same semantics even though not every word in the description is exactly the same as in the ground truth. We set the threshold to 0.5, representing that if more than half of the words in the generated description exist in the ground truth, we consider that it can represent the visual observation.

Extractor: once we have the predicted action $\hat{a}_i$, we use the text encoder from Miech et al. (2020). The text encoder is pretrained on HowTo100M Miech et al. (2019) dataset. For each $\hat{a}_i$, the paired language description $D_{\hat{a}_i}$ is fed to the text encoder to obtain the text embedding $E_{\hat{a}_i}$ of $\hat{a}_i$.

3.4 Planning

We propose an ensemble planning model to generate plans. Specifically, we train a base planning model and a language-conditioned planning model separately and average the logits of both models during inference.

Base Planning Model We use DDPM Ho et al. (2020) as the base planning model. The input x_0 to the diffusion model is,

$$x_0 = \begin{bmatrix} a_s & \dots & \dots & \dots & a_g \\ E_{\hat{a}_s} & 0 & \dots & 0 & E_{\hat{a}_g} \end{bmatrix}, \tag{4}$$

where a_s and a_g are the start and goal action, respectively, and $E_{\hat{a}_s}$ and $E_{\hat{a}_g}$ are the text embeddings of the start and goal action. Following Wang et al. (2023b), the dimension of text embeddings is not noised. In the forward passing of the diffusion phase, a Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is added to the action dimension iteratively. The denoising phase follows,

$$x_n = \begin{bmatrix} \epsilon & \epsilon & \dots & \epsilon & \epsilon \\ E_{\hat{a}_s} & 0 & \dots & 0 & E_{\hat{a}_g} \end{bmatrix}, \tag{5}$$

where the text embedding dimension is kept unchanged.

Language-Conditioned Planning Model. The base planning model does not using any information from intermediate actions, the language-conditioned planning model also uses DDPM but additionally uses language-guided clustering features as conditions in the diffusion model for planning. More specifically, we finetune an LLM to generate the descriptions of intermediate actions $\hat{D}_{\text{int}}$, given $\hat{a}_s$ and $\hat{a}_g$. Each generated description is embedded with a text encoder and the resulting embeddings are quantized via K-means clustering into K semantic clusters $\{\mu_1, \dots, \mu_K\}$. For a given plan, we assign its generated intermediate description to its nearest centroid,

$$k^\star = \underset{k \in \{1,\dots,K\}}{\arg\min} \left\| E_{\text{int}} - \mu_k \right\|_2, \tag{6}$$

and use the corresponding centroid $\mu_{k^\star}$ as the conditioning embedding c_{emb}. The conditioning embeddings is then added to the time embeddings in the UNet and injected into every residual temporal block throughout the network, so that both the diffusion timestep and the conditioning signal jointly modulate feature extraction at all resolutions.

Ensemble Inference. At inference time, we run the reverse diffusion process independently with each of the two planning models and obtain two predicted action-sequence logits $\hat{y}_{\text{base}}$ and $\hat{y}_{\text{cond}}$. The final prediction is a weighted average of the two,

$$\hat{y} = \lambda_p \, \hat{y}_{\text{cond}} + (1 - \lambda_p) \, \hat{y}_{\text{base}}, \tag{7}$$

where $\lambda_p \in [0, 1]$ is a weight that averages the two branches and combines the predictions of the base model with the semantically informed, cluster-conditioned predictions from the language-conditioned planning model. We use 0.5 for λ_p.

4 Experiments

4.1 Datasets And Data Curation

To evaluate our proposed LAP, we used three procedure planning datasets: CrossTask Zhukov et al. (2019), Coin Tang et al. (2019), and NIV Alayrac et al. (2016). CrossTask dataset has 2,750 videos, 18 tasks, and 105 action classes. The average number of actions in a video is 7.6. Coin dataset has 11,827 videos with 3.6 actions per video on average, 180 different tasks, and 778 action classes. NIV dataset is the smallest dataset among all three and has the least number of videos, tasks, and action classes. It contains 150 videos, five types of tasks, and 18 action classes. The average number of actions in a video is 9.5.

We follow the data curation process used in previous work Chang et al. (2020). For each video in the dataset, we extract action sequences using a sliding window with horizon T. There are two settings for getting the start visual observation o_s and goal visual observation o_g. We refer to the two settings as the PDPP setting Wang et al. (2023b) and the KEPP setting Nagasinghe et al. (2024). In the PDPP setting, the start observation o_s period is from the beginning of the first action to three seconds after the first action. The goal observation o_g begins two seconds before the last action in a sequence and ends one second after the last action. In the KEPP setting, o_s begins one second before the first action and ends two seconds after the first action. The goal observation o_g starts at one second before the last action and finishes at two seconds after the last action. In this paper, we follow the KEPP setting, as most of the recent baselines also use KEPP setting.

4.2 Baselines

We compared our method to SOTA methods for procedure planning on the instructional video datasets: PDPP Wang et al. (2023b), ActionDiffusion Shi et al. (2025), SCHEMA Niu et al. (2024), SkipPlan Li et al. (2023), KEPP Nagasinghe et al. (2024), PlanLLM Yang et al. (2025), and MTID

Zhou et al. (2025). All the baselines use pre-extracted visual features for o_s and o_g, and they are extracted by using a model Miech et al. (2020) pre-trained on the HowTo100M dataset Miech et al. (2019). Our method, on the other hand, needs the original video for training. For the CrossTask dataset, some videos are no longer available from YouTube. Therefore, we remove the unavailable videos from CrossTask for our training. This results in 1870 videos in the new training set and 777 in the new test set, while the original training and test sets have 1921 and 797 videos, respectively.

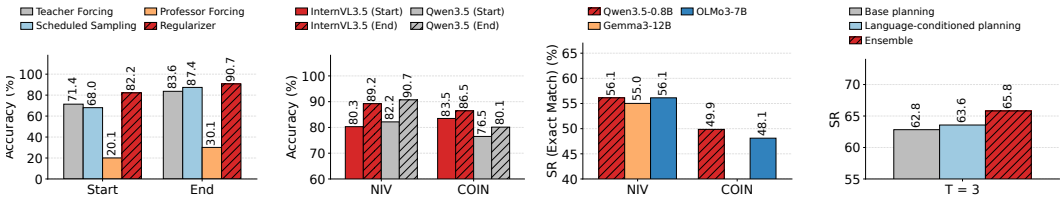
Since the pre-extracted features used by other baselines are extracted from the original CrossTask dataset, and we can only use a subset of videos, we train all baselines on the new dataset splits to ensure a fair comparison. We train all baseline models with the official code provided by the authors, and with data curation using KEPP setting. We also keep all training parameters the same as reported in the original papers.

4.3 Metrics

We adopted Success Rate (SR), mean Accuracy (mAcc), and mean Single Intersection over Union (mSIoU) for evaluation. SR assesses whether a generated action plan is completely successful, i.e. a plan is considered successful only if every action and its sequential order are correct, this makes SR the most challenging metric among all three metrics. mAcc is computed based on the correct actions in the action plan, without considering their order. Similarly, mSIoU measures Intersection over Union (IoU) between the predicted and ground-truth actions and does not account for action ordering. Early works Chang et al. (2020); Zhao et al. (2022); Bi et al. (2021) computed mIoU across all action plans in a mini-batch. However, Wang et al. (2023b) later demonstrated that mIoU depends on the batch size. Consequently, subsequent studies Shi et al. (2025); Zhou et al. (2025) have adopted mSIoU instead.

5 Results

5.1 Design Choices



(a) Finetuning strategy (b) VLM backbone (c) LLM for intermediate action prediction (d) Ensemble planning

Figure 4: Design choices of LAP. (a) Compariosn of training with regularization and teacher forcing, scheduled sampling, and professor forcing on start and goal accuracy. (b) Comparison of VLM backbones on start and goal accuracy. (c) Comparison of LLMs for intermediate action generation. (d) Comparison of Ensemble planning.

We justify the design choices of our method in this section. We start with the training strategy of VLMs. Standard VLM finetuning uses teacher-forcing and may encounter exposure bias, approaches to overcome this limitation including professor-forcing Lamb et al. (2016) and scheduled sampling Bengio et al. (2015). We compare our strategy with the three strategies in Figure 4a. We use QWen3.5-9B Qwen Team (2026) as the VLM backbone and compare the start and goal accuracy on NIV with $T = 3$. Our finetuning strategy outperforms teacher-forcing, scheduled sampling and professor-forcing for start and goal prediction, showing the effectiveness of our strategy. Next, we compare different VLMs, i.e. QWen3.5-9B and InternVL3.5-8B Wang et al. (2025), on start and goal action prediction with $T = 3$ (Figure 4b). QWen3.5-9B is slightly better than InternVL3.5-8B on NIV. We then further compare them on Coin where InternVL3.5-8B outperforms QWen3.5-9B by around 7% accuracy on start and goal. We therefore use InternVL3.5-8B as the VLM backbone. On the LLM in language-conditioned planning, we first compare three LLMs with different number of parameters and from different model families on NIV with $T = 3$ in Figure 4c. We compare

generated text of action with GT text, when all characters match, we consider it is successful. All models perform comparably, Qwen3.5-0.8B and Olmo3-7B are slightly better than Gemma3-12B and both has 56.1 SR. Then we further compare them on Coin and Qwen3.5-0.8B is 1.8 higher than Olmo-7B. Therefore we choose Qwen3.5-0.8B as the LLM in language-conditioned planning. Next we compare base planning model, language-conditioned planning model and ensemble inference. The language-conditioned planning model outperforms base model and ensemble inference further increases SR to 65.8.

5.2 Comparison with SOTA

Table 1: Comparisons with baseline methods on CrossTask dataset with time horizon from $T = 3$ to $T = 6$. ↑ means higher is better. The best results are shown in **bold**. The second best numbers are shown with underline. All baselines and our method use KEPP setting for data curation.

	T=3			T=4			T=5	T=6
	SR↑	mAcc↑	mSIoU↑	SR↑	mAcc↑	mSIoU↑	SR↑	SR↑
PDPP Wang et al. (2023b) (CVPR 2023)	33.9	61.39	/	20.77	56.57	/	12.72	8.07
SkipPlan Li et al. (2023) (ICCV 2023)	27.6	59.9	/	14.06	54.64	/	6.51	5.24
ActionDiffusion Shi et al. (2025) (WACV 2025)	33.25	61.47	65.09	20.01	56.74	63.98	12.48	8.07
KEPP Nagasinghe et al. (2024) (CVPR 2024)	33.27	60.16	/	19.41	54.45	/	12.01	7.89
SCHEMA Niu et al. (2024) (ICLR 2024)	32.34	57.3	/	22.51	51.47	/	12.89	7.54
LAP (Ours)	**54.3**	**81.71**	**77.09**	**31.87**	**73.36**	**72.97**	**18.23**	**12.04**

Table 1, Table 2 and Table 3 compare LAP with state-of-the-art baselines on CrossTask, COIN and NIV, respectively. LAP achieves the best performance on every metric, at every horizon, and on all three datasets. On CrossTask, LAP improves SR at $T = 3$ from 33.90 to 54.30 and mAcc from 61.47 to 81.71. On COIN, it surpasses the strongest baseline by 22.6 SR at $T = 3$ and 12 SR at $T = 4$. The gains are largest on NIV, where LAP roughly doubles the SR of the second-best method at both horizons. The advantage also persists when the planning horizon grows. On CrossTask, LAP improves SR at $T = 6$ by roughly 49% relative to the best baseline, even though errors accumulate over longer action sequences. The results show that LAP substantially raises the level of performance in procedure planning across datasets of varying scale.

Table 2: Comparisons with baseline methods on Coin dataset with time horizon $T = 3$ and $T = 4$. ↑ means higher is better. The best results are shown in **bold**. The second best numbers are shown with underline. All baselines and our method use KEPP setting for data curation. † indicates the model uses intermediate visual observations.

	T=3			T=4		
	SR↑	mAcc ↑	mSIoU↑	SR↑	mAcc ↑	mSIoU↑
PDPP Wang et al. (2023b) (CVPR 2023)	21.33	45.62	51.82	13.67	42.58	/
SkipPlan Li et al. (2023) (ICCV 2023)	23.65	47.12	/	16.04	43.19	/
KEPP Nagasinghe et al. (2024) (CVPR 2024)	20.25	39.87	/	15.63	39.53	/
SCHEMA Niu et al. (2024) (ICLR 2024)	32.09	49.84	/	22.02	45.33	/
PlanLLM † Yang et al. (2025) (AAAI 2025)	33.22	54.33	/	25.31	48.79	/
MTID Zhou et al. (2025) (ICLR 2025)	30.44	51.70	59.74	22.74	49.9	61.25
LAP (Ours)	**55.86**	**78.69**	**77.6**	**37.29**	**70.61**	**75.47**

5.3 Plan with Text vs Plan with Visual Observation

To evaluate the contribution of language-aware procedure planning, we conduct an ablation study comparing language-aware planning with visual observation. We compare LAP with text embeddings and LAP with visual observations (only). For planning based solely on visual observations (denoted as LAP-vo), we use the features extracted from the start and goal observations in the diffusion model. Concretely, Equation 4 becomes,

$$x_0 = \begin{bmatrix} a_s & ... & ... & ... & a_g \\ E_{V_s} & 0 & ... & 0 & E_{V_g} \end{bmatrix}, \tag{8}$$

Table 3: Comparisons with baseline methods on NIV dataset with time horizon $T = 3$ and $T = 4$. ↑ means higher is better. The best results are shown in **bold**. The second best numbers are shown with underline. All baselines and our method use KEPP setting for data curation. † indicates the model uses intermediate visual observations.

	T=3			T=4		
	SR↑	mAcc ↑	mSIoU↑	SR↑	mAcc ↑	mSIoU↑
PDPP Wang et al. (2023b) (CVPR 2023)	22.22	39.50	/	21.30	39.24	/
KEPP Nagasinghe et al. (2024) (CVPR 2024)	24.44	43.46	/	22.71	41.59	/
SCHEMA Niu et al. (2024) (ICLR 2024)	27.93	41.64	/	23.26	39.93	/
PlanLLM † Yang et al. (2025) (AAAI 2025)	26.74	42.97	/	27.08	46.96	/
MTID Zhou et al. (2025) (ICLR 2025)	28.52	44.44	56.46	24.89	44.54	57.46
LAP (Ours)	**65.8**	**82.53**	**81.49**	**55.31**	**78.1**	**79.46**

Table 4: Ablations on using text or visual observation for procedure planning on all datasets with $T = 3$ and $T = 4$. LAP-vo indicates using visual feature. The best numbers are in **bold**.

		T=3			T=4		
		SR↑	mAcc↑	mSIoU↑	SR↑	mAcc↑	mSIoU↑
CrossTask	LAP-vo	32.03	60.16	63.34	19.78	55.81	63.55
	LAP (Ours)	**54.3**	**81.71**	**77.09**	**31.87**	**73.36**	**72.97**
Coin	LAP-vo	17.54	38.39	44.83	13.2	36.88	45.03
	LAP (Ours)	**55.86**	**78.69**	**77.6**	**37.29**	**70.61**	**75.47**
NIV	LAP-vo	27.04	43.09	56.96	20.96	42.9	57.78
	LAP (Ours)	**65.8**	**82.53**	**81.49**	**55.31**	**78.1**	**79.46**

where E_{V_s} and E_{V_g} are the features of the start visual observation and the goal visual observation, respectively. Following previous work Wang et al. (2023b); Shi et al. (2025), E_{V_s} and E_{V_g} are extracted by using the visual encoder from Miech et al. (2020).

Table 4 shows the results on CrossTask, Coin and NIV dataset with time horizon $T = 3$ and $T = 4$. The results demonstrate that LAP with text descriptions in the diffusion model consistently and significantly outperforms LAP using only the visual observation (LAP-vo) across all metrics and time horizons. The improvements on longer horizons are not as large as on shorter horizons. This is not a limitation of using language embedding, the reason is that longer horizon increases the search space of intermediate actions. Baselines that use visual observations show the same trend, this validates the cause of performance degradation on longer horizons.

6 Conclusion

In this work, we introduce LAP, a Language-Aware Planning model for procedure planning in instructional videos. Our approach addresses the inherent ambiguity in visual observations by leveraging the expressiveness and distinctiveness of language descriptions in the latent space. We conduct experiments on procedure planning benchmarks: CrossTask, Coin and NIV. The experimental results show that LAP achieves SOTA performance on multiple metrics by a large margin. Ablation studies confirm the importance of using language features for procedure planning due to their distinctive representation, especially in cases where visual observations possess more ambiguities.

7 Disclose AI Use

In this work, we used generative AI tools to implement methods. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. Furthermore, generate synthetic data sets, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses are not applicable to this work. Additionally,

we used generative AI tools to create or edit software code, edit a research paper to improve readability and create or modify method figures. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by 1 author. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

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*, pp. 4575–4583, 2016.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

Jing Bi, Jiebo Luo, and Chenliang Xu. Procedure planning in instructional videos via contextual modeling and model-based policy learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15611–15620, 2021.

Sonia Castelo, Joao Rulff, Erin McGowan, Bea Steers, Guande Wu, Shaoyu Chen, Iran Roman, Roque Lopez, Ethan Brewer, Chen Zhao, et al. Argus: Visualization of ai-assisted task guidance in ar. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1313–1323, 2023.

Chien-Yi Chang, De-An Huang, Danfei Xu, Ehsan Adeli, Li Fei-Fei, and Juan Carlos Niebles. Procedure planning in instructional videos. In *European Conference on Computer Vision*, pp. 334–350. Springer, 2020.

Gautier Dagan, Frank Keller, and Alex Lascarides. Dynamic planning with a llm. *arXiv preprint arXiv:2308.06391*, 2023.

Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. Leveraging pretrained large language models to construct and utilize world models for model-based task planning. *Advances in Neural Information Processing Systems*, 36:79081–79094, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. *arXiv preprint arXiv:2311.17842*, 2023.

Wenlong Huang, Fei Xia, Dhruv Shah, Danny Driess, Andy Zeng, Yao Lu, Pete Florence, Igor Mordatch, Sergey Levine, Karol Hausman, et al. Grounded decoding: Guiding text generation with grounded models for embodied agents. *Advances in Neural Information Processing Systems*, 36:59636–59661, 2023.

Ruipeng Jia, Yanan Cao, Haichao Shi, Fang Fang, Pengfei Yin, and Shi Wang. Flexible non-autoregressive extractive summarization with threshold: How to extract a non-fixed number of summary sentences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 13134–13142, 2021.

Chuhan Jiao, Yao Wang, Guanhua Zhang, Mihai Bâce, Zhiming Hu, and Andreas Bulling. Diffgaze: A diffusion model for continuous gaze sequence generation on 360 {\deg} images. *arXiv preprint arXiv:2403.17477*, 2024.

Sanghwan Kim, Daoji Huang, Yongqin Xian, Otmar Hilliges, Luc Van Gool, and Xi Wang. Palm: Predicting actions through language models. In *European Conference on Computer Vision*, pp. 140–158. Springer, 2024.

Alex M Lamb, Anirudh Goyal ALIAS PARTH GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. *Advances in neural information processing systems*, 29, 2016.

Zhiheng Li, Wenjia Geng, Muheng Li, Lei Chen, Yansong Tang, Jiwen Lu, and Jie Zhou. Skip-plan: Procedure planning in instructional videos via condensed action space learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10297–10306, 2023.

Jiateng Liu, Sha Li, Zhenhailong Wang, Manling Li, and Heng Ji. A language-first approach for procedure planning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 1941–1954, 2023.

Yujie Lu, Pan Lu, Zhiyu Chen, Wanrong Zhu, Xin Eric Wang, and William Yang Wang. Multimodal procedural planning via dual text-image prompting. *arXiv preprint arXiv:2305.01795*, 2023.

Jianlan Luo, Charles Xu, Fangchen Liu, Liam Tan, Zipeng Lin, Jeffrey Wu, Pieter Abbeel, and Sergey Levine. Fmb: a functional manipulation benchmark for generalizable robotic learning. *The International Journal of Robotics Research*, 44(4):592–606, 2025.

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 IEEE/CVF international conference on computer vision*, pp. 2630–2640, 2019.

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/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9879–9889, 2020.

Kumaranage Ravindu Yasas Nagasinghe, Honglu Zhou, Malitha Gunawardhana, Martin Renqiang Min, Daniel Harari, and Muhammad Haris Khan. Why not use your textbook? knowledge-enhanced procedure planning of instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18816–18826, 2024.

Yulei Niu, Wenliang Guo, Long Chen, Xudong Lin, and Shih-Fu Chang. Schema: State changes matter for procedure planning in instructional videos. *arXiv preprint arXiv:2403.01599*, 2024.

Dhruvesh Patel, Hamid Eghbalzadeh, Nitin Kamra, Michael Louis Iuzzolino, Unnat Jain, and Ruta Desai. Pretrained language models as visual planners for human assistance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15302–15314, 2023.

Andreas Persson, Pedro Zuidberg Dos Martires, Luc De Raedt, and Amy Loutfi. Semantic relational object tracking. *IEEE Transactions on Cognitive and Developmental Systems*, 12(1):84–97, 2019.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Lei Shi and Andreas Bulling. Clad: Constrained latent action diffusion for vision-language procedure planning. *arXiv preprint arXiv:2503.06637*, 2025.

Lei Shi, Paul-Christian Bürkner, and Andreas Bulling. Inferring human intentions from predicted action probabilities. *arXiv preprint arXiv:2308.12194*, 2023.

Lei Shi, Paul-Christian Bürkner, and Andreas Bulling. ActionDiffusion: An Action-aware Diffusion Model for Procedure Planning in Instructional Videos. In *Proc. IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

Jiankai Sun, De-An Huang, Bo Lu, Yun-Hui Liu, Bolei Zhou, and Animesh Garg. Plate: Visually-grounded planning with transformers in procedural tasks. *IEEE Robotics and Automation Letters*, 7(2):4924–4930, 2022.

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/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1207–1216, 2019.

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

Hanlin Wang, Yilu Wu, Sheng Guo, and Limin Wang. Pdpp: Projected diffusion for procedure planning in instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14836–14845, 2023b.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

Te-Lin Wu, Alex Spangher, Pegah Alipoormolabashi, Marjorie Freedman, Ralph Weischedel, and Nanyun Peng. Understanding multimodal procedural knowledge by sequencing multimodal instructional manuals. *arXiv preprint arXiv:2110.08486*, 2021.

Haodong Yan, Zhiming Hu, Syn Schmitt, and Andreas Bulling. Gazemodiff: Gaze-guided diffusion model for stochastic human motion prediction. *arXiv preprint arXiv:2312.12090*, 2023.

Dejie Yang, Zijing Zhao, and Yang Liu. Planllm: Video procedure planning with refinable large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 9166–9174, 2025.

Ruihan Yang, Jiangjie Chen, Yikai Zhang, Siyu Yuan, Aili Chen, Kyle Richardson, Yanghua Xiao, and Deqing Yang. Selfgoal: Your language agents already know how to achieve high-level goals. *arXiv preprint arXiv:2406.04784*, 2024.

He Zhao, Isma Hadji, Nikita Dvornik, Konstantinos G Derpanis, Richard P Wildes, and Allan D Jepson. P3iv: Probabilistic procedure planning from instructional videos with weak supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2938–2948, 2022.

Luowei Zhou, Chenliang Xu, and Jason Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Yufan Zhou, Zhaobo Qi, Lingshuai Lin, Junqi Jing, Tingting Chai, Beichen Zhang, Shuhui Wang, and Weigang Zhang. Masked temporal interpolation diffusion for procedure planning in instructional videos. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Representation Learning*, volume 2025, pp. 46540–46560, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/731ff01ca533403062eaa4ef09608878-Paper-Conference.pdf.

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/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3537–3545, 2019.

APPENDIX

A IMPLEMENTATION DETAILS

A.1 VLM FINETUNING

We use InternVL3.5-8B as the VLM. Table 5 summarizes the full hyperparameter configuration. The base model is loaded in 4-bit precision using the nf4 data type with double quantization and bfloat16 compute dtype. All base-model parameters remain frozen and quantized throughout training, only the adapter parameters are updated. We adapt the model with LoRA, rank $r = 32$, scaling $\alpha = 64$, dropout 0.05, and no bias adaptation. LoRA adapters are inserted into the query, key, value, and output projections and into the gated MLP projections of every decoder layer. The vision encoder is kept frozen.

Each training example consists of 4 frames sampled uniformly from a video clip and is resized to 448×448 pixels to match the vision encoder's configuration. We prompt the model with a fixed natural-language instruction, *"Describe the action being performed in this video."* The substitution probability in scheduled sampling follows a linear schedule,

$$p_{\text{step}} = p_{\max} \cdot \min\left(1, \tfrac{\text{step}}{T}\right), \qquad p_{\max} = 0.25, \tag{9}$$

with T the total number of optimizer steps, so that training begins as pure teacher forcing and gradually exposes the model to its own predictions. We set λ in Equation 3 to 0.1 We use AdamW with learning rate 2×10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and zero weight decay on one-dimensional parameters (biases and normalization weights). The learning rate follows a cosine decay schedule with 1 warmup epoch, decaying from 2×10^{-4} to a floor of 10^{-6}, computed over 5 total epochs. Gradients are clipped to a maximum global ℓ_2-norm of 1.0. We use a batch size of 12 with no gradient accumulation (update frequency 1). Training and evaluation were run on a single NVIDIA H200 GPU. Model, optimizer, and scheduler state are checkpointed after every epoch, together with a resumable training step counter, allowing exact continuation. We release the full training configuration (Table 5) and random seed to facilitate exact reproduction of the reported numbers.

Table 5: Hyperparameters for InternVL3.5-8B finetuning.

Hyperparameter	**Value**
Base model	OpenGVLab/InternVL3_5-8B-HF
Quantization	4-bit NF4, double quant, bf16 compute
LoRA rank r / α / dropout	32 / 64 / 0.05
LoRA target modules	q,k,v,o,gate,up,down proj (decoder only)
Vision encoder	frozen, not adapted
Frames per clip K	4
Frame resolution	448×448
Optimizer	AdamW (β_1=0.9, β_2=0.999, ϵ=10^{-8})
Learning rate	2×10^{-4} (cosine decay, floor 10^{-6})
Warmup	1 epoch
Weight decay	0
Gradient clipping (global ℓ_2)	1.0
Batch size / gradient accumulation	12 / 1
Epochs	5
Scheduled-sampling schedule	linear, $p_{\max} = 0.25$
Sampling temperature / top-p	0.7 / 0.95
Regularization weight λ	0.1
Regularizer	BiLSTM (2 layers, hidden 128) + MLP
Decoding (evaluation)	max 32 tokens
Repetition penalty / no-repeat n-gram	1.3 / 3

A.2 LLM FINETUNING

We use Qwen3.5-0.8B as the LLM in planning. The base model is loaded in 4-bit nf4 precision with double quantization and a bfloat16 compute dtype, and a LoRA adapter is trained on top of the frozen quantized base. The adapter targets the query/key/value/ output projections and the gated MLP branches of every decoder layer. The vision tower is kept frozen throughout. All hyperparameters are summarized in Table 6. At evaluation time we decode with a repetition penalty of 1.3, a no-repeat 3-gram constraint, and a cap of 32 new tokens.

Table 6: Hyperparameters for finetuning Qwen3.5-0.8B.

Hyperparameter	**Value**
Base model	Qwen/Qwen3.5-0.8B
Quantization	4-bit NF4, double quantization
Compute / master dtype	bfloat16
LoRA rank r	32
LoRA α	64
LoRA dropout	0.05
Optimizer	AdamW (β_1=0.9, β_2=0.999, ϵ=10^{-8})
Weight decay	0.0 (no decay on 1-D / bias / norm params)
LR schedule	cosine, 1 warmup epoch
Peak / warmup-start / final LR	2×10^{-4} / 1×10^{-6} / 1×10^{-6}
Per-device batch size	8
Gradient accumulation steps	8
Effective batch size	64
Gradient clipping (max value)	1.0
Epochs	5
Random seed	0

A.3 DIFFUSION MODEL

We use the same hyperparameters and training strategy for both base planning model and language-conditioned model. We use AdamW as the optimizer and set the hidden size of UNet to 256 for all datasets, with a consistent batch size of 128. The training configurations are as follows: for Crosstask, we train for 120 epochs (200 steps/epoch) with 200 diffusion steps; for Coin, 800 epochs (200 steps/epoch) with 200 diffusion steps; and for NIV, 130 epochs (50 steps/epoch) with 50 diffusion steps. All models employ a linear learning rate warm-up for the first 20 epochs, increasing to $5e^{-4}$ for Crosstask and $1e^{-4}$ for Coin. For NIV, the learning rate warms up to $3e^{-4}$ over the first 90 epochs. A learning rate decay is applied to Crosstask (decay by 0.5 every 5 epochs for the last 30 epochs) and Coin (decay by 0.5 over the last 50 epochs). For language-conditioned model, before clustering, we pool each plan's embedding sequence into a single D-dimensional vector by averaging over the temporal dimension, the clustering operates on one feature vector per plan rather than per individual intermediate action. Clustering is performed once on the pooled training-set features, test-time plans are never used to fit the centroids and are only assigned to their nearest centroid post hoc . We run for a maximum of 100 iterations, terminating early once the largest centroid displacement across all clusters between consecutive iterations falls below a tolerance of 10^{-4}. At inference time, a held-out plan's pooled intermediate-action embedding is assigned to its nearest fitted centroid under the same distance metric used during fitting; the fitted centroids themselves are not updated using test-time data. The number of clusters K and the dimensionality D of the pooled embeddings are reported in Table 7 for each dataset.

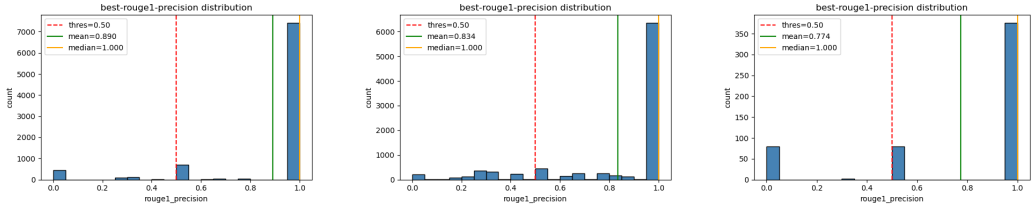
B ADDITIONAL ABLATION STUDIES

B.1 EFFECT OF ROUGE THRESHOLD

Our method LAP employs a ROUGE-1 score threshold to determine if a generated description accurately represents a video action. A higher threshold imposes a stricter condition for lexical over-

Table 7: K-means clustering hyperparameters.

Dataset	Number of clusters K	Embedding dim. D	Embedding source
CrossTask	18	512	S3D
Coin	105	512	S3D
NIV	5	512	S3D

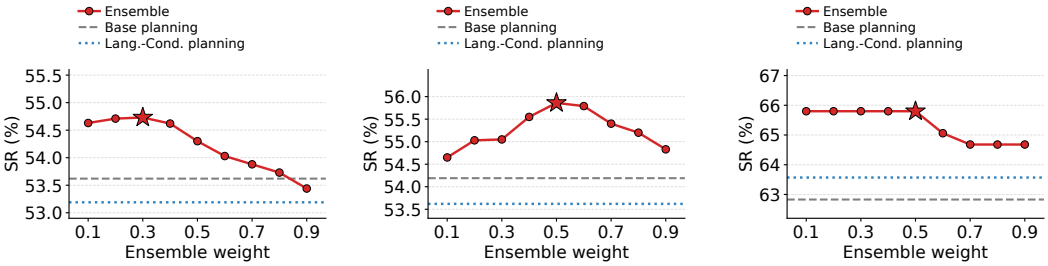


(a) CrossTask (b) Coin (c) NIV

Figure 5: ROUGE-1 score distribution of LAP on CrossTask, Coin and NIV dataset.

lap with the ground truth, i.e. more matching words between the generated description and the ground truth. Figure 5 shows the distribution of ROUGE-1 scores for generated descriptions on the CrossTask, COIN, and NIV datasets. We observe that the majority of the samples has perfect ROUGE-1 scores, which indicates that the VLM already generates correct descriptions of action in most videos.

B.2 Effect of Ensemble Weight



(a) CrossTask (b) COIN (c) NIV

Figure 6: Effect of the ensemble weight on (a) CrossTask, (b) COIN and (c) NIV at $T = 3$. The red curve covers the weight of the language-conditioned planning model. The dashed and dotted lines are base planning and language-conditioned planning alone. The star marks the weight that has the highest SR.

Figure 6 ablate the weight λ_p used in ensemble inference on all three datasets. We test the weight from 0.1 to 0.9 with step size 0.1. Across all datasets the ensemble inference outperforms both base planning and language-conditioned planning for every weight in $[0.1, 0.8]$. The best weight is 0.3 on CrossTask, but 0.5 is still higher than the other two planning models. On Coin and NIV, both achieves highest SR when the weight is 0.5. The ensemble improves over base planning on CrossTask and COIN even though the language-conditioned model is weaker than base planning, this indicates that two models make complementary predictions rather than one simply dominating the other.